

Matrix

Notes

| | | |
|----------------|--------------------------------|--|
| Output Created | 20-OCT-2020 15:56:54 | |
| Comments | | |
| Input | Data | C:\Users\Rizwan\Desktop\SPSS Data\Data-sav (CSR & Marketing).sav |
| | Active Dataset | DataSet1 |
| | Filter | <none> |
| | Weight | <none> |
| | Split File | <none> |
| | N of Rows in Working Data File | 856 |

Syntax

```
matrix.  
get dat/file = */variables =  
Operational_Performance Pro_Env_Strgy  
Eco_Innovation /names = vnames/missing =  
9999.  
compute ninit = nrow(dat).  
get dat/file = */variables =  
Operational_Performance Pro_Env_Strgy  
Eco_Innovation /names = vnames/missing =  
omit.  
get tmp/file = */variables =  
Operational_Performance /names =  
yname/missing = omit.  
get tmp2/file = */variables = Pro_Env_Strgy  
/names = xname/missing = omit.  
get tmp/file = */variables = Eco_Innovation  
/names = mnames/missing = omit.  
get tmp/file = */variables = w999999t z999999t  
v999999t q999999t.  
compute wname=tmp(1,1).  
do if (wname = ' ').  
compute wname = 'xxx'.  
end if.  
compute zname=tmp(1,2).  
do if (zname = ' ').  
compute zname = 'xxx'.  
end if.  
compute vname=tmp(1,3).  
do if (vname = ' ').  
compute vname = 'xxx'.  
end if.  
compute qname=tmp(1,4).  
do if (qname = ' ').  
compute qname = 'xxx'.  
end if.  
compute n = nrow(dat).  
compute p0=-.322232431088.  
compute p1 = -1.  
compute p2 = -.342242088547.  
compute p3 = -.0204231210245.
```

```

compute p4 = -.0000453642210148.
compute q0 = .0993484626060.
compute q1 = .588581570495.
compute q2 = .531103462366.
compute q3 = .103537752850.
compute q4 = .0038560700634.
compute badend = 0.
compute priorlo = -9999999.
compute priorhi = 9999999.
compute criterr = 0.
compute cluster = 0.
compute clsdmy = 0.
compute jndich = 0.
compute wvdich=0.
compute zqdich=0.
compute mod74dic=0.
compute warnrep=0.
compute contrast = trunc( 0 ).
do if (contrast > 2 or contrast < 0).
compute contrast = 0.
end if.
compute booterr = 0.
compute effsize = ( 0 = 1).
compute note = make(10,1,0).
compute notes = 1.
compute iterr = 0.
compute clsmtn = 0.
compute quantile = ( 0 = 1).
compute mmodval= 999.
compute jn = ( 0 = 1).
compute covcoeff=( 0 =1).
compute center = ( 0 = 1).
compute detail = ( 1 = 1).
compute coeffci = ( 1 = 1).
compute conf = 95.
compute longname = ( 1 =1).
compute bconoff=( 0 <> 1).
compute covmy = trunc( 0 ).
do if (covmy < 0 or covmy > 2).
compute covmy = 0.

```

```

end if.
do if (trunc(conf) >= 100 or (trunc(conf) <=
50)).
compute conf = 95.
compute note(notes,1) = 1.
compute notes = notes + 1.
end if.
do if (n < ninit).
compute nmiss = ninit-n.
compute note(notes,1) = 11.
compute notes = notes + 1.
end if.
compute errs = 0.
compute quantd = {0,0,0,0,0,0}.
compute quantc = {0,0,0,0,0,0}.
compute mcheck = 0.
compute ttt = 0.
compute plot = trunc( 0 ).
compute runerrs = make(50,1,0).
compute model = trunc( 4 ).
compute normal = 1.
compute ws= ( 0 =1).
compute olsdichm= ( 0 =1).
compute olsdichy= ( 0 =1).
compute mcx=trunc( 0 ).
compute mcm=trunc( 0 ).
do if (ws = 1).
do if (effsize = 1).
compute note(notes,1) = 19.
compute notes = notes + 1.
compute effsize=0.
end if.
do if (normal=1).
compute note(notes,1) = 16.
compute notes = notes + 1.
compute normal=0.
end if.
end if.
do if (model <> 4).
compute ws = 0.

```



```

end if.
do if (model > 0 and model < 4 and covmy <>
0).
compute covmy=0.
compute note(notes,1) = 33.
compute notes = notes + 1.
end if.
print/space=0.
do if (jn = 1 and model <> 1 and model <> 3).
compute note(notes,1) = 7.
compute notes = notes + 1.
end if.
do if ((model > 76) or (model < 1)).
compute model = 77.
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 19.
end if.
compute itprob=0.
do if (mcx < 0).
compute mcx=0.
end if.
do if (mcx > 4).
compute mcx=0.
end if.
do if (mcm < 0).
compute mcm=0.
end if.
do if (mcm > 4).
compute mcm=0.
end if.
do if (mcx > 0 and mcm > 0).
compute errs=errs+1.
compute runerrs(errs,1)=34.
compute criterr=1.
end if.
do if ((model <> 1 and model <> 4) and (mcx >
0 or mcm > 0)).
compute errs=errs+1.
compute runerrs(errs,1)=35.

```

```
compute criterr=1.  
end if.  
do if ((model = 4) and (mcm > 0)).  
compute errs=errs+1.  
compute runerrs(errs,1)=41.  
compute criterr=1.  
end if.  
do if ((mcx > 0) and jn <> 0).  
compute jn=0.  
compute note(notes,1) = 23.  
compute notes = notes + 1.  
end if.  
do if (mmodval <> 999 and mcm > 0).  
compute note(notes,1) = 24.  
compute notes = notes + 1.  
end if.  
compute toteff = 0.  
compute toteff = (( 0 = 1)*( 4 = 4 or 4 = 6)).  
compute varorder = 2.  
compute hc3 = ( 0 <> 0).  
compute cname = 'xxx'.  
compute centvar = {'xxx'}.  
compute nmeds = ncol(mnames).  
do if (longname=0).
```

compute criterr = 1.

compute errs = errs+1.

compute runerrs(errs,1) = 33.

1

2

3

```
end if.  
compute modelm =  
{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,6;  
1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,7;  
1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8;  
1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,9;  
1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,10;  
1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,11;  
1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,0,12;  
1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,13;  
0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,14;  
0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,15;  
0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,16;  
0,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,17;  
0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,18;  
0,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,19;  
0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,20;  
1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,21;
```

1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,7,22;
1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,23;
1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,24; 1,1,1,1

,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,25;
1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,26;
1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,27;
1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,7,28;
1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,7,29;
1,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,30;
1,1,0,1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,31;
1,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,32;
1,1,1,1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,33;
1,1,1,1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,34;
1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,35;
1,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,36;
1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,37;
1,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,38;
1,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,39;
1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,40;
1,0,0,1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,41;
1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,42;
1,0,0,1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,43;
1,0,0,1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,44;
1,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,45;
1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,46;
1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,47;
1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,48;
1,1,0,1,1,0,1,1,0,0,0,

0,0,0,0,0,0,0,0,49;
1,1,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,50;
1,1,1,1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,51;
1,1,1,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,52;
1,1,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,53;
1,1,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,54;
1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,55;
1,1,1,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,56;
1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,57;
1,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,58;

```
1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,59;  
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,60;  
1,1,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,61;  
1,1,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,62;  
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,0,0,0,0,63;  
1,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,64;  
1,0,0,1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,65;  
1,0,0,1,0,0,0,0,0,1,0,0,1,0,0,0,0,0,0,66;  
1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,67;  
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,68;  
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,69;  
1,0,0,1,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,70;  
1,0,0,1,0,0,1,0,0,1,0,0,1,1,1,0,0,0,0,71;  
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,1,1,0,0,72;  
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,1,1,0,0
```

```
,73; 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,74;
```

```
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,75;
```

```
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,1,0,0,0,76;
```

```
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,77}.
```

```
compute wm = modelm(model, 1).
```

```
compute zm = modelm(model, 2).
```

```
compute wzm = modelm(model, 3).
```

```
compute vy = modelm(model, 4).
```

```
compute qy = modelm(model, 5).
```

```
compute vqy = modelm(model, 6).
```

```
compute wy = modelm(model, 7).
```

```
compute zy = modelm(model, 8).
```

```
compute wzy = modelm(model, 9).
```

```
compute vxy = modelm(model, 10).
```

```
compute qxy = modelm(model, 11).
```

```
compute vqxy = modelm(model, 12).
```

```
compute wmy = modelm(model, 13).
```

```
compute wvmy = modelm(model, 14).
```

```
compute wvxy = modelm(model, 15).
```

```
compute zmy = modelm(model,16).
```

```
compute wzmy = modelm(model,17).
```

```
compute xmy = modelm(model,18).
```

```
compute imm=modelm(model,19).
```

```
do if (model = 74).
```

```

compute vdich=0.
end if.
do if (ncol(xname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 20.
compute criterr = 1.
end if.
do if (ncol(mnames) > 10).
compute errs = errs+1.
compute runerrs(errs,1) = 39.
compute criterr = 1.
end if.
do if (ncol(yname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 21.
compute criterr = 1.
end if.
do if (( 999 =0) and ( 999 =0) and (model=3 or
model=2) and (contrast=1)).
compute errs = errs+1.
compute runerrs(errs,1) = 31.
compute criterr = 1.
end if.
compute xlist = (wm or zm or wzm or wy or zy
or wzy or vxy or qxy or vqxy or wvxy or xmy).
compute mlist = (vy or qy or vqy or zmy or wmy
or wzmy or xmy or (model < 4)).
compute bad=0.
do if (criterr = 0).
compute werr = 0.
compute verr = 0.
compute qerr = 0.
compute zerr = 0.
compute yerr = 1.
compute xerr = 1.
compute wlist = (wm or wzm or wy or wzy or
wm or wvmy or wvmy or wvxy or wzmy).
do if (wlist = 1 and wname = 'xxx').
compute werr = 1.
compute wlist = 0.

```



```

compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (wlist = 1 and ((wname = qname) or
(wname = vname) or (wname = zname) or
(wname = xname) or (wname = yname))).
compute werr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 12.
end if.
compute zlist = (zm or wzm or zy or wzy or
zmy or wzmy).
do if (zlist = 1 and zname = 'xxx').
compute zerr = 1.
compute zlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (zlist = 1 and ((zname = qname) or
(zname = vname) or (zname = wname) or
(zname = xname) or (zname = yname))).
compute zerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 13.
end if.
compute qlist = (qy or vqy or qxy or vqxy).
do if (qlist = 1 and qname = 'xxx').
compute qerr = 1.
compute qlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (qlist = 1 and ((qname = zname) or
(qname = vname) or (qname = wname) or
(qname = xname) or (qname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 14.
end if.

```

```

compute vlist = (vy or vqy or vxy or vqxy or
wvmy or wvxy).
do if (vlist = 1 and vname = 'xxx').
compute verr = 1.
compute vlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (vlist = 1 and ((vname = zname) or
(vname = qname) or (vname = wname) or
(vname = xname) or (vname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 15.
end if.
do if (wlist = 0 and wname <> 'xxx').
compute werr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 8.
end if.
do if (zlist = 0 and zname <> 'xxx').
compute zerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 9.
end if.
do if (qlist = 0 and qname <> 'xxx').
compute qerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 10.
end if.
do if (vlist = 0 and vname <> 'xxx').
compute verr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 11.
end if.
do if (hc3 = 1).
compute note(notes,1) = 3.
compute notes = notes+1.
end if.
compute alpha2 = (1-(conf/100))/2.

```

```

compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute cons = make(n,1,1).
compute temp = (n*sscp(dat))-
(t(csum(dat))*(csum(dat))).
compute temp = temp/(n*(n-1)).
compute temp = csum((((diag(temp))=0)).
compute temp2=1.
do if (ws = 1).
compute temp2 = (n*sscp(tmp2))-
(t(csum(tmp2))*(csum(tmp2))).
compute temp2 = temp2/(n*(n-1)).
end if.
do if ((temp > 0 and ws = 0) or (temp > 0 and
ws = 1 and temp2 > 0)).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 27.
end if.
compute nmeds = ncol(mnames).
compute mcmats=ident(nmeds*2).
compute mccoeff=make(nmeds*2,1,0).
compute sobel = make(nmeds,4,-999).
do if (model = 6 and nmeds > 4).
compute errs = errs+1.
compute runerrs(errs,1)=2.
end if.
do if (model < 4 and nmeds > 1).
compute errs = errs+1.
compute runerrs(errs,1) = 3.
end if.
compute nmods = (model = 74).
compute bad = 0.
compute intcnt = 1.
compute modvals = 0.
compute modvalsd = 0.
compute yintemp = {'int_1', 'int_2', 'int_3',
'int_4', 'int_5', 'int_6', 'int_7', 'int_8', 'int_9',

```

```

'int_10', 'int_11', 'int_12', 'int_13', 'int_14',
'int_15'}.
compute yintemp = {yintemp, 'int_16', 'int_17',
'int_18', 'int_19', 'int_20', 'int_21', 'int_22',
'int_23', 'int_24', 'int_25', 'int_26', 'int_27',
'int_28'}.
compute yintemp = {yintemp, 'int_29', 'int_30',
'int_31', 'int_32', 'int_33', 'int_34', 'int_35',
'int_36', 'int_37', 'int_38', 'int_39', 'int_40'}.
compute yintemp = {yintemp, 'int_41', 'int_42',
'int_43', 'int_44', 'int_45', 'int_46', 'int_47',
'int_48', 'int_49', 'int_50', 'int_51', 'int_52'}.
compute yintemp = {yintemp, 'int_53', 'int_54',
'int_55', 'int_56', 'int_57', 'int_58', 'int_59',
'int_60', 'int_61', 'int_62', 'int_63', 'int_64'}.
compute
cntname={'(C1)';'(C2)';'(C3)';'(C4)';'(C5)';'(C6)';'(
C7)';'(C8)';'(C9)';'(C10)';'(C11)';'(C12)';'(C13)';(
C14)';'(C15)';'(C16)';'(C17)'}.
compute
cntname={cntname;'(C18)';'(C19)';'(C20)';'(C21
)';'(C22)';'(C23)';'(C24)';'(C25)';'(C26)';'(C27)';(
C28)';'(C29)';'(C30)';'(C31)'}.
compute
cntname={cntname;'(C32)';'(C33)';'(C34)';'(C35
)';'(C36)';'(C37)';'(C38)';'(C39)';'(C40)';'(C41)';(
C42)';'(C43)';'(C44)';'(C45)'}.
compute
cntname={cntname;'(C46)';'(C47)';'(C48)';'(C49
)';'(C50)';'(C51)';'(C52)';'(C53)';'(C54)';'(C55)';(
C56)';'(C57)';'(C58)';'(C59)'}.
compute
cntname={cntname;'(C60)';'(C61)';'(C62)';'(C63
)';'(C64)';'(C65)';'(C66)';'(C67)';'(C68)';'(C69)';(
C70)';'(C71)';'(C72)';'(C73)'}.
compute
cntname={cntname;'(C74)';'(C75)';'(C76)';'(C77
)';'(C78)';'(C79)';'(C80)';'(C81)';'(C82)';'(C83)';(
C84)';'(C85)';'(C86)';'(C87)'}.

```

```

compute
cntname={cntname;'(C88)';'(C89)';'(C90)';'(C91
)';'(C92)';'(C93)';'(C94)';'(C95)';'(C96)';'(C97)';'(
C98)';'(C99)';'(C100)';'(C101)'}.
compute
cntname={cntname;'(C102)';'(C103)';'(C104)';'(
C105)'}.
compute apathnam={'a path'; 'a1 path'; 'a2
path'; 'a3 path'; 'a4 path'; 'a5 path'; 'a6 path';
'a7 path'; 'a8 path'; 'a9 path'; 'a10 path'}.
compute bpathnam={'b path'; 'b1 path'; 'b2
path'; 'b3 path'; 'b4path'; 'b5path'; 'b6 path'; 'b7
path'; 'b8 path'; 'b9 path'; 'b10 path'}.
compute modvnm = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute modvnm2 = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute mlab = { ' M1 ='; ' M2 ='; ' M3 ='; '
M4 ='; ' M5 ='; ' M6 ='; ' M7 ='; ' M8 ='; '
M9 ='; ' M10 ='}.
compute m = make(n,nmeds,1).
compute mmat = make(16,nmeds,0).
compute ymat = make(8,nmeds,0).
compute deco = make(10,1,0).
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
compute iterate = abs(trunc( 10000 )).
compute converge = abs( 0.00000001 ).
compute boot = abs(trunc( 5000 )).
do if (ws=1 and 0 > 0).
compute note(notes,1)=17.
compute notes=notes+1.
end if.
compute mc=(abs(trunc( 0 ))*(1-ws)).
do if (mc > 0 and model > 5).
do if (boot = 0).
compute boot=mc.
end if.

```

```

compute mc=0.
compute note(notes,1)=12.
compute notes = notes+1.
end if.
compute adjust = 0.
do if (mc > 0 and (model > 3 and model < 6)).
compute boot=0.
compute bconoff=0.
end if.
do if (boot > 0 and mc > 0).
compute mc = 0.
end if.
compute savboot = 0.
compute saveboot = ( 0 = 1).
do if (ws=1 and saveboot=1).
compute saveboot=0.
compute note(notes,1) = 20.
compute notes = notes + 1.
compute normal=0.
end if.
do if (saveboot =1 and boot > 0 and model >
3).
compute savboot = 1.
end if.
do if (boot <> 0 or mc <> 0).
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
loop.
compute cilow = rnd(bootsz*(1-(conf/100))/2).
compute cihigh =
trunc((bootsz*(conf/100)+(bootsz*(1-
(conf/100))/2)))+1.
do if (cilow < 1 or cihigh > bootsz).
compute
bootsz=trunc((bootsz+1000)/1000)*1000.
compute adjust = 1.
end if.
end loop if (cilow gt 0 and cihigh le bootsz).

```

```

compute boot=bootsz.
do if (mc > 0).
compute mc = bootsz.
end if.
do if (boot > 0 and mc > 0 and (model > 3 and
model < 6)).
compute boot=0.
end if.
do if (adjust = 1 and boot > 0).
compute note(notes,1)=6.
compute notes = notes+1.
end if.
do if (adjust = 1 and mc > 0).
compute note(notes,1)=13.
compute notes = notes+1.
end if.
end if.
do if (model = 6 and nmeds > 1).
compute mmpaths =
make((nmeds+2),(nmeds+2),0).
do if (nmeds = 2).
compute indboot = make(boot+1, 3, 999).
else if (nmeds = 3).
compute indboot = make(boot+1, 7, 999).
else if (nmeds = 4).
compute indboot = make(boot+1, 15, 999).
end if.
compute indlbl = {'Total:','Ind1 :','Ind2 :','Ind3
:','Ind4 :','Ind5 :','Ind6 :','Ind7 :','Ind8 :','Ind9
:','Ind10:','Ind11:','Ind12:','Ind13:','Ind14:','
Ind15:'}.
compute indlbl2 = {'Ind1','Ind2','Ind3','Ind4';
'Ind5','Ind6','Ind7','Ind8','Ind9','Ind10';
'Ind11','Ind12','Ind13','Ind14','Ind15'}.
compute indces = make(boot+1, 4, 999).
end if.
do if (model < 4).
compute boot = 0.
compute cmat = make(10,1,0).
compute zmat = make(10,1,0).

```

```

end if.
compute nvarch = make(1,ncol(dat),0).
compute wmatch = 0.
compute zmatch = 0.
compute vmatch = 0.
compute qmatch = 0.
compute mmatch=0.
compute minprobe=0.
compute maxprobe=0.
loop i = 1 to ncol(vnames).
do if (vnames(:,i)=yname).
compute y = dat(:,i).
compute nvarch(1,i)=1.
compute yerr = 0.
do if ((yname = xname) or (yname = wname) or
(yname = zname) or (yname = vname) or
(yname = qname)).
compute errs = errs+1.
compute runerrs(errs,1)=17.
end if.
end if.
do if (vnames(:,i)=xname).
compute x = dat(:,i).
compute nvarch(1,i)=1.
compute xdich = 1.
compute xerr=0.
loop jj = 1 to n.
do if ((x(jj,1) <> cmin(x)) and (x(jj,1) <>
cmax(x))).
compute xdich = 0.
break.
end if.
end loop.
compute xmean = csum(x)/n.
do if (center = 1 and (model < 4 or xlist > 0)
and mcx = 0).
compute meanvec = make(n,1, xmean).
compute x = x-meanvec.
compute centvar = {centvar, xname}.
end if.

```



```

compute xmean = csum(x)/n.
compute tmp = x-(cons*xmean).
compute xsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
do if (xdich = 0).
compute quantc(1,6) = 1.
compute matx = {xmean-xsd; xmean;
xmean+xsd}.
do if ( 999 = 999 and quantile = 0).
do if ((matx(1,1) < cmin(x)) and model = 74).
compute matx(1,1)=cmin(x).
compute minprobe=1.
end if.
do if ((matx(3,1) > cmax(x)) and model = 74).
compute matx(3,1)=cmax(x).
compute maxprobe=1.
end if.
end if.
do if (quantile = 1).
compute quantd(1,6) = 1.
compute quantc(1,6) = 0.
compute tmp = x.
compute tmp(GRADE(x)) = x.
compute matx =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (xdich = 1).
compute matx = {cmin(x); cmax(x)}.
do if (model=74).
compute matx=cmin(x).
compute mod74dic=1.
end if.
end if.
do if ( 999 <> 999).
compute matx = 999.
compute quantd(1,6) = 0.
compute quantc(1,6) = 0.
end if.

```

```

end if.
do if (werr = 0 and wlist = 1).
do if (vnames(:,i)=wname).
compute werr = 0.
compute wmatch = 1.
compute w = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(w)/n).
compute w = w-meanvec.
compute centvar = {centvar, wname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute wmean = csum(w)/n.
compute tmp = w-(cons*wmean).
compute wsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute wdich = 1.
loop jj = 1 to n.
do if ((w(jj,1) <> cmin(w)) and (w(jj,1) <>
cmax(w))).
compute wdich = 0.
break.
end if.
end loop.
do if (model = 3).
compute jndich=wdich.
compute jnmin=cmin(w).
compute jnmax=cmax(w).
end if.
do if (wdich = 0).
compute matw = {wmean-wsd; wmean;
wmean+wsd}.
do if ( 999 = 999 and quantile = 0).
do if (matw(1,1) < cmin(w)).
compute matw(1,1)=cmin(w).
compute minprobe=1.
end if.
do if (matw(3,1) > cmax(w)).
compute matw(3,1)=cmax(w).
compute maxprobe=1.

```

```

end if.
end if.
compute quantc(1,1) = 1.
do if (quantile = 1).
compute quantd(1,1) = 1.
compute quantc(1,1) = 0.
compute tmp = w.
compute tmp(GRADE(w)) = w.
compute matw =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (wdich = 1).
compute matw = {cmin(w); cmax(w)}.
compute wvdich=1.
compute cmaw=cmax(w).
compute cminw=cmin(w).
end if.
do if ( 999 <> 999).
compute matw = 999.
compute quantd(1,1) = 0.
compute quantc(1,1) = 0.
end if.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm(1,1)=wname.
compute modmatp(1,1) = 1.
end if.
end if.
do if (zerr = 0 and zlist = 1).
do if (vnames(:,i)=zname).
compute zerr = 0.
compute zmatch = 1.
compute z = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(z)/n).
compute z = z-meanvec.
compute centvar = {centvar, zname}.

```

```

end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute zmean = csum(z)/n.
compute tmp = z-(cons*zmean).
compute zsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute zdich = 1.
loop jj = 1 to n.
do if ((z(jj,1) <> cmin(z)) and (z(jj,1) <>
cmax(z))).
compute zdich = 0.
break.
end if.
end loop.
do if (zdich = 0).
compute matz = {zmean-zsd; zmean;
zmean+zsd}.
do if ( 999 = 999 and quantile = 0).
do if (matz(1,1) < cmin(z)).
compute matz(1,1)=cmin(z).
compute minprobe=1.
end if.
do if (matz(3,1) > cmax(z)).
compute matz(3,1)=cmax(z).
compute maxprobe=1.
end if.
end if.
compute quantc(1,2) = 1.
do if (quantile = 1).
compute quantd(1,2) = 1.
compute quantc(1,2) = 0.
compute tmp = z.
compute tmp(GRADE(z)) = z.
compute matz = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (zdich = 1).
compute matz = {cmin(z); cmax(z)}.

```

```

compute cmaxz=cmax(z).
compute cminz=cmin(z).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matz = 999.
compute quantd(1,2) = 0.
compute quantc(1,2) = 0.
end if.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm(1,2)=zname.
compute modmatp(1,2) = 1.
end if.
end if.
do if (verr = 0 and vlist = 1).
do if (vnames(:,i)=vname).
compute verr = 0.
compute vmatch = 1.
compute v = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(v)/n).
compute v = v-meanvec.
compute centvar = {centvar, vname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute vmean = csum(v)/n.
compute tmp = v-(cons*vmean).
compute vsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute vdich = 1.
loop jj = 1 to n.
do if ((v(jj,1) <> cmin(v)) and (v(jj,1) <>
cmax(v))).
compute vdich = 0.
break.
end if.
end loop.
do if (vdich = 0).

```

```

compute matv = {vmean-vsd; vmean;
vmean+vsd}.
do if ( 999 = 999 and quantile = 0).
do if (matv(1,1) < cmin(v)).
compute matv(1,1)=cmin(v).
compute minprobe=1.
end if.
do if (matv(3,1) > cmax(v)).
compute matv(3,1)=cmax(v).
compute maxprobe=1.
end if.
end if.
compute quantc(1,3) = 1.
do if (quantile = 1).
compute quantd(1,3) = 1.
compute quantc(1,3) = 0.
compute tmp = v.
compute tmp(GRADE(v)) = v.
compute matv = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (vdich = 1).
compute matv = {cmin(v); cmax(v)}.
compute wvdich=1.
compute cmaxv=cmax(v).
compute cminv=cmin(v).
end if.
do if ( 999 <> 999).
compute matv = 999.
compute quantd(1,3) = 0.
compute quantc(1,3) = 0.
end if.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm(1,3)=vname.
compute modmatp(1,3) = 1.
end if.
end if.

```

```

do if (qerr = 0 and qlist = 1).
do if (vnames(:,i)=qname).
compute qerr = 0.
compute qmatch = 1.
compute q = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(q)/n).
compute q = q-meanvec.
compute centvar = {centvar, qname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute qmean = csum(q)/n.
compute tmp = q-(cons*qmean).
compute qsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute qdich = 1.
loop jj = 1 to n.
do if ((q(jj,1) <> cmin(q)) and (q(jj,1) <>
cmax(q))).
compute qdich = 0.
break.
end if.
end loop.
do if (qdich = 0).
compute matq = {qmean-qsd; qmean;
qmean+qsd}.
do if ( 999 = 999 and quantile = 0).
do if (matq(1,1) < cmin(q)).
compute matq(1,1)=cmin(q).
compute minprobe=1.
end if.
do if (matq(3,1) > cmax(q)).
compute matq(3,1)=cmax(q).
compute maxprobe=1.
end if.
end if.
compute quantc(1,4) = 1.
do if (quantile = 1).
compute quantd(1,4) = 1.
compute quantc(1,4) = 0.

```

```

compute tmp = q.
compute tmp(GRADE(q)) = q.
compute matq = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (qdich = 1).
compute matq = {cmin(q); cmax(q)}.
compute cmaxq=cmax(q).
compute cminq=cmin(q).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matq = 999.
compute quantd(1,4) = 0.
compute quantc(1,4) = 0.
end if.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm(1,4)=qname.
compute modmatp(1,4) = 1.
end if.
end if.
do if (vnames(:,i)=cname).
compute cld = dat(:,i).
compute cvname = vnames(:,i).
compute nvarch(1,i)=1.
compute clsmth = 1.
end if.
loop j = 1 to ncol(mnames).
do if (vnames(:,i)=mnames(1,j)).
compute mmatch = mmatch + 1.
compute m(:,j)=dat(:,i).
do if (center = 1 and nvarch(1,i) = 0 and mlist >
0 and mcm = 0).
compute meanvec = make(n,1,csum(m(:,j))/n).
compute m(:,j) = m(:,j)-meanvec.
compute centvar = {centvar, mnames(1,j)}.
end if.

```



```

compute nvarch(1,i)=1.
compute dichm = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute dichm = 0.
break.
end if.
end loop.
do if ((dichm = 1) and (olsdichm = 1)).
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichm = 1 and model > 3 and mcheck = 0
and olsdichm = 0).
compute errs= errs+1.
compute runerrs(errs,1) = 1.
compute mcheck = 1.
end if.
do if ((model <= 3) and (ncol(mnames)=1)).
compute nmods = nmods + 1.
compute mmean = csum(m(:,j))/n.
compute tmp = m(:,j)-(cons*mmean).
compute msd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute mdich = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute mdich = 0.
break.
end if.
end loop.
do if (model = 1).
compute jndich=mdich.
compute jnmin=cmin(m(:,j)).
compute jnmax=cmax(m(:,j)).
end if.
do if (mdich = 0).
compute matm = {mmean-msd; mmean;
mmean+msd}.

```

```

do if ( 999 = 999 and quantile = 0).
do if (matm(1,1) < cmin(m(:,j))).
compute matm(1,1)=cmin(m(:,j)).
compute minprobe=1.
end if.
do if (matm(3,1) > cmax(m(:,j))).
compute matm(3,1)=cmax(m(:,j)).
compute maxprobe=1.
end if.
end if.
compute quantc(1,5) = 1.
do if (quantile = 1).
compute quantd(1,5) = 1.
compute quantc(1,5) = 0.
compute tmp = m(:,j).
compute tmp(GRADE(m(:,j))) = m(:,j).
compute matm = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (mdich = 1).
compute matm = {cmin(m); cmax(m)}.
end if.
do if ( 999 <> 999).
compute matm = 999.
compute quantd(1,5) = 0.
compute quantc(1,5) = 0.
end if.
compute modmatv(1,5)=nrow(matm).
compute modmat((1:nrow(matm)), 5) = matm.
compute modvnm(1,5)=mnames(1,j).
compute modmatp(1,5) = 1.
end if.
end if.
end loop.
end loop.
do if (minprobe=1).
compute note(notes,1) = 14.
compute notes = notes + 1.

```

```

end if.
do if (maxprobe=1).
compute note(notes,1) = 15.
compute notes = notes + 1.
end if.
do if (cname <> 'xxx' and clsmth = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 23.
end if.
do if (cname <> 'xxx').
do if ((cname = zname) or (cname = vname)
or (cname = wname) or (cname = xname) or
(cname = yname) or (cname = qname)).
compute errs = errs+1.
compute runerrs(errs,1) = 24.
end if.
end if.
do if (wlist = 1 and werr = 0 and wmatch = 0).
compute werr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (zlist = 1 and zerr = 0 and zmatch = 0).
compute zerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (qlist = 1 and qerr = 0 and qmatch = 0).
compute qerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (vlist = 1 and verr = 0 and vmatch = 0).
compute verr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (yerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 16.

```

```

end if.
do if (xerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 32.
end if.
do if (model = 6 and nmeds < 2).
compute errs = errs+1.
compute runerrs(errs,1) = 18.
end if.
do if (mmatch < ncol(mnames)).
compute errs=errs+1.
compute runerrs(errs,1) = 25.
end if.
do if (cname <> 'xxx').
compute cld = design(cld).
compute cluster = ncol(cld).
compute cld = cld(:,2:ncol(cld)).
compute clsdmy = ncol(cld).
do if (clsdmy > 19).
compute errs = errs+1.
compute runerrs(errs,1) = 26.
end if.
do if (mcx > 0 or mcm > 0).
compute errs = errs+1.
compute runerrs(errs,1) = 38.
end if.
end if.
compute dichy = 1.
loop jj = 1 to n.
do if ((y(jj,1) <> cmin(y)) and (y(jj,1) <>
cmax(y))).
compute dichy = 0.
break.
end if.
end loop.
do if ((dichy = 1) and (olsdichy = 1)).
compute dichy=0.
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.

```

```

do if (dichy = 1).
do if (toteff=1).
compute toteff=0.
compute note(notes,1) = 31.
compute notes = notes + 1.
end if.
compute jncrit=xp2*xp2.
do if (ws = 1).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 28.
end if.
end if.
compute ncovs = ncol(dat)-rsum(nvarch).
do if (effsize = 1 and covmy <> 0 and ncovs >
0 and model > 3 and model < 7).
compute note(notes,1) = 22.
compute notes = notes + 1.
compute effsize = 0.
end if.
do if (ws = 1 and ncovs < nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 29.
end if.
do if (ws = 1 and ncovs > nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 30.
end if.
end if.
do if (errs = 0).
do if (rsum(quantd) > 0).
compute note(notes,1) = 4.
compute notes = notes+1.
end if.
do if (rsum(quantc) > 0).
compute note(notes,1) = 5.
compute notes = notes+1.
end if.

```

```

do if (ncovs > 0).
compute c = make(n,ncovs,0).
compute cnames = {'x'}.
compute j = 1.
loop i = 1 to ncol(vnames).
do if (nvarch(1,i)) = 0.
compute c(:,j) = dat(:,i).
compute nvarch(1,i)=1.
compute j=j+1.
compute cnames = {cnames, vnames(:,i)}.
end if.
end loop.
compute cnames = cnames(1,2:ncol(cnames)).
do if (ws = 1).
compute covmean=csum(c)/n.
loop i = 1 to ncovs.
compute meanvec = make(n,1, covmean(1,i)).
compute c(:,i) = c(:,i)-meanvec.
end loop.
compute centvar = {centvar, cnames}.
compute x=c(:,1:ncovs).
do if (ncovs=nmeds).
compute ncovs=0.
end if.
end if.
end if.
compute names = {yname, xname, mnames,
wname, zname, vname, qname}.
do if (ncovs > 0).
compute names = {names, cnames}.
end if.
do if (dichy = 1 and effsize = 1).
compute note(notes,1) = 2.
compute notes = notes+1.
end if.
do if (model > 3 and model < 6).
compute indeff=make(nmeds,1,0).
compute indboot=make(boot+1,nmeds,999).
do if (mc > 0).
compute indboot=make(mc+1,nmeds,999).

```

```

end if.
do if (effsize = 1 and dichy = 0).
compute rmeff=make(boot+1,nmeds+1,999).
compute abpseff=make(boot+1,nmeds+1,999).
compute abcseff=make(boot+1,nmeds+1,999).
compute pmeff=make(boot+1,nmeds+1,999).
compute r245 = make(boot+1,1,999).
compute kappa2 = make(boot+1,1,999).
end if.
end if.
do if (model = 6 and effsize = 1 and dichy = 0).
compute
rmeff=make(boot+1,ncol(indboot),999).
compute
abpseff=make(boot+1,ncol(indboot),999).
compute
abcseff=make(boot+1,ncol(indboot),999).
compute
pmeff=make(boot+1,ncol(indboot),999).
end if.
do if (nmods > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm(1,tmp) = modvnm(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:nmods).
compute modvnm=modvnm(:,1:nmods).
compute modmatv=modmatv(:,1:nmods).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.

```

```

compute modvals =
make((modmatv(1,1)*modprod(1,1)),
nmods,0).
loop i = 1 to nmods.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvals)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvals(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
do if (model = 74).
compute modvals=matx.
compute modvnm=xname.
end if.
compute vmat = make(8,nrow(modvals),0).
compute vmat(1,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute vmat(5,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute indeff=make(nrow(modvals),1,0).
do if (model <> 5).
compute indboot = make(((boot+1)*nmeds),
nrow(modvals), -99999999).
compute indbootp =
make(boot+1,(nmeds*nmods), -99999999).
do if (imm > 4 and imm < 8).
do if (imm = 7).
compute immmv1=matw.
compute immmv2=matv.
end if.
do if (imm = 5).
compute immmv1=matw.
compute immmv2=matz.
end if.

```



```

do if (imm = 6).
compute immmv1=matv.
compute immmv2=matq.
end if.
compute
indboop1=make((boot+1),(nmeds*(nrow(immmv2))),-9999999).
compute
indboop2=make((boot+1),(nmeds*(nrow(immmv1))),-9999999).
compute onesmv1=make(1,nrow(immmv1),1).
compute onesmv2=make(1,nrow(immmv2),1).
compute
nimvs=nrow(immmv1)+nrow(immmv2).
compute cmmval=0.
compute cmmlbs1={'xx'}.
compute cmmlbs2={'xx'}.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv2}.
compute
cmmlbt=make(nrow(immmv2),1,mnames(1,i)).
compute cmmlbs1={cmmlbs1;cmmlbt}.
end loop.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv1}.
compute
cmmlbt=make(nrow(immmv1),1,mnames(1,i)).
compute cmmlbs2={cmmlbs2;cmmlbt}.
end loop.
compute cmmval=cmmval(2:nrow(cmmval),:).
end if.
end if.
end if.
do if (nmods> 0).
loop i = 1 to ncol(modvals).
do if (modvnm(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm(1,i)=zname).
compute zcol = i.

```

```

end if.
do if (modvnm(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm(1,i)=qname).
compute qcol = i.
end if.
end loop.
end if.
do if (dichy = 1).
compute omx = cmax(y).
compute omn = cmin(y).
compute y = (y = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute data = {cons,y,m,x}.
compute datamed = data.
compute datayed = data.
compute datanm = {'constant'; yname;
t(mnames); xname}.
compute datanmm = {'constant'; yname;
t(mnames); xname}.
compute datanmy = {'constant'; yname;
t(mnames); xname}.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
do if (model < 4 and errs = 0).
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' ', ' '}.
compute datayed = {datayed, x&*m}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(1,i) = 1.
compute vmat(2,i) = modvals(i,1).
end loop.
compute mmat = make(16,nmeds,1).
end if.
do if (model = 2 or model = 3).
compute int1 = x&*w.

```

```

compute datayed = {datayed, w, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,2).
compute vmat(3,i) = modvals(i,1).
compute vmat(4,i) = modvals(i,1)*modvals(i,2).
end loop.
end if.
do if (model = 3).
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*m.
compute int2 = x&*w&*m.
compute datayed = {datayed, int1, int2}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' X', wname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (model = 4 or model = 5).
compute vmat = make(8,1,1).
end if.
compute yintkey2 = yintkey.
do if (wm = 1).
compute int1 = x&*w.
compute datamed = {datamed, w,int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmm = {datanmm; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).

```

```

compute vmat(2,i) = modvals(i,wcol).
end loop.
do if (zm = 1).
compute int1 = x*z.
compute datamed = {datamed, z, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(3,i) = modvals(i,zcol).
end loop.
end if.
do if (wzm = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w*z.
compute int2 = x*w*z.
compute datamed = {datamed, int1, int2}.
loop i = 1 to nrow(modvals).
compute vmat(4,i) =
modvals(i,wcol)*modvals(i,zcol).
end loop.
end if.
end if.
compute mdatacol=ncol(datamed).
compute mintkey = yintkey.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
compute medints = intcnt-1.
do if (vy = 1 or xmy = 1).
compute mp = 1.

```

```

do if (xmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,1).
end loop.
end if.
do if (vy = 1).
compute datayed = {datayed, v}.
compute datanmy = {datanmy; vname}.
compute mmods = 1.
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,vcol).
end loop.
do if (qy = 1).
compute mp = 2.
compute datayed = {datayed,q}.
compute datanmy = {datanmy; qname}.
compute mmods = 2.
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,qcol).
end loop.
end if.
do if (vqy = 1).
compute mp = 3.
compute datayed = {datayed, v*&q}.
compute mmods = 3.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,vcol)*modvals(i,qcol).
end loop.
end if.
end if.
compute mints = make(n,(nmeds*mp),0).
loop i = 0 to (nmeds-1).
do if (i = 0 and vqy = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
vname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.

```

```

do if (vy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (xmy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))*x.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', xname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (qy = 1).
compute mints(:,((i*mp)+2))=m(:,(i+1))*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqy = 1).
compute mints(:,((i*mp)+3))=m(:,(i+1))*v*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end loop.
compute datayed = {datayed, mints}.
end if.
compute mp = 1.
do if (wvmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).

```

```

compute vmat(8,i) =
modvals(i,wcol)*modvals(i,vcol).
end loop.
end if.
compute mints2 = make(n,(nmeds*mp),0).
do if (wmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,wcol).
end loop.
do if (wy = 0 and model > 3).
compute datayed = {datayed, w}.
compute datanmy = {datanmy; wname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wvmy = 1).
compute datayed = {datayed,w&*v}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints2(:,((i*mp)+1))= m(:,(i+1))&*w.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wvmy = 1).
compute mints2(:,((i*mp)+2))=
m(:,(i+1))&*w&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints2}.
end if.

```

```

compute mp = 1.
do if (zmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,zcol).
end loop.
do if (wzmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,zcol)&*modvals(i,wcol).
end loop.
end if.
end if.
do if (zmy = 1).
compute mints3 = make(n,(nmeds*mp),0).
do if (zy = 0).
compute datayed = {datayed, z}.
compute datanmy = {datanmy; zname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wzmy = 1 and wzy = 0).
compute datayed = {datayed,w&*z}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints3(:,((i*mp)+1))= m(:,(i+1))&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wzmy = 1).
compute mints3(:,((i*mp)+2))=
m(:,(i+1))&*w&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', zname}.

```



```

compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints3}.
end if.
compute decoc = 1.
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
do if (wy = 1 and model > 3).
compute datayed = {datayed, w, x&*w}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm2(1,1)=wname.
compute modmatp(1,1) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (zy = 1).
compute datayed = {datayed,z,x&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm2(1,2)=zname.
compute modmatp(1,2) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.

```

```

do if (wzy = 1).
compute datayed = {datayed,w&*z,x&*w&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vxy = 1).
compute datayed = {datayed, x&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm2(1,3)=vname.
compute modmatp(1,3) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (qxy = 1).
compute datayed = {datayed, x&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm2(1,4)=qname.
compute modmatp(1,4) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.

```

```

compute intcnt = intcnt+1.
do if (vqxy = 1).
compute datayed = {datayed, x&*v&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, 'X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end if.
do if (wvxy = 1).
compute datayed = {datayed, x&*w&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, 'X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute modvalsd = 0.
compute ttt = rsum(modmatp).
do if (rsum(modmatp) > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm2(1,tmp) = modvnm2(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:ttt).
compute modvnm2=modvnm2(:,1:ttt).
compute modmatv=modmatv(:,1:ttt).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.

```

```

loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvalsd =
make((modmatv(1,1)*modprod(1,1)), ttt,0).
loop i = 1 to ttt.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvalsd)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvalsd(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
end if.
do if (ttt > 0).
loop i = 1 to ncol(modvalsd).
do if (modvnm2(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm2(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm2(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm2(1,i)=qname).
compute qcol = i.
end if.
end loop.
compute directv = make(nrow(modvalsd),1,1).
do if (wy = 1).
compute directv ={directv, modvalsd(:,wcol)}.
end if.

```

```

do if (zy = 1).
compute directv = {directv, modvalsd(:,zcol)}.
end if.
do if (wzy = 1).
compute directv = {directv,
(modvalsd(:,wcol)*modvalsd(:,zcol))}.
end if.
do if (vxy = 1).
compute directv = {directv, modvalsd(:,vcol)}.
end if.
do if (qxy = 1).
compute directv = {directv, modvalsd(:,qcol)}.
end if.
do if (vqxy = 1).
compute directv = {directv,
(modvalsd(:,vcol)*modvalsd(:,qcol))}.
end if.
do if (wvxy = 1).
compute directv = {directv,
(modvalsd(:,vcol)*modvalsd(:,wcol))}.
end if.
end if.
compute ydatacol = ncol(datayed).
do if (ncovs > 0).
do if (covmy <> 2).
compute datamed = {datamed,c}.
end if.
do if (covmy <> 1).
compute datayed = {datayed,c}.
end if.
do if (covmy = 1 and model = 4 and mcx > 0).
compute datayed = {datayed,c}.
end if.
compute covmeans = csum(c)/n.
end if.
do if (cluster > 0).
compute datamed = {datamed,cld}.
compute datayed = {datayed, cld}.
compute cldmeans = csum(cld)/n.
end if.

```

```

compute mst = 3.
compute mnd = mst+nmeds-1.
compute ydatacol=ncol(datayed).
compute mdatacol=ncol(datamed).
do if (ncovs > 0).
compute datanmy = {datanmy; t(cnames)}.
do if (model > 3).
compute datanmm = {datanmm; t(cnames)}.
end if.
end if.
compute datanmy = {'constant';
datanmy(3:nrow(datanmy),1)}.
do if (model > 3).
compute datanmm = {'constant';
datanmm(3:nrow(datanmm),1)}.
end if.
compute amm = make(2,1,0).
compute abmm = make(2,1,0).
compute mnv = csum(datayed(:,2)/n).
compute mnv = make(n,1,mnv).
compute ssty = csum((datayed(:,2)-mnv)**2).
compute sigma = (n*sscp(datayed)-
(t(csum(datayed))*(csum(datayed)))).
compute sigma = sigma/(n*(n-1)).
do if (ws=1).
compute stddevm=diag(sigma).
compute
stddevm=sqrt(stddevm(3:(nmeds+2),1)).
end if.
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute r2xy =
(sigma(2,(3+nmeds))/(stddevy*stddevx))**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))**2.
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).

```

```

do if (model = 4 and nmeds = 1 and cluster = 0
and ncovs = 0).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
compute datatm = datamed.
compute dataty = datayed.
compute mdlrms2 = { '4' ; yname; xname}.
compute mdlrms = {'Model ='; ' Y ='; ' X ='
}.
do if (ws = 1).
compute mdlrms2 = { '4' ; yname}.
compute mdlrms = {'Model ='; ' Y ='}.
end if.
loop i = 1 to ncol(mnames).
compute mdlrms2 = {mdlrms2; mnames(1,i)}.
do if (i = 1 and ncol(mnames) = 1).
compute mdlrms = {mdlrms; ' M ='}.
else.

```

```

compute mdlInms = {mdlInms; mlab(i,1)}.
end if.
end loop.
do if (wname <> 'xxx').
compute mdlInms2 = {mdlInms2; wname}.
compute mdlInms = {mdlInms; ' W = '}.
end if.
do if (zname <> 'xxx').
compute mdlInms2 = {mdlInms2; zname}.
compute mdlInms = {mdlInms; ' Z = '}.
end if.
do if (vname <> 'xxx').
compute mdlInms2 = {mdlInms2; vname}.
compute mdlInms = {mdlInms; ' V = '}.
end if.
do if (qname <> 'xxx').
compute mdlInms2 = {mdlInms2; qname}.
compute mdlInms = {mdlInms; ' Q = '}.
end if.
do if (jn = 1 and model = 1 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
do if (jn = 1 and model = 3 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
compute mdpbe=0.
compute medte2=0.
do if ( 4 =4 and mcx > 0).
compute medte2=1.
end if.
do if (((mcx > 0) or (mcm > 0)) AND model =
1).
compute mdpbe=1.
compute dd={datayed(:,4),datayed(:,3)}.
do if (ncovs > 0).
compute dd={datayed(:,6:(5+ncovs)),dd}.
end if.
compute dd={datayed(:,2),dd}.

```



```

end if.
compute yes = ((nrow(modvals)=1) and
(contrast = 1) and (model = 3 or model = 2)).
compute yes2 = ((nrow(modvals)=1) and
(contrast = 1) and (model = 2)).
print/title = '***** PROCESS Procedure
for SPSS Release 2.16.3 *****'.
print/title = '      Written by Andrew F. Hayes,
Ph.D.    www.afhayes.com'.
print/title = '  Documentation available in
Hayes (2013).
www.guilford.com/p/hayes3'/space=0.
print mdlnms2/title =
*****
*****'/rnames = mdlnms/format =
a8.
do if (ncovs > 0).
print cnames/title = 'Statistical Controls: '/rlabels
= 'CONTROL='/format a8.
end if.
print n/title = 'Sample size'/format F10.0.
do if (cluster > 0).
print cluster/rnames = cvname/title =
'Clustering Variable and Number of Clusters'.
end if.
do if (model > 3 and ( 'random' <> 'random')).
compute seedt= 'random'.
print seedt/title='Custom'/format=A12/rlabel =
'Seed:'.
end if.
do if (mdpbe=0 and medte2 = 0).
loop bt = 1 to (boot+1).
do if (bt = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef = 0.
do if (bt > 1).
loop.
compute v=trunc(uniform(n,1)*n)+1.

```

```

compute datayed = dataty(v,:).
do if (ws=1).
compute
bmeans=csum(datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed)))/n.
compute cntmat=make(n,nmeds,1).
compute bmeans2=bmeans.
do if (nmeds > 1).
compute bmeans2=mdia(bmeans).
end if.
compute cntmat=cntmat*bmeans2.
compute datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed))=datayed(:,(ncol(data
yed)-nmeds+1):ncol(datayed))-cntmat.
end if.
compute detcheck=det(t(datayed)*datayed).
compute rk=(detcheck <> 0).
do if (model > 3).
compute datamed = datatm(v,:).
compute detcheck=det(t(datamed)*datamed).
do if (rk=1).
compute rk=(detcheck <> 0).
end if.
end if.
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
compute temp=diag(sigma).
compute bad = bad+(1-rk).
compute false = 1.
end loop if (rk = 1).
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nme
ds)).
do if (model = 4 and nmeds = 1 and ncovs = 0
and cluster = 0).

```

```

compute r2xy =
(sigma(2,4)/(stddevy*stddevx))&**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))&**2.
compute sstot = sigma(2,2)*(n-1).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
end if.
do if (model > 3).
loop im = 1 to nmeds.
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant';
datanmm((2+nmeds):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
do if (model = 6).

```

```

do if (im = 1).
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
end if.
do if (im > 1).
compute xm={cons, datamed(:,3:(im+1)),
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant'; datanmm(2:im,1);
datanmm((mnd):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
compute
mmpaths((im+1),(2:im))=t(coeff(2:im,1)).
end if.
end if.
do if (ws = 1).
compute
coeff(1,1)=csum(datamed(:,(2+im)))/n.
end if.
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
do if (bt = 1).
compute resid=datamed(:,(2+im))-xm*coeff.
compute sse =cssq(resid).
compute mse = sse/(n-ncol(xm)).
compute mnv = csum(data(:,(2+im)))/n.
compute mnv = make(n,1,mnv).
compute sstm = csum((data(:,(2+im))-
mnv)**2).
compute k3 = nrow(coeff).
do if (hc3 = 1).
compute h = xm(:,1).
loop i3=1 to n.
compute h(i3,1)= xm(i3,:)*invXtX*t(xm(i3,:)).
end loop.

```

```

loop i3=1 to k3.
compute xm(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xm(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xm(:,i3) = sqrt(mse)&*xm(:,i3).
end loop.
end if.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute hccov=invXtX*t(xm)*xm*invXtX.
compute mcmats(im,im)=hccov(2,2).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*hccov*lmat)*((t(lmat)
)*coeff))/dfnum.
compute coeff = coeff(1:(nrow(coeff)-
clsdmy),1).
compute mccoeff(im,1)=coeff(2,1).
compute standerr =
sqrt(diag(invXtX*t(xm)*xm*invXtX)).
compute standerr = standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1).
compute standerr(1,1)=stddevm(im,1)/sqrt(n).
end if.
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-
ncol(xm)))).
compute temp=(n-ncol(xm)).
do if (ws=1).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute temp=(n-1).
end if.
compute xd = abs(xp2).

```

```

compute temp = (temp* (exp((temp-
(5/6))*((xd/(temp-(2/3)+(.11/temp))))*(xd/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = coeff-
sqrt(abs(temp))*standerr.
compute temp2 =
coeff+sqrt(abs(temp))*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (ws=1).
compute op=op(1,:).
compute xnmn =
apathnam(((nmeds>1)+1+(im-1)),1).
end if.
compute sobel(im,1) = coeff(2,1).
compute sobel(im,2)=standerr(2,1).
compute temp = mnames(1,im).
compute r2full = 1-(sse/sstm).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute summ = {sqrt(r2full), r2full, mse,
Fratio, dfnum, dfden, pfr}.
do if (detail = 1).
print temp/title =
*****
*****'/rlabels = 'Outcome:'/format
= A8.
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels 'R',
'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
print op/title = 'Model'/rnames = xnmn/clabels
'coeff', 'se', 't', 'p', 'LLCI', 'ULCI'/format = F10.4.
do if (covcoeff=1 and ws <> 1).
compute
hccovtmp=hccov(1:nrow(op),1:nrow(op)).
compute cnamestp=t(xnmn).

```

```

print hccovtmp/title='Covariance matrix of
regression parameter
estimates'/rnames=xmnm/cnames=cnamestp/f
ormat= F10.4.
end if.
do if (nmods > 0 and nrow(mintkey) > 1).
print mintkey/title = 'Product terms key:.'/format
= A8.
end if.
do if (ws = 1).
compute wsdf=n-1.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
end if.
end if.
compute ymat(1,im) = coeff(2,1).
do if (ws = 1).
compute ymat(1,im)=coeff(1,1).
end if.
do if (wm = 1).
compute ymat(2,im) = coeff(4,1).
do if (zm = 1).
compute ymat(3,im) = coeff(6,1).
do if (wzm = 1).
compute ymat(4, im) = coeff(8,1).
end if.
end if.
end if.
do if (model = 6).
compute mmpaths((im+1),1)=coeff((im+1),1).
end if.
end loop.
end if.
loop totlp = 1 to (1+(toteff*(bt = 1))).
do if (toteff = 1 and totlp =2).
compute xy =
{cons,datayed(:,(3+nmeds):ydatacol)}.
else.
compute xy={cons, datayed(:,3:ydatacol)}.
end if.

```

```

do if (dichy = 1).
compute pt2 =
make(nrow(datayed(:,2)),1,(csum(datayed(:,2))
/n)).
compute LL3 = datayed(:,2)*ln(pt2)+(1-
datayed(:,2))*ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xy),1,0).
compute LL1 = 0.
compute xy22=xy.
loop jjj = 1 to iterate.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.
compute coeff =
bt1+inv(t(xy22)*xy)*t(xy)*(datayed(:,2)-pt1).
compute pt1 = 1/(1+exp(-(xy*coeff))).
compute itprob = csum((pt1 <
.00000000000001) or (pt1 >
.99999999999999)).
do if (itprob = 0).
compute LL = datayed(:,2)*ln(pt1)+(1-
datayed(:,2))*ln(1-pt1).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
break.
end if.
compute bt1 = coeff.
compute LL1 = LL2.
end loop.
do if (jjj >= iterate and iterr = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
computer iterr = 1.
end if.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.

```



```

compute covmat = inv(t(xy22)*xy).
release xy22.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
end if.
do if (dichy = 0).
compute invXtX = inv(t(xy)*xy).
compute coeff =invXtX*t(xy)*datayed(:,2).
do if (effsize=1 and model > 3 and model < 7
and (ncovs > 0 or clsdmy > 0)).
compute xysdy =
{cons,datayed(:,(4+nmeds):ydatacol)}.
compute coeffsd
=inv(t(xysdy)*xysdy)*t(xysdy)*datayed(:,2).
compute resid=datayed(:,2)-(xysdy*coeffsd).
compute sdy cov =sqrt(csum(resid&*resid)/(n-
ncol(xysdy))).
compute xvaron=datayed(:,(3+nmeds)).
compute
coeffx2=inv(t(xysdy)*xysdy)*t(xysdy)*xvaron.
compute residx2=xvaron-(xysdy*coeffx2).
compute sdx cov
=sqrt(csum(residx2&*residx2)/(n-ncol(xysdy))).
end if.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4 and bt > 1).
compute resid=datayed(:,2)-xy*coeff.
compute sse = cssq(resid).
compute r2full = 1-(sse/sstot).
end if.
do if (bt =1).
compute resid=data(:,2)-xy*coeff.

```

```

compute k3 = nrow(coeff).
compute sse = cssq(resid).
compute mse = sse/(n-ncol(xy)).
compute
hcnor=coeff&/(sqrt(diag(mse*invXtX))).
do if (hc3 = 1).
compute h = xy(:,1).
loop i3=1 to n.
compute h(i3,1)= xy(i3,:)*invXtX*t(xy(i3,:)).
end loop.
loop i3=1 to k3.
compute xy(:,i3) = (resid(:,ncol(resid))&/(1-
h))*xy(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xy(:,i3) = sqrt(mse)*xy(:,i3).
end loop.
end if.
compute covmat = (invXtX*t(xy)*xy*invXtX).
end if.
end if.
do if (bt = 1).
do if (model = 2).
compute xy2={cons, datayed(:,3:ydatacol)}.
compute temp = ncol(xy2).
do if (temp > 6).
compute xy3=xy2(:,7:temp)}.
end if.
compute xy2={xy2(:,1:3), xy2(:,5)}.
do if (temp > 6).
compute xy2={xy2, xy3}.
release xy3.
end if.
compute invXtX = inv(t(xy2)*xy2).
compute coeff2 =invXtX*t(xy2)*datayed(:,2).
compute ssem2=cssq(datayed(:,2)-xy2*coeff2).
release xy2.
end if.

```

```

compute stderr = sqrt(diag(covmat)).
do if (totlp = 1).
compute
mcmats((nmeds+1):ncol(mcmats),(nmeds+1):n
col(mcmats))=covmat(2:(1+nmeds),2:(1+nmed
s)).
end if.
compute stderr=stderr(1:(nrow(stderr)-
clsdmy),1).
do if (ws=1 and totp=2).
compute stderr(1,1)=stddevy/sqrt(n).
end if.
compute coeffplt = coeff.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*coeff)))/dfnum).
compute coeff=coeff(1:(nrow(coeff)-clsdmy),1).
do if (totlp=1).
compute
mccoeff((nmeds+1):nrow(mccoeff))=coeff(2:(1+
nmeds),1).
end if.
compute bbbb=coeff(2,1).
do if totp = 1.
compute deco(1,1)=2+nmeds.
compute deco = deco(1:decoc,1).
compute covdirt = make((nrow(covmat)-
clsdmy),(ncol(covmat)-clsdmy),0).
compute covdirt = covmat(deco,:).
compute covdir =
make(nrow(covdirt),ncol(covdirt),0).
compute covdir = covdirt(:,t(deco)).
compute deco=coeff(deco,1).
do if (ttt > 0).
compute sedir =
sqrt(diag(directv*covdir*t(directv))).

```

```

compute directv = directv*deco.
end if.
compute sobel(:,3)=coeff(2:(1+nmeds),1).
compute sobel(:,4) = standerr(2:(1+nmeds),1).
compute sobel2 = sobel&*sobel.
do if (varorder <> 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)&*sobel2(:,4)+sobel2(:,3)&*sob
el2(:,2)).
end if.
do if (varorder = 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)&*sobel2(:,4)+sobel2(:,3)&*sob
el2(:,2)+sobel2(:,2)&*sobel2(:,4)).
end if.
compute sobel(:,1) = sobel(:,1)&*sobel(:,3).
compute sobel(:,3) = sobel(:,1)&/sobel(:,2).
compute sobel(:,4) = 2*(1-
cdfnorm(abs(sobel(:,3)))).
end if.
do if (dichy = 0).
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
do if (ws= 1 and totlp=2).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute dfden=n-1.
end if.
compute cnms = {'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute op = {coeff, standerr, tratio, p}.
end if.
do if (dichy = 1).
compute tratio = (coeff&/standerr).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute wald = tratio&*tratio.
compute cnms = {'coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute temp=coeff-abs(xp2)*standerr.
compute op = {coeff, standerr, tratio, p, temp}.
compute temp=coeff+abs(xp2)*standerr.

```

```

compute op = {op, temp}.
end if.
do if (detail = 1).
do if (totlp = 2).
print yname/title = '*****'
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:.'/format = A8.
end if.
do if (totlp <> 2).
print yname/title =
*****
*****'/rlabels = 'Outcome:.'/format
= A8.
end if.
end if.
do if (dichy = 1 and bt = 1 and totp = 1).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:.'/cnames = nmsd/format = F9.2.
end if.
do if (dichy = 0).
compute r2full = 1-(sse/ssty).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute jndf=dfden.
compute wsdf=dfden.
compute xd = abs(xp2).
compute jncrit = (dfden* (exp((dfden-
(5/6))*((xd/(dfden-
(2/3)+(.11/dfden)))*(xd/(dfden-
(2/3)+(.11/dfden)))))-1)).
compute summ = {sqrt(r2full), r2full, mse,
fratio, dfnum, dfden, pfr}.
compute temp1=coeff-sqrt(jncrit)*standerr.
compute temp2=coeff+sqrt(jncrit)*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (detail = 1).
do if (ws <> 1).

```

```

print summ/title = 'Model Summary'/clabels =
'R', 'R-sq', 'MSE' 'F', 'df1', 'df2', 'p'/format =
F10.4.
end if.
end if.
end if.
do if (dichy = 1).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(coeff)-
1)).
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
do if (detail = 1).
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format =
F10.4.
end if.
end if.
do if (totlp = 2).
compute datanmy={'constant';
datanmy((nmeds+2):nrow(datanmy),1)}.
end if.
do if (detail = 1).
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
do if (ws=1).
compute op2=op(1:(1+nmeds),:).
compute datanmy2 = {'c" path'; bpathnam(
(1+(nmeds>1)): ((1+(nmeds>1))+(nmeds-
1)),1)}.
do if (totlp=2).
compute datanmy2={'c path'}.
compute op2=op(1,:).
compute wsdf=n-1.
end if.

```

```

print op2/title = 'Model'/rnames =
datanmy2/cnames = cnms/format = F10.4.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
do if (ws <> 1).
print op/title = 'Model'/rnames =
datanmy/cnames = cnms/format = F10.4.
do if (covcoeff=1).
compute
covmattp=covmat(1:nrow(op),1:nrow(op)).
compute cnamestp=t(datanmy).
print covmattp/title='Covariance matrix of
regression parameter
estimates'/rnames=datanmy/cnames=cnamest
p/format= F10.4.
end if.
end if.
end if.
do if (ttt = 0 and totlp = 1).
compute deco = op((nmeds+2),:).
do if (ws = 1).
compute deco=op(1,:).
end if.
end if.
do if (ttt = 0 and totlp = 2).
compute decotot = op(2,:).
do if (ws = 1).
compute decotot=op(1,:).
end if.
end if.
do if (nmods > 0 and model > 4 and detail = 1
and nrow(yintkey) > 1)).
print yintkey/title = 'Product terms key:'/format
= A8.
end if.
do if (nmods > 0 and model < 4 and detail = 1).
print yintkey2/title = 'Product terms key:'/format
= A8.
do if ((model = 1 or model = 2) and dichy = 0).

```

```

compute temp={((op(4,3)**2)*(1-r2full))/dfden,
op(4,3)**2, 1, dfden, op(4,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(4,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(2,1).
do if (model = 2).
compute temp={temp;((op(6,3)**2)*(1-
r2full))/dfden, op(6,3)**2, 1, dfden, op(6,4)}.
do if (hc3=1).
compute temp(2,1)=((hcnot(6,1)**2)*(1-
r2full))/dfden.
end if.
compute frat2=(dfden*(r2full-(1-
(ssem2/ssty))))/(2*(1-r2full)).
do if (hc3=1).
compute lmat=make(nrow(coeffplt),2,0).
compute lmat(4,1)=1.
compute lmat(6,2)=1.
compute frat2 =
(t(t(lmat)*coeffplt)*inv(t(lmat)*covmat*lmat)*((t(l
mat)*coeffplt)))/2).
end if.
compute temp={temp;(r2full-(1-
(ssem2/ssty))),frat2,2,dfden,1-
fcdf(frat2,2,dfden)}.
compute rnms={rnms; yintkey2(3,1),'Both'}.
end if.
print temp/title = 'R-square increase due to
interaction(s):'/rnames=rnms/clabels = 'R2-
chng', 'F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (model = 3 and dichy = 0).
compute temp={((op(8,3)**2)*(1-r2full))/dfden,
op(8,3)**2, dfden, op(8,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(8,1)**2)*(1-
r2full))/dfden.
end if.

```



```

compute rnms=yintkey2(5,1).
print temp/title = 'R-square increase due to
three-way interaction:.'/rnames=rnms/clabels =
'R2-chng', 'F(1,df2)', 'df2', 'p'/format = F10.4.
end if.
end if.
end if.
do if (model = 6 and totlp = 1).
compute
mmpaths(nrow(mmpaths),1)=coeff(nrow(mmpa
ths),1).
compute
mmpaths(nrow(mmpaths),(2:(nmeds+1)))=t(co
eff(2:(nmeds+1),1)).
end if.
do if (totlp = 1).
loop im = 1 to nmeds.
do if (model < 4).
compute ymat(1,im) = coeff(3,1)*(1-yes).
compute ymat(2,im) = coeff(4,1).
compute cmat(1,im) = covmat(3,3).
compute cmat(2,im) = covmat(4,4).
compute cmat(5,im) = covmat(3,4).
compute jnb1=coeff(3,1).
compute jnb3=coeff(4,1).
compute jnsb1=covmat(3,3).
compute jnsb3=covmat(4,4).
compute jnsb1b3=covmat(3,4).
do if (model = 2 or model = 3).
compute ymat(3,im) = coeff(6,1).
compute cmat(3,im) = covmat(6,6).
compute cmat(6,im) = covmat(3,6).
compute cmat(8,im) = covmat(4,6).
end if.
do if (model = 3).
compute ymat(4,im) = coeff(8,1).
compute cmat(4,im) = covmat(8,8).
compute cmat(7,im) = covmat(3,8).
compute cmat(9,im) = covmat(4,8).
compute cmat(10,im) = covmat(6,8).

```

```

compute jnb1=coeff(4,1).
compute jnb3=coeff(8,1).
compute jnsb1=covmat(4,4).
compute jnsb3=covmat(8,8).
compute jnsb1b3=covmat(4,8).
end if.
end if.
do if (model > 3).
compute ymat(5,im) = coeff((1+im),1).
end if.
do if (xmy = 1).
compute ymat(6,im) = coeff((2+nmeds+im),1).
end if.
do if (vy = 1).
compute ymat(6,im) = coeff((3+nmeds+im),1).
end if.
do if (qy = 1 and vy = 1).
compute ymat(6,im) = coeff((5+nmeds+((im-1)*2)),1).
compute ymat(7,im) = coeff((6+nmeds+((im-1)*2)),1).
end if.
do if (vqy = 1).
compute ymat(6,im) = coeff((6+nmeds+((im-1)*3)),1).
compute ymat(7,im) = coeff((7+nmeds+((im-1)*3)),1).
compute ymat(8,im) = coeff((8+nmeds+((im-1)*3)),1).
end if.
do if (wmy = 1).
compute ymat(7,im) = coeff((3+nmeds+im-wy),1).
end if.
do if (wmy = 1 and vy = 1).
compute ymat(7,im) =coeff((4+(nmeds*2)+im-wy),1).
end if.
do if (wmy = 1 and vy = 1 and wvmy = 1).

```

```

compute ymat(7,im) =
coeff((6+(nmeds*2)+((im-1)*2)-wy),1).
compute ymat(8,im) =
coeff((7+(nmeds*2)+((im-1)*2)-wy),1).
end if.
do if (wmy = 1 and zmy = 1).
compute ymat(6,im) = coeff((6-(wzy*3)+(wzm-
1)+(nmeds*2)+((im-1)*2)*wzm)+((im-1)*(1-
wzm))-((zy-wzy)*2),1).
compute ymat(7,im) = coeff((3-
zy+im+nmeds),1).
do if (wzmy = 1).
compute ymat(8,im) = coeff((7-
(wzy*3)+(nmeds*2)+((im-1)*2)),1).
end if.
end if.
do if (nmods > 0 and model <> 5).
loop indlp = 1 to nrow(modvals).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp)).
do if (model > 6).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp))*csum(ym
at(5:8,im)*vmat(5:8,indlp)).
end if.
end loop.
compute indboot((bt+(im-
1)*(boot+1)),:)=t(indeff).
do if (imm = 1 or imm = 3).
compute
indbootp(bt,im)=ymat(2,im)*ymat(5,im).
do if (imm=3).
compute
indbootp(bt,(im+nmeds))=ymat(3,im)*ymat(5,i
m).
do if (zdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxz-cminz).
end if.

```

```

end if.
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
end if.
end if.
do if ((imm= 2 or imm = 4) or model = 74).
compute
indbootp(bt,im)=ymat(1,im)*ymat(6,im).
do if (imm=4).
compute
indbootp(bt,(im+nmeds))=ymat(1,im)*ymat(7,i
m).
do if (qdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxq-cminq).
end if.
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
end if.
end if.
do if (imm=7).
compute
indbootp(bt,im)=ymat(2,im)*ymat(6,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(2,im)*y
mat(6,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(6,im)*onesmv1)+(ymat(2,im)*y
mat(6,im)*t(immmv1)).
do if (wdich=1).

```

```

compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxv-cminv).
end if.
end if.
do if (imm=5).
compute
indbootp(bt,im)=ymat(4,im)*ymat(5,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(4,im)*y
mat(5,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(3,im)*ymat(5,im)*onesmv1)+(ymat(4,im)*y
mat(5,im)*t(immmv1)).
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).

```

```

end if.
do if (zdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxz-
cminz).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxz-cminz).
end if.
end if.
do if (imm=6).
compute
indbootp(bt,im)=ymat(1,im)*ymat(8,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(1,im)*ymat(6,im)*onesmv2)+(ymat(1,im)*y
mat(8,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(7,im)*onesmv1)+(ymat(1,im)*y
mat(8,im)*t(immmv1)).
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxv-cminv).
end if.
do if (qdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxq-
cminq).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-

```

```

1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxq-cminq).
end if.
end if.
do if ((model = 58 or model = 59) and
wvdich=1).
compute
indbootp(bt,im)=(ymat(1,im)*ymat(7,im)*(cmax
w-cminw))+(ymat(2,im)*ymat(5,im)*(cmaxw-
cminw))+(ymat(2,im)*ymat(7,im)*((cmaxw*cma
xw)-(cminw*cminw))).
end if.
end if.
do if (model = 4 or model = 5).
compute
indboot(bt,im)=csum(ymat(1:4,im)*vmat(1:4,1
))*csum(ymat(5:8,im)*vmat(5:8,1)).
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.000000000000001.
end if.
compute sumind=rsum(indboot(bt,:)).
do if (im=nmeds).
compute
pmeff(bt,2:(im+1))=indboot(bt,:)/(sumind+coeff(
(2+nmeds),1)).
end if.
compute
rmeff(bt,(im+1))=indboot(bt,im)/coeff((2+nmeds
),1).
compute
abpseff(bt,(im+1))=indboot(bt,im)/stddevy.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*stddevx.
do if (ncovs > 0 or clsdmy > 0).
compute
abpseff(bt,(im+1))=indboot(bt,im)/sdy cov.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*sdx cov.
end if.

```

```

do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,1) = r2my-(r2full-r2xy).
compute abmmr = 1.
do if (indboot(bt,im) < 0).
compute abmmr = cmin(abmm).
end if.
do if (indboot(bt,im) > 0).
compute abmmr = cmax(abmm).
end if.
compute kappa2(bt,1)=indboot(bt,im)/abmmr.
compute tmp = indboot(bt,im)/abmmr.
end if.
end if.
end if.
do if (model = 6).
do if (nmeds = 2).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(4,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3).
compute
indboot(bt,3)=mmpaths(3,1)*mmpaths(4,3).
end if.
do if (nmeds = 3).
compute indboot(bt,1) =
mmpaths(2,1)*mmpaths(5,2).
compute indboot(bt,2) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3).
compute indboot(bt,3) =
mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4).
compute indboot(bt,4) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4).
compute indboot(bt,5) =
mmpaths(3,1)*mmpaths(5,3).
compute indboot(bt,6) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4).

```



```

compute indboot(bt,7) =
mmpaths(4,1)*mmpaths(5,4).
end if.
do if (nmeds = 4).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(6,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(6,3).
compute
indboot(bt,3)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(6,4).
compute
indboot(bt,4)=mmpaths(2,1)*mmpaths(5,2)*mmpaths(6,5).
compute
indboot(bt,5)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(6,4).
compute
indboot(bt,6)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3)*mmpaths(6,5).
compute
indboot(bt,7)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,8)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,9)=mmpaths(3,1)*mmpaths(6,3).
compute
indboot(bt,10)=mmpaths(3,1)*mmpaths(4,3)*mmpaths(6,4).
compute indboot(bt,11)
=mmpaths(3,1)*mmpaths(5,3)*mmpaths(6,5).
compute indboot(bt,12) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,13) =
mmpaths(4,1)*mmpaths(6,4).

```

```

compute indboot(bt,14) =
mmpaths(4,1)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,15) =
mmpaths(5,1)*mmpaths(6,5).
end if.
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.00000000000001.
end if.
compute
pmeff(bt,:)=indboot(bt,:)/(rsum(indboot(bt,:))+m
mpaths(nrow(mmpaths),1)).
compute rmeff(bt,:) =
indboot(bt,:)/mmpaths(nrow(mmpaths),1).
compute abpseff(bt,:)=indboot(bt,:)/stddevy.
compute
abcseff(bt,:)=(stddevx*indboot(bt,:))/stddevy.
do if (ncovs > 0 or clsdmy > 0).
compute abpseff(bt,:)=indboot(bt,:)/sdy cov.
compute
abcseff(bt,:)=(sdx cov*indboot(bt,:))/sdy cov.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,:) = r2my-(r2full-r2xy).
end if.
end if.
end if.
end loop.
end if.
end loop.
do if (savboot = 1 and bt > 1).
compute bootstrp((bt-1),:)=bootcoef.
end if.
end loop.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.

```

```

release datayed, dat, datamed, data, y, x,
datatm, dataty, mints2, cons, mnv, tmp.
do if (mc > 0).
compute x1 = sqrt(-
2*ln(uniform(mc,nrow(mcmats))))&*cos((2*3.14
159265358979)*uniform(mc,nrow(mcmats))).
compute x1=x1*chol(mcmats).
loop i=1 to nrow(x1).
compute x1(i,:)=x1(i,:)+t(mccoeff).
end loop.
loop i = 1 to nmeds.
compute
indboot(2:nrow(indboot),i)=x1(:,i)&*x1(:,(i+nme
ds)).
end loop.
end if.
do if (ttt = 0 and model > 3).
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
end if.
do if (model <> 74).
do if (dichy = 0).
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 't', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 't', 'p', 'LLCI', 'ULCI'/format =
F10.4.
else.
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 'Z', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.

```

```

print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 'Z', 'p', 'LLCI', 'ULCI' /format =
F10.4.
end if.
end if.
end if.
do if (ttt > 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
compute clbs = {modvnm2, 'Effect', 'SE', 't', 'p',
'LLCI', 'ULCI'}.
compute tratio = (directv&/sedir).
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
compute outp = {modvalsd, directv, sedir,
tratio, p}.
do if (dichy = 0).
compute temp1=directv-sqrt(jncrit)*sedir.
compute temp2=directv+sqrt(jncrit)*sedir.
compute outp = {outp, temp1, temp2}.
end if.
do if (dichy = 1).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute temp = directv-abs(xp2)*sedir.
compute outp = {outp, temp}.
compute temp = directv+abs(xp2)*sedir.
compute outp = {outp, temp}.
compute clbs = {modvnm2, 'Effect', 'SE', 'Z', 'p',
'LLCI', 'ULCI'}.
end if.
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
print outp/title = 'Conditional direct effect(s) of
X on Y at values of the moderator(s):'/format =
F10.4 /cnames = clbs.
end if.
do if (nmods > 0 and model <> 5).
do if (model < 4).

```

```

print/title =
*****
*****!

compute zmat(1,1) = 1*(1-yes).
compute cfse = make(nrow(modvals),1,0).
loop #m = 1 to nrow(modvals).
do if (model = 1).
compute zmat(2,1)=modvals(#m,1)**2.
compute zmat(5,1)=2*modvals(#m,1).
end if.
do if (model = 2 or model = 3).
compute zmat(2,1)=modvals(#m,2)**2.
compute zmat(3,1)=modvals(#m,1)**2.
compute
zmat(4,1)=(modvals(#m,1)**2)*(modvals(#m,2)
**2)*(1-yes2).
compute zmat(5,1)=2*modvals(#m,2)*(1-yes).
compute zmat(6,1)=2*modvals(#m,1)*(1-yes).
compute
zmat(7,1)=2*modvals(#m,1)*modvals(#m,2)*(1-
yes).
compute
zmat(8,1)=2*modvals(#m,1)*modvals(#m,2).
compute
zmat(9,1)=2*modvals(#m,1)*(modvals(#m,2)**
2)*(1-yes2).
compute
zmat(10,1)=2*(modvals(#m,1)**2)*modvals(#m
,2)*(1-yes2).
end if.
compute cfse(#m,1)=sqrt(csum(zmat&*cmat)).
end loop.
end if.
do if (nmods > 0).
compute clbs = {modvnm, 'Effect'}.
loop im = 1 to nmeds.
compute obs = t(indboot(1+(im-1)*(boot+1),:)).
compute outp = {modvals, obs}.
do if (model < 4).
compute tstat = obs&/cfse.

```

```

do if (dichy = 0).
compute pval = 2*(1-tcdf(abs(tstat), (n-
ncol(xy)))).
compute temp=obs-sqrt(jncrit)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp=obs+sqrt(jncrit)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 't', 'p', 'LLCI', 'ULCI'}.
compute jnclbs=clbs.
end if.
do if (dichy = 1).
compute pval = 2*(1-cdfnorm(abs(tstat))).
compute temp = obs-abs(xp2)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp = obs+abs(xp2)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute jnclbs=clbs.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indboot((1+(im-
1)*(boot+1)),:).
compute indboot2 = indboot((2+(im-
1)*(boot+1)):(1+(im-1)*(boot+1)+boot),:).
compute mnind = t(csum(indboot2)/boot).
compute stdind=t(sqrt((cssq(indboot2)-
((csum(indboot2)**2)/boot))/(boot-1))).
compute llci=make(1,ncol(indboot2),-999).
compute ulci=make(1,ncol(indboot2),-999).
loop #e = 1 to ncol(indboot2).
.
compute temp = indboot2(:,#e).
compute temp(GRADE( indboot2(:,#e) )) =
indboot2(:,#e).
compute badlo = 0.
compute badhi = 0.

```

```

do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.

```

```

do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {modvals, obs, stdind, t(llci),
t(ulci)}.
compute clbs = {modvnm, 'Effect', 'Boot SE',
'BootLLCI', 'BootULCI'}.
end if.
compute mtemp = mnames(1, im).
compute rlbs =
make(nrow(modvals),1,mnames(1,im)).
do if (model < 4).
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
do if (yes=0).
print outp/title = 'Conditional effect of X on Y at
values of the moderator(s):'/cnames =
clbs/format = F10.4.
else.
compute outp=outp(:,3:ncol(outp)).
compute clbs=clbs(:,3:ncol(clbs)).
compute clbs(1,1)='Contrast'.
print outp/title = 'Contrast of conditional effects
of X on Y'/cnames = clbs/format = F10.4.
end if.
end if.
do if (model > 5 and mod74dic <> 1).
do if (im = 1).
print/title = 'Conditional indirect effect(s) of X on
Y at values of the moderator(s):'.
end if.
print outp/title = 'Mediator'/rnames =
rlbs/cnames = clbs/format = F10.4.

```



```

else if (mod74dic = 1 and model = 74).
do if (im = 1).
print/title = 'Indirect effect(s) of X on Y:'.
compute clbs3=clbs(1,2:ncol(clbs)).
end if.
compute outp3=outp(1,2:ncol(outp)).
print outp3/title = 'Mediator'/rnames =
rlbs/cnames = clbs3/format = F10.4.
end if.
end loop.
loop i = notes to 1 by -1.
do if (note(i,1) = 4 and yes=0).
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 5 and yes=0).
print/title = 'Values for quantitative moderators
are the mean and plus/minus one SD from
mean.'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 14 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    below the mean was replaced
with the minimum because one SD below the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.
do if (note(i,1) = 15 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    above the mean was replaced
with the maximum because one SD above the
mean'/space=0.

```

```

print/title = '      is outside of the range of the
data.'/space=0.
end if.
end loop.
do if (model = 3 and yes=0).
compute jnvals=make(nrow(matw),7,0).
compute jnvals(:,1)=matw.
compute jnvals(:,2)=jnb1+jnb3*jnvals(:,1).
compute
jnvals(:,3)=sqrt(jnsb1+2*jnvals(:,1)*jnsb1b3+(jn
vals(:,1)*jnvals(:,1))*jnsb3).
compute jnvals(:,4)=jnvals(:,2)&/jnvals(:,3).
do if (dichy = 0).
compute jnvals(:,5)=2*(1-tcdf(abs(jnvals(:,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(:,5)=2*(1-
cdfnorm(abs(jnvals(:,4)))).
end if.
compute jnvals(:,6)=jnvals(:,2)-
sqrt(jncrit)*jnvals(:,3).
compute
jnvals(:,7)=jnvals(:,2)+sqrt(jncrit)*jnvals(:,3).
compute clbs={clbs(:,1),clbs(:,3:ncol(clbs))}.
do if (coeffci = 0).
compute jnvals =jnvals(:,1:(ncol(jnvals)-2)).
end if.
print jnvals/title = 'Conditional effect of X*M
interaction at values of W:.'/cnames
=clbs/format = F10.4.
end if.
do if (jn = 1 and (model = 1 or model = 3) and
jndich = 0 and yes=0).
print/title = '***** JOHNSON-
NEYMAN TECHNIQUE
*****'.
compute ajn =(jncrit*jnsb3)-(jnb3*jnb3).
compute bjn = 2*((jncrit*jnsb1b3)-(jnb1*jnb3)).
compute cjn = (jncrit*jnsb1)-(jnb1*jnb1).

```

```

compute radarg = (bjn*bjn)-(4*ajn*cjn).
compute den = 2*ajn.
compute nrts = 0.
do if (radarg >= 0 and den <> 0).
compute x21 = (-bjn+sqrt(radarg))/den.
compute x22 = (-bjn-sqrt(radarg))/den.
compute roots = 0.
do if (x21 >= jnmin and x21 <= jnmax).
compute nrts = 1.
compute roots = {roots; x21}.
end if.
do if (x22 >= jnmin and x22 <= jnmax).
compute nrts = nrts + 1.
compute roots = {roots; x22}.
end if.
compute roots={roots,make(nrow(roots),2,0)}.
compute modtemp=m.
do if (model=3).
compute modtemp=w.
end if.
do if (nrts > 0).
compute roots = roots(2:nrow(roots),1:3).
compute roots(1,2)=(csum(modtemp <
roots(1,1))/n)*100.
compute roots(1,3)=(csum(modtemp >
roots(1,1))/n)*100.
do if (nrow(roots)=2).
compute roots(2,2)=(csum(modtemp <
roots(2,1))/n)*100.
compute roots(2,3)=(csum(modtemp >
roots(2,1))/n)*100.
end if.
print roots/title = 'Moderator value(s) defining
Johnson-Neyman significance
region(s)'/clabels = 'Value', '% below', '%
above'/format = F10.4.
compute jnvals=make((21+nrts),7,0).
loop i= 0 to 20.
compute jnvals((i+1),1)=jnmin+(i*((jnmax-
jnmin)/20)).

```

```

end loop.
loop i = 1 to nrts.
loop j = 2 to nrow(jnvals).
do if ((roots(i,1) > jnvals((j-1),1)) and (roots(i,1)
< jnvals(j,1))).
compute
jnvals((j+1):(21+i),1)=jnvals(j:(20+i),1).
compute jnvals(j,1)=roots(i,1).
end if.
end loop.
end loop.
loop i = 1 to nrow(jnvals).
compute jnvals(i,2)=jnb1+jnb3*jnvals(i,1).
compute
jnvals(i,3)=sqrt(jnsb1+2*jnvals(i,1)*jnsb1b3+(jn
vals(i,1)*jnvals(i,1))*jnsb3).
compute jnvals(i,4)=jnvals(i,2)/jnvals(i,3).
do if (dichy = 0).
compute jnvals(i,5)=2*(1-tcdf(abs(jnvals(i,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(i,5)=2*(1-
cdfnorm(abs(jnvals(i,4)))).
end if.
compute jnvals(i,6)=jnvals(i,2)-
sqrt(jncrit)*jnvals(i,3).
compute
jnvals(i,7)=jnvals(i,2)+sqrt(jncrit)*jnvals(i,3).
end loop.
do if (model = 1).
print jnvals/title = 'Conditional effect of X on Y
at values of the moderator (M)'/cnames
=jnclbs/format = F10.4.
end if.
do if (model = 3).
compute
jnclbs={jnclbs(:,1),jnclbs(:,3:ncol(jnclbs))}.

```

```

print jnvals/title = 'Conditional effect of X*M on
Y at values of the moderator (W)'/cnames
=jnclbs/format = F10.4.
end if.
end if.
do if (nrts = 0).
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
else.
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
end if.
end if.
do if (model < 4 and (plot = 1)).
compute dataplot =
make((nrow(modvals)*nrow(matx)),(ncol(modv
als)+1),0).
compute tmp = 1.
loop i = 1 to nrow(modvals).
loop j = 1 to nrow(matx).
compute dataplot(tmp,:)={matx(j,1),
modvals(i,:)}.
compute tmp=tmp+1.
end loop.
end loop.
compute dataplot = {dataplot,
make(nrow(dataplot),(1+dichy),0)}.
compute dataplo2 = make(nrow(dataplot),1,1).
do if (model = 1).
compute dataplo2 = {dataplo2, dataplot(:,2),
dataplot(:,1), (dataplot(:,1)*dataplot(:,2))}.
end if.
do if (model = 2 or model = 3).
compute dataplo2 = {dataplo2, dataplot(:,3),
dataplot(:,1), (dataplot(:,1)*dataplot(:,3)),
dataplot(:,2), (dataplot(:,1)*dataplot(:,2))}.

```

```

do if (model = 3).
compute dataplo2 = {dataplo2,
(dataplot(:,2)*dataplot(:,3)),
(dataplot(:,1)*dataplot(:,2)*dataplot(:,3))}.
end if.
end if.
loop i = 1 to nrow(dataplot).
compute tmp=dataplo2(i,:).
do if (ncovs > 0).
compute tmp = {tmp, covmeans}.
end if.
do if (cluster > 0).
compute tmp = {tmp, cldmeans}.
end if.
compute dataplot(i,(ncol(dataplot)-
(dichy)))=tmp*coeffplt.
do if (dichy = 1).
compute
dataplot(i,(ncol(dataplot)))=exp(tmp*coeffplt)/(1
+exp(tmp*coeffplt)).
end if.
end loop.
compute clbs = {xname, modvnm, 'yhat'}.
do if (dichy = 1).
compute clbs = {xname, modvnm, 'ln(odds)',
'prob'}.
end if.
print/title =
*****
*****!

```

```

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (model = 1).
do if (dichy = 0).
print/title = 'DATA LIST FREE/Pro_Env_Strgy
Eco_Innovation Operational_Performance.'
/space=1.
else if (dichy = 1).
print/title = 'DATA LIST FREE/Pro_Env_Strgy
Eco_Innovation logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH Operational_Performance BY
Eco_Innovation.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Operational_Performance BY
Pro_Env_Strgy.' /space=0.
end if.
end if.
do if (dichy = 1).

```

```

do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH logodds BY Eco_Innovation.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH prob BY Eco_Innovation.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Pro_Env_Strgy.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Pro_Env_Strgy.' /space=0.
end if.
end if.
end if.
do if (model = 2 or model = 3).
do if (dichy = 0).
print/title = 'DATA LIST FREE/Pro_Env_Strgy
Eco_Innovation Operational_Performance.'
/space=1.
end if.
do if (dichy = 1).
print/title = 'DATA LIST FREE/Pro_Env_Strgy
Eco_Innovation logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH Operational_Performance BY
Eco_Innovation/PANEL ROWVAR=' /space=0.
end if.
do if (xdich<>0).

```



```

print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Operational_Performance BY
Pro_Env_Strgy/PANEL ROWVAR=.' /space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH logodds BY Eco_Innovation/PANEL
ROWVAR=.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH prob BY Eco_Innovation/PANEL
ROWVAR=.' /space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Pro_Env_Strgy/PANEL
ROWVAR=.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Pro_Env_Strgy/PANEL
ROWVAR=.' /space=0.
end if.
end if.
end if.
do if (ncovs > 0).
print/title = '* Estimates are based on setting
covariates to their sample means.'.
end if.
end if.
end if.
do if ((imm > 0) or (model = 74 and
mod74dic=0) or ((model = 58 or model = 59)
and wvdich=1))).
do if (imm > 4 and imm < 8).

```

```

compute
indbootp={indbootp(:,1:nmeds),indboop1,indbo
op2}.
end if.
compute obsprod = t(indbootp(1,:)).
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indbootp(1,:).
compute indbootp = indbootp(2:(boot+1),:).
compute mnindp = t(csum(indbootp)/boot).
compute stdindp=t(sqrt((cssq(indbootp)-
((csum(indbootp)&*2)/boot))/(boot-1))).
compute llcip=make(1,ncol(indbootp),-999).
compute ulcip=make(1,ncol(indbootp),-999).
loop #e = 1 to ncol(indbootp).

.
compute temp = indbootp(:,#e).
compute temp(GRADE( indbootp(:,#e) )) =
indbootp(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).

```

```

compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcip(1,#e)=llcit.
compute ulcip(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {obsprod, stdindp, t(llcip),
t(ulcip)}.
compute clbs = {'Effect', 'SE(Boot)', 'BootLLCI',
'BootULCI'}.
do if (model = 8 or model = 12).
print/title='-----'.
compute outpimm=outp(1:nmeds,:).

```

```

print/title = 'Indirect effect of highest order
product:.'/space=0.
print outpimm/title = 'Mediator'/cnames =
clbs/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
compute clbs3 = {'Index', 'SE(Boot)',
'BootLLCI', 'BootULCI'}.
do if (imm < 5).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=outp((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
do if (imm > 4 and imm < 8).

```

```

print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'.

compute outpimm=outp(1:nmeds,:).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
compute
outpimm=outp((nmeds+1):nrow(outp),:).
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.

loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),clbs3}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3(1,1)=modvnm(1,1).
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.

```

```

print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
end if.
do if (boot = 0).
do if (model = 8 or model = 12).
print/title='-----'.
compute obsimm=obsprod(1:nmeds,:).
print/title = 'Indirect effect of highest order
product.'/space=0.
print obsimm/title = 'Mediator'/clabels =
'Effect'/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
do if ((imm > 0 and imm < 5) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=obsprod((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
end loop.
end if.

```

```

do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****!
.
compute outpimm=obsprod(1:nmeds,1).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
compute
outpimm=obsprod((nmeds+1):nrow(obsprod),:)
.
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),'Index'}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3={modvnm(1,1),'Index'}.
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator: '/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
end if.
end if.

```

```

end if.
compute conmake=0.
compute concols = 0.
do if ((model > 3 and model < 7) and (contrast
> 0) and nmods = 0 and nmeds > 1 ).
compute concols =
((ncol(indboot)*(ncol(indboot)-1))/2).
compute indcon=make(nrow(indboot),concols,-
999).
compute conkey = { ' ', ' ', ' ', ' ' }.
compute temp=1.
compute conmake=1.
loop i = 1 to (ncol(indboot)-1).
loop j = (i+1) to (ncol(indboot)).
compute indcon(:,temp)=indboot(:,i)-
indboot(:,j).
do if (contrast = 2).
compute indcon(:,temp)=abs(indboot(:,i))-
abs(indboot(:,j)).
end if.
do if (model <= 6).
compute conkey={conkey; mnames(1,i),
'minus', mnames(1,j)}.
end if.
do if (model = 6).
compute conkey={conkey; indlbl2(i,1), 'minus',
indlbl2(j,1)}.
end if.
compute temp=temp+1.
end loop.
end loop.
end if.
do if (model = 4 or model = 5).
compute clbs = {'Effect'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obs = {csum(obs); obs}.
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.

```



```

compute rlbs = {rlbs;
cntname(1:ncol(indcon),1)}.
end if.
compute outp = obs.
compute outp2=outp.
do if (effsize = 1 and dichy = 0).
compute
pmeff(:,1)=rsum(pmeff(:,2:ncol(pmeff))).
compute
rmeff(:,1)=rsum(rmeff(:,2:ncol(rmeff))).
compute
abpseff(:,1)=rsum(abpseff(:,2:ncol(abpseff))).
compute
abcseff(:,1)=rsum(abcseff(:,2:ncol(abcseff))).
compute eff = {pmeff, rmeff, abpseff, abcseff}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute eff = {eff, r245}.
compute r245obs = {r245(1,1);r245(1,1)}.
compute kappa2ob = kappa2(1,1).
end if.
compute pmobs = t(pmeff(1,1:(nmeds+1))).
compute rmobs = t(rmeff(1,1:(nmeds+1))).
compute psobs = t(abpseff(1,1:(nmeds+1))).
compute csobs = t(abcseff(1,1:(nmeds+1))).
do if (contrast = 0).
compute outp2 = {obs, psobs, csobs, pmobs,
rmobs}.
end if.
do if (contrast > 0).
compute obs2=obs(1:nrow(psobs),:).
compute outp2 = {obs2, psobs, csobs, pmobs,
rmobs}.
end if.
compute clbs = {'ab', 'ab_ps', 'ab_cs', 'ab/c',
'ab/c'}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute outp2 = {outp2, r245obs}.
compute clbs = {clbs, 'R-sq_med'}.

```

```

do if (boot = 0 and mc = 0).
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
do if (boot = 0 and mc = 0).
do if (nmeds = 1).
compute outp2 = outp2(2,:).
compute rlbs = rlbs(2,1).
end if.
do if (model=5).
compute outp2=outp2(:,1:3).
end if.
print outp2/title = 'Indirect effect(s) of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1 and nmeds >
1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0 or mc > 0).
compute indboot = {rsum(indboot), indboot}.
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
compute ones = make(bootSZ,1,1).
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(bootSZ+1),:).
compute mnind = t(csum(indboot)/bootSZ).

```

```

compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/bootSZ))/(bootSZ-1))).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).

.

compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.

```

```

end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llici(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
do if (effsize = 1 and dichy = 0 and boot > 0).
compute estmte=eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).
.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.

```

```

compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).

```

```

compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
end if.
end if.
do if (boot > 0 or mc > 0).
compute outp = {obs, stdind, t(llci), t(ulci)}.
do if (nmeds = 1).
compute outp = outp(2,:).
compute rlbs = rlbs(2,1).
end if.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
do if (mc > 0).
compute clbs = {'Effect', 'MC SE', 'MC LLCI',
'MC ULCI'}.
end if.
print outp/title = 'Indirect effect of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (dichy = 0 and effsize = 1 and boot > 0).
compute outp = {psobs,
stdindf((2*(nmeds+1)+1):(3*(nmeds+1)),1),
t(llcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))),
t(ulcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Partially standardized indirect
effect of X on Y'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute outp = {csobs,
stdindf((3*(nmeds+1)+1):(4*(nmeds+1)),1),
t(llcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))),
t(ulcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.

```

```

print outp/title = 'Completely standardized
indirect effect of X on Y'/rnames = rlbs/cnames
= clbs/format = F10.4.
compute outp = {pmobs,
stdindf(1:(nmeds+1),1),t(llcif(1,1:(nmeds+1))),
t(ulcif(1,1:(nmeds+1))) }.
do if (model = 4).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
compute outp = {rmobs, stdindf(((nmeds+1)+1)
:(2*(nmeds+1)) ,1),
t(llcif(1,((nmeds+1)+1):(2*(nmeds+1)))),
t(ulcif(1,((nmeds+1)+1):(2*(nmeds+1))))}.
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
do if (nmeds = 1 and cluster = 0 and ncovs = 0
and model = 4).
compute r245obs = r245obs(1,1).
compute outp = {r245obs,
stdindf((4*(nmeds+1)+1):(4*(nmeds+1)+1),1),
t(llcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1))),
t(ulcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1)))}.
print outp/title = 'R-squared mediation effect
size (R-sq_med)'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
end if.
do if (normal = 1).

```

```

do if (nmeds = 1).
print sobel/title = 'Normal theory tests for
indirect effect'/clabels = 'Effect', 'se', 'Z',
'p'/format = F10.4.
end if.
do if (nmeds > 1).
compute rlbs2 = rlbs(2:nrow(rlbs),1).
print sobel/title = 'Normal theory tests for
specific indirect effects'/rnames = rlbs2/clabels
= 'Effect', 'se', 'Z', 'p'/format = F10.4.
end if.
end if.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
do if (model = 6).
compute clbs = {'eff'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obsfsum=csum(obs).
compute obs = {obsfsum; obs}.
compute indlbl = indlbl(1:nrow(obs),1).
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute indlbl = {indlbl;
cntname(1:ncol(indcon),1)}.
end if.
compute obs2=obs.
do if (boot = 0).
do if (dichy = 0 and effsize = 1).
compute clbs = {'eff', 'eff_ps', 'eff_cs', 'eff/c',
'eff/c'}.
do if (ncovs = 0).

```



```

compute obs = {obs, obs/stddevy,
(obs*stddevx/stddevy), obs/ctot,
obs/mmpaths(nrow(mmpaths),1)}.
end if.
do if (ncovs > 0 or clsdmy > 0).
compute obs = {obs, obs/sdycov,
(obs*sdxcov/sdycov),
obs/(obsfsum+mmpaths(nrow(mmpaths),1)),
obs/mmpaths(nrow(mmpaths),1)}.
end if.
compute obs2=obs.
do if (contrast > 0).
compute obs2 = obs(1:(nrow(obs)-concols),:).
end if.
end if.
print obs2/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute indboot = {rsum(indboot), indboot}.
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(boot+1),:).
compute mnind = t(csum(indboot)/boot).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/boot))/(boot-1))).
compute temp = nrow(indboot).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).

```

```

loop #e = 1 to ncol(indboot).

.

compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.

```

```

do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute obs = {obs, stdind, t(llci), t(ulci)}.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
print obs/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (effsize = 1 and dichy = 0).
compute indboot=indboot(:,1:(ncol(indboot)-
concols)).
compute eff = {rsum(abpseff), abpseff,
rsum(abcseff), abcseff, rsum(pmeff), pmeff,
rsum(rmeff), rmeff}.
compute effobs = eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).
.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).

```

```

compute badlo = 0.
compute badhi = 0.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(effobs(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.

```

```

compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute temp2 = stdindf(1:ncol(indboot),1).
compute temp3 = effobs(:,1:ncol(indboot)).
compute templo = llcif(1,1:ncol(indboot)).
compute temphi = ulcif(1,1:ncol(indboot)).
compute outp = {t(temp3), temp2,t(templo),
t(temphi)}.
print outp/title = 'Partially standardized indirect
effect of X on Y'/cnames = clbs/rnames =
indlbl/format = F10.4.
compute temp2 =
stdindf((ncol(indboot)+1):(2*ncol(indboot)),1).
compute temp3 =
effobs(:,(ncol(indboot)+1):(2*ncol(indboot))).
compute templo =
llcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute temphi =
ulcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templo),
t(temphi)}.
print outp/title = 'Completely standardized
indirect effect of X on Y'/cnames = clbs/rnames
= indlbl/format = F10.4.
compute temp2 =
stdindf((2*(ncol(indboot))+1):(3*ncol(indboot)),1
).
compute temp3 =
effobs(:,(2*(ncol(indboot))+1):(3*ncol(indboot)))
.

```

```

compute templow =
  llcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute temphi =
  ulcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
  t(temphi)}.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
compute temp =
  eff(:,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temp2 =
  stdindf((3*(ncol(indboot))+1):(4*ncol(indboot)),1
  ).
compute temp3 =
  effobs(:,(3*(ncol(indboot))+1):(4*ncol(indboot)))
  .
compute templow =
  llcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temphi =
  ulcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
  t(temphi)}.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
end if.
end if.
do if (nmeds = 2).
compute effkey = {xname, '->', mnames(1,1), '-
->', yname, '', ''}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, '', ''}.
compute effkey = {indlbl(2:4,1), effkey}.
end if.
do if (nmeds = 3).
compute effkey = {xname, '->' , mnames(1,1), '-
->', yname, '', '', ''}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, '',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, '',' ',' ',' '}.
compute effkey = {indbl(2:8,1), effkey}.
end if.
do if (nmeds = 4).
compute effkey = {xname, '->' , mnames(1,1), '-
>', yname, '',' ',' ',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,4), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname, '',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,4), '->', yname, '',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, '',' '}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', mnames(1,4), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,4), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', mnames(1,4), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,4), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {indlbl(2:16,1), effkey}.
end if.
print effkey/title = 'Indirect effect key'/format =
A8.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
end if.
DO IF (medte2=1).
compute
name=t(mdlnms2(2:nrow(mdlnms2),1)).
compute tmpdat=datayed(:,2:ncol(datayed)).
compute dd=tmpdat(:,1).

```



```

compute
dd={dd,tmpdat(:,(2+nmeds)),tmpdat(:,2:(1+nmeds))}.
compute stratify=0.
do if (ncovs > 0).
compute cname = cnames.
compute
dd={dd,tmpdat(:,(3+nmeds):(2+nmeds+ncovs))}.
end if.
compute method = 1.
compute samples = boot.
do if (mc > 0).
compute samples = mc.
compute method=2.
end if.
compute omnibus = 1.
compute booterr = 0.
do if (mcx < 0 or mcx > 4).
compute mcx = 1.
end if.
compute ovals = ncol(design(dd(:,1))).
compute nte = notes.
compute criterr = 0.
compute xskip = 0.
compute dumok = 0.
compute badend = 0.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute priorlo = -9999999.
compute priorhi = 9999999.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute reps = samples.
do if (contrast = 1).
compute note(nte,1) = 26.
compute nte=nte+1.

```

```

end if.
do if (effsize = 1).
compute note(nte,1) = 27.
compute nte=nte+1.
end if.
do if (normal = 1).
compute note(nte,1) = 28.
compute nte=nte+1.
end if.
do if (method = 2).
do if (omnibus = 1).
compute note(nte,1) = 25.
compute nte=nte+1.
end if.
compute samples = 0.
do if (reps < 1000).
compute reps = 1000.
end if.
end if.
do if (method = 1).
compute samples = reps.
end if.
compute temp = ncol(dd).
compute n = nrow(dd).
compute ones = make(n,1,1).
compute temp=dd(:,2).
compute nxd = ncol(temp).
compute nx = nxd.
do if (nx = 1 and mcx > 0).
compute temp = dd.
compute temp(GRADE(dd(:,2)),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,2)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute mnvls = cmin(t(nnvls)).
do if (mnvls < 2).
compute errs = errs + 1.
compute runerrs(errs,1) = 37.
compute criterr = 1.

```

```

end if.
do if (nvls > 9).
compute errs = errs+1.
compute runerrs(errs,1) = 36.
compute criterr = 1.
end if.
do if (criterr = 0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,2).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,2) <> nnvls((temp-1),1)).
compute nnvls(temp,1)=dd(i,2).
compute temp = temp+1.
end if.
end loop.
do if (mcx > 0).
compute x = dummy(:,2:ncol(dummy)).
compute nx = ncol(x).
compute minus1 = make(1,ncol(x),-1).
compute xdes=make((nx+1),3,0).
compute xdes(1,1)=dd(1,2).
compute xdes(1,2)=1.
compute temp = 2.
loop k = 2 to n.
do if (dd(k,2) <> dd((k-1),2)).
compute xdes(temp,2) = k.
compute xdes(temp,1) = dd(k,2).
compute xdes((temp-1),3) = k-1.
compute temp=temp+1.
end if.
end loop.
compute xdes((temp-1),3)=n.
compute xdes = {xdes, (xdes(:,3)-xdes(:,2)+1)}.
do if (stratify <> 0).
compute note(nte,1) = 7.
compute nte=nte+1.
end if.
do if (mcx = 4).

```

```

loop k = 1 to n.
do if (rsum(x(k,:)) = 0).
compute x(k,:) = minus1.
end if.
end loop.
end if.
do if (mcx = 2 or mcx = 3).
compute note(nte,1)=4.
compute nte = nte+1.
loop k = 1 to n.
do if (rsum(x(k,:)) > 0).
loop i = 1 to ncol(x).
do if (x(k,i) = 0).
compute x(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcx = 3).
compute conmat1={-8,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1; 0,0,0,0,0,-
3,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.
loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,:)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute x(k,:)=conmat1((rsum(x(k,:))+1),:).
end loop.
end if.
end if.
compute xdata = x.
compute xskip = 1.
compute xname = ddd(1,1:nx).
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).

```

```

compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcx = 4).
compute dummat(1,:) = minus1.
end if.
do if (mcx = 2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcx = 3).
compute dummat=conmat1.
end if.
compute dummat={nnvls, dummat}.
compute xname2={name(1,2), xname}.
end if.
end if.
end if.
compute nm=nmeds.
compute r2b = make(nm,1,-999).
compute indeff = make((nx*nm),1,-999).
compute cov =
make((nx*nm+nm),(nx*nm+nm),0).
compute mns=make(nrow(cov),1,0).
compute ny = ncol(dd)-nm-nxd-ncovs.
compute y = dd(:,1).
compute ydata = y.
compute yname = name(1,1).
do if (xskip = 0).
compute x = dd(:,2:(1+nxd)).
compute xdata = x.
compute xname = name(1,2:(1+nxd)).
compute indlbs = t(xname).
end if.
compute m = dd(:,(2+nxd):(1+nxd+nm)).
compute mdata = m.

```

```

compute mlab = { ' M1 = ' ; ' M2 = ' ; ' M3 = ' ; '
M4 = ' ; ' M5 = ' ; ' M6 = ' ; ' M7 = ' ; ' M8 = ' ; '
M9 = ' ; ' M10 = ' }.
compute mlab = {mlab; ' M11 = ' ; ' M12 = ' ; '
M13 = ' ; ' M14 = ' ; ' M15 = ' }.
compute mname =
name(1,(2+nxd):(1+nxd+nm)).
do if (ncovs > 0).
compute c =
dd(:,(2+nxd+nm):(1+nxd+nm+ncovs)).
compute x = {x, c}.
compute xdata = x.
compute cdata = c.
compute xname = {xname, cname}.
end if.
compute nms={yname; t(mname);
t(xname(1,1:(ncol(xname)-ncovs)))}.
do if (dumok = 1).
compute nms={yname; t(mname); name(1,2)}.
end if.
do if (nm < 16).
do if (dumok = 1).
print dummat/title = 'Coding of categorical X
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
compute
boots=make((samples+1),((nx+1)*nm),-999).
do if (errs = 0 and criterr = 0).
loop k = 1 to (samples+1).
do if (k = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef=0.
do if (k > 1).
compute u=trunc(uniform(n,1)*n)+1.
compute xdata = x(u,:).
compute mdata = m(u,:).

```

```

compute ydata = y(u,:).
do if (ncovs > 0).
compute cdata = c(u,:).
end if.
end if.
loop i = 1 to nm.
compute start = 1+(i-1)*nx.
compute xmat = {ones, xdata}.
do if (covmy=2).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = mdata(:,i).
compute amat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*amat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-(((1-r2)*(n-1))/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute r2cha = r2.
compute adjr2cha = adjr2.
do if (ncovs > 0 and covmy <> 2).
compute xmat = {ones, cdata}.
compute atmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*atmat)**2).
compute r2step1 = (sstotal-ssresid2)/sstotal.
compute r2cha = r2-r2step1.
compute adjr2cha = adjr2- (1-(((1-r2step1)*(n-
1))/(n-ncol(xmat)))).
compute xmat = {ones, xdata}.
end if.
compute bootcoef={bootcoef,t(amat)}.
compute r2b(i,1)=adjr2cha.
do if (k = 1).

```

```

compute hcinvtX=inv(t( xmat )' xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,1)*hcinvtX*t( xmat
(i3,1)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvtX*t( xmat )' xmat
*hcinvtX
.
compute tratio = amat&/sqrt(diag(covmat)).
compute rnms = mname(1,i).
print rnms/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.
compute lmat = ident(nrow(amat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*amat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*amat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/'format = F10.4
/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'.

```



```

compute op = {amat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*(xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute cov(start:(start+(nx-
1)),start:(start+(nx-
1)))=covmat(2:(ncol(covmat)-(ncovs*(covmy <>
2))),2:(ncol(covmat)-(ncovs*(covmy <> 2)))).
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns(start:(start+(nx-
1)),1)=amat(2:(1+nx)).
end loop.
compute xmat = {ones,mdata,xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (k = 1).
print yname/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.

```

```

do if (ovals = 2).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:'/cnames = nmsd/format = F9.2.
end if.
end if.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(yamat)/n)).
compute LL3 = yamat*&ln(pt2)+(1-yamat)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= yamat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).

```

```

compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute bootcoef={bootcoef,t(bmat)}.
compute LL1 = LL2.
compute covmat=varb.
end if.
do if (ovals <> 2).
compute bmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*bmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-(((1-r2)*(n-1))/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute bootcoef={bootcoef,t(bmat)}.
end if.
do if (k = 1).
do if (ovals = 2).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.

```

```

compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).
compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms2/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute
opdirsum=outp((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
end if.
do if (ovals <> 2).
.
compute hcinvXtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtXt*( xmat
(i3,:)).

```

```

end loop.
loop i3=1 to k3.
compute xmat(:,i3) = ( resid(:,ncol( resid
))&/(1-h))&* xmat(:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat(:,i3) = sqrt( mse )&* xmat(:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(bmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/(ncol(xmat)-1).
compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat((2+nmeds):(2+nmeds+(nx-1)),:)=
lmat2.
compute fratiodr =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/nx.
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute df2=(n-ncol(xmat)).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), df2, pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = bmat&/sqrt(diag(covmat)).
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute op = {bmat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat)))) }.

```

```

compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp))))*(xp2/(temp-
(2/3)+(.11/temp)))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms2/format =
F10.4.
compute
opdirsum=op((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
compute xmat = {ones,mdata}.
do if (ncovs > 0).
compute xmat = {xmat, cdata}.
end if.
compute btmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*btmat)**2).
compute r2direct = r2-((sstotal-
ssresid2)/sstotal).
compute pfr = 1-fcdf(fratiodr,(nx),(n-1-nx-nm-
ncovs)).
compute r2direct = {r2direct, fratiodr, nx, (n-1-
nx-nm-ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms2/cnames=rnms2/form
at= F10.4.
end if.
end if.

```

```

do if (k = 1 and toteff = 1).
print yname/title = '*****'
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:.'/format = A8.
compute xmat = {ones, xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(ymat)/n)).
compute LL3 = ymat*&ln(pt2)+(1-ymat*)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= ymat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.

```

```

compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute LL1 = LL2.
compute covmat=varb.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(bmat)-
1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms = {'Constant'; t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).

```



```

compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute optotsum=outp(2:(1+nx),:).
end if.
do if (ovals <> 2).
compute cmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*cmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).

.

compute hcinvtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).

```

```

loop i3=1 to k3.
compute xmat(:,i3) = sqrt( mse )&* xmat(:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(cmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = cmatrix/sqrt(diag(covmat)).
compute op = {cmatrix, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*(xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute optotsum=op(2:(1+nx),:).
compute ssresid2 = ssresid.
compute r2total = r2.

```

```

compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat(2:(1+nx),:) = lmat2.
compute fratio3 =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/nx.
do if (ncovs > 0).
compute xmat = {ones, cdata}.
compute ctmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*ctmat)**2).
compute r2total = r2-((sstotal-ssresid2)/sstotal).
end if.
compute pfr = 1-fcdf(fratio3,(nx),(n-1-nx-
ncovs)).
compute r2tot2 = {r2total, fratio3, nx, (n-1-nx-
ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns((nrow(mns)-
nm+1):nrow(mns),1)=bmat(2:(1+nm),1).
compute tmp = 1.
loop i = 1 to (nm*nx).
compute
boots(k,i)=mns(i,1)*mns(((nm*nx)+tmp),1).
do if ((i/nx)=trunc(i/nx)).
compute
boots(k,((nm*nx)+tmp))=r2b(tmp,1)*mns(((nm*
nx)+tmp),1).
compute tmp = tmp+1.
end if.
end loop.
do if (savboot = 1 and k > 1).

```

```

compute bootstrp((k-1),:)=bootcoef.
end if.
end loop.
do if (method = 1).
compute llci=make(1,ncol(boots),-999).
compute ulci=make(1,ncol(boots),-999).
compute indeff = t(boots(1,1:(nm*nx))).
compute indeff22=t(boots(1,:)).
compute omni = t(boots(1,((nm*nx)+1) :
ncol(boots))).
do if (nrow(boots) > 1).
compute boots = boots(2:nrow(boots),:).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
loop i = 1 to ncol(boots).
.
compute temp = boots(:,i).
compute temp(GRADE( boots(:,i) )) = boots(:,i).
compute badlo = 0.
compute badhi = 0.
do if ( indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(indeff22(i,1)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.

```

```

do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,i)=llcit.
compute ulci(1,i)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute llci = t(llci).
compute ulci= t(ulci).
compute llci={llci,ulci}.
end if.
end if.
do if (method = 2).
compute tmp=1.
loop i = 1 to (nm*nx).
compute
indeff(i,1)=mns(i,1)&*mns(((nm*nx)+tmp),1).

```

```

do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = sqrt(-
2*ln(uniform(reps,nrow(cov))))&*cos((2*3.1415
9265358979)*uniform(reps,nrow(cov))).
compute boots=boots*chol(cov).
loop i = 1 to ncol(boots).
compute boots(:,i)=boots(:,i)+mns(i,1).
end loop.
compute tmp=1.
loop i = 1 to (nm*nx).
compute
boots(:,i)=boots(:,i)&*boots(:,((nm*nx)+tmp)).
compute temp = boots(:,i).
compute temp(GRADE(boots(:,i))) = boots(:,i).
compute boots(:,i) = temp.
do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = boots(:,1:(nm*nx)).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
compute llci =
t(((boots(cilow,:));(boots(cihigh,:)))).
end if.
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
compute rnmssum=rnms(2:(1+nx),1).
do if (ovals <> 2).
print optotsum/title = 'Relative total effects of X
of Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.

```

```

print r2tot2/title = 'Omnibus test of total effect of
X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print optotsum/title = 'Relative total effects of X
on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
end if.
compute
rnmssum=rnms2((2+nmeds):(1+nmeds+nx),1).
do if (ovals <> 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.
print r2direct/title = 'Omnibus test of direct
effect of X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
do if (omnibus = 1 and (method = 1 or method
= 3)).
compute indlbs = {indlbs; 'Omnibus'}.
end if.
loop i = 1 to nm.
compute op = mname(1,i).
print op/title = 'Relative indirect effect(s) of X on
Y through:.'/format = A8/space=0.
do if (method = 2).
compute clbs = {'Effect', 'SE MC', 'MC LLCI',
'MC ULCI'}.
compute op = {indeff, bootse(1:(nm*nx)), lici}.
end if.

```

```

do if (method = 1).
compute clbs = {'Effect', 'SE(boot)', 'LLCI',
'ULCI'}.
compute op = indeff.
do if (samples > 0).
compute op = {indeff, bootse(1:(nm*nx),:),
llci(1:(nm*nx),:)}).
end if.
end if.
compute temp = op((((i-1)*nx)+1):(nx*i),:).
do if ((method = 1) and omnibus = 1).
do if (samples = 0).
compute temp={temp;omni(i,1)}.
end if.
do if (samples > 0).
compute temp = {temp; omni(i,1),
bootse(((nm*nx)+i),:), llci(((nm*nx)+i),:)}).
end if.
end if.
print temp/title = ' '/rnames = indlbs/cnames =
clbs/format = F10.4 /space=0.
print/title = '-----'.
end loop.
end if.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
END IF.
DO IF (mdpbe=1).
compute nm={yname,xname,mnames}.
do if (ncovs > 0).

```



```

compute
nm={yname,cnames,xname,mnames}.
end if.

compute tpx=xname.
compute x = dd(:,2:ncol(dd)).
compute tpy = yname.
compute tempy= dd(:,1).
compute n = nrow(dd).
compute nx=1.
compute criterr=0.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute ddd1 = {'int_1', 'int_2', 'int_3', 'int_4',
'int_5', 'int_6', 'int_7', 'int_8', 'int_9'}.
compute dumok = 0.
compute mcfoc=mcx.
compute mcmmod=mcm.
compute mcloc=(mcfoc>0).
compute con = make(n,1,1).
compute ncovs=ncol(x)-2.
do if (mcfoc > 0 or mcmmod > 0).
compute temp = dd.
compute temp(GRADE(dd(:,(ncol(dd)-
mcloc))),:)= dd.
compute dd = temp.
compute dummy = design(dd(:,ncol(dd)-
mcloc)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute toosmall=rsum(nnvls < 2).
compute mnvls = cmin(t(nnvls)).
do if ((rsum(nnvls < 2)) > 0).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=37.
end if.
do if (nvls > 10).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=36.

```

```

end if.
do if (criterr=0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,ncol(dd)-mcloc).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,ncol(dd)-mcloc) <> nnvls((temp-
1),1)).
compute nnvls(temp,1)=dd(i,ncol(dd)-mcloc).
compute temp = temp+1.
end if.
end loop.
compute dummy = dummy(:,2:ncol(dummy)).
do if (mcfoc=4 or mcmmod=4).
compute minus1=make(1,ncol(dummy),-1).
loop k = 1 to n.
do if (rsum(dummy(k,:)) = 0).
compute dummy(k,:)=minus1.
end if.
end loop.
end if.
do if (mcfoc = 2 or mcfoc = 3) or (mcmmod = 2 or
mcmmod = 3)).
loop k = 1 to n.
do if (rsum(dummy(k,:)) > 0).
loop i = 1 to ncol(dummy).
do if (dummy(k,i) = 0).
compute dummy(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcfoc = 3 or mcmmod=3).
compute conmat1={-8,1,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1,1; 0,0,0,0,0,-
3,1,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.

```

```

loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,:)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute
dummy(k,:)=conmat1((rsum(dummy(k,:))+1),:).
end loop.
end if.
end if.
compute nx = ncol(dummy).
compute xname = ddd(1,1:nx).
compute xdata = dummy.
compute xname = ddd(1,1:nx).
compute xname2={nm(1,(ncol(nm)-mcloc)),
xname}.
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcfoc = 2 or mcmmod=2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcfoc = 3 or mcmmod=3).
compute dummat=conmat1.
end if.
do if (mcfoc = 4 or mcmmod = 4).
compute dummat(1,:)=minus1.
end if.
compute dummat={nnvls, dummat}.
end if.
end if.
do if (criterr=0).
compute y = dd(:,1).
compute ovals = ncol(design(dd(:,1))).

```

```

compute itprob = 0.
do if (ovals = 2).
compute omx = cmax(y(:,1)).
compute omn = cmin(y(:,1)).
compute y(:,1) = (y(:,1) = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute sstotal = csum((y-(csum(y)/n))&**2).
compute outv = t(nm(1,1)).
compute mdtr = nm(1,ncol(dd)).
compute fciv = nm(1,(ncol(dd)-1)).
do if (mcfoc <> 0 or mcmod <> 0).
compute inter=make(n,nx,0).
loop i = 1 to nx.
compute inter(:,i)=dummy(:,i)&*dd(:,ncol(dd)-
(1-mcloc)).
end loop.
do if (ncovs = 0).
compute x={con,dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
do if (ncovs > 0).
compute
x={con,dd(:,2:(1+ncovs)),dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={t(nm(1,2:(1+ncovs)));
nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
end if.
compute xmns = csum(x)/n.
compute focvals = {1,0,0}.
compute highwarn = 0.
compute lowwarn = 0.

```

```

do if (ncol(x) > (2*nx+2)).
compute covmns = {1, xmns(1,2:(ncol(x)-1-
(2*nx)))}.
compute focvals = {covmns,0,0}.
end if.
compute dfres = n-ncol(x).
do if (ovals = 2).
compute pt2 = make(n,1,(csum(y)/n)).
compute LL3 = y*&ln(pt2)+(1-y)&ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(x),1,0).
compute LL1 = 0.

.

compute pt1lp= pt1.
compute xlp= x.
compute ylp= y.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).

```

```

compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
else if (ovals <> 2).
compute invXtX=inv(t(x)*x).
compute b = invXtX*t(x)*y.
compute resid = y-(x*b).
compute ssresid=csum(resid*resid).
compute msresid=ssresid/dfres.
compute varb = msresid*invXtX.
compute
seint=sqrt(varb(nrow(varb),nrow(varb))).
compute k3 = nrow(b).
do if (hc3 = 1).
compute xhc=x.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)= xhc(i3,:)*invXtX*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).
end loop.
compute varb=(invXtX*t(xhc)*xhc*invXtX).
end if.
compute seb = sqrt(diag(varb)).
compute r2 = 1-(ssresid/sstotal).

```

```

do if (mcfoc <> 0 or mcmod <> 0).
compute ytp=y.
compute xtp=x(:,1:(ncol(x)-nx)).
compute btp = inv(t(xtp)*xtp)*t(xtp)*ytp.
compute residt = ytp-(xtp*btp).
compute ssresidt=csum(residt*residt).
compute r2noint = 1-(ssresidt/sstotal).
end if.
compute pr = ncol(x)-1.
compute lmat = ident(nrow(b)).
compute lmat = lmat(:,2:ncol(lmat)).
compute f =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
pr).
compute pf = 1-fcdf(f,pr,dfres).
compute pf = {sqrt(r2),r2,msresid,f,pr,dfres,pf}.
compute xd=abs(xp2).
end if.
do if (itprob = 0).
compute tstat = b&/seb.
do if (ovals <> 2).
compute p = 2*(1-tcdf(abs(tstat), dfres)).
else if (ovals = 2).
compute p = 2*(1-cdfnorm(abs(tstat))).
end if.
compute outp = {b,seb,tstat,p}.
compute nms = {'constant'; nms; 'interact'}.
do if (ovals <> 2).
compute tval = (dfres* (exp((dfres-
(5/6))*((xd/(dfres-(2/3)+(.11/dfres)))*(xd/(dfres-
(2/3)+(.11/dfres)))))-1)).
compute outp={outp,(b-
sqrt(tval)&*seb),(b+sqrt(tval)&*seb)}.
end if.
compute bb = tval.
do if (dumok = 1).
do if (mcfoc > 0).
print dummat/title = 'Coding of categorical X
variable for analysis:'/cnames = xname2/format
= F5.2.

```

```

else if (mcmod > 0).
print dummat/title = 'Coding of categorical M
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
print yname/title =
*****
*****'/rlabels = 'Outcome: '/format
= A8.
do if (ovals = 2).
compute nmsd = {outv, 'Analysis'}.
print rcd/title = 'Coding of binary Y for
analysis: '/cnames = nmsd/format = F10.4.
compute outp = {outp,(outp(:,1)-
sqrt(tval)&*outp(:,2)),(outp(:,1)+sqrt(tval)&*outp
(:,2))}.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
print outp/title = 'Logistic Regression
Model'/rnames = nms/clabels 'Coeff' 'se' 'Z' 'p'
'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
compute varbtmp=varb.
compute btmp=b.

```



```

compute LL2f=LL2.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=x(:,1:(ncol(x)-nx)).
compute bt1 = make(ncol(xlp),1,0).

.

compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(y-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)*&ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.

```

```

compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL2f.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(LLdiff,nx).
compute rcha={LLdiff,nx,pchi}.
end if.
do if (ovals <> 2).
print pf/title = 'Model Summary'/clabels = 'R' 'R-
sq' 'MSE' 'F' 'df1' 'df2' 'p'/format F10.4.
compute lmat=make(nrow(b),1,0).
compute lmat(nrow(lmat),1)=1.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
1).
compute
rcha=((b(nrow(b),1)/seint)*(b(nrow(b),1)/seint))*
(1-r2)/dfres.
compute rcha = {rcha, fcha, 1, dfres,
outp((pr+1),4)}.
print outp/title = 'Model'/rnames = nms/clabels
'coeff' 'se' 't' 'p' 'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
do if (mcfoc > 0 or mcmod > 0).
compute rcha=r2-r2noint.
compute lmat=make((nrow(b)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
nx).
compute pvalr2c=1-fcdf(fcha,nx,dfres).

```

```

compute rcha = {rcha, fcha, nx, dfres,pvalr2c}.
end if.
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute intkey = {'a', 'b', 'c', 'd', 'e'}.
loop i = 1 to nx.
compute intkey={intkey; ddd1(1,i), ' : ', ddd(1,i),
' X ', nm(1,ncol(dd)-(1-mcloc))}.
end loop.
compute intkey=intkey(2:nrow(intkey),:).
print intkey/title='Product terms key:'/format =
a8.
end if.
do if (ovals <> 2).
print rcha/title = 'R-square increase due to
interaction:.'/clabels 'R2-chng' 'F' 'df1' 'df2'
'p'/format F10.4.
else if (ovals = 2).
print rcha/title = 'Likelihood ratio test for
interaction:.'/clabels 'Chi-sq' 'df' 'p'/format F10.4.
end if.
print/title =
*****
*****'.
compute mdvar = x(:,(ncol(x)-1)).
do if (mcfoc = 0 and mcmmod = 0).
compute g1 = b((ncol(x)-2),1).
compute g3 = b(ncol(x),1).
compute vg1 = varb((ncol(x)-2),(ncol(x)-2)).
compute vg3 = varb(ncol(x),ncol(x)).
compute covg1g3 = varb((ncol(x)-2), ncol(x)).
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute mdvar = x(:,(ncol(x)-(2*nx))).
end if.
compute mdmin = cmin(mdvar).
compute mdmax = cmax(mdvar).
compute fvar = x(:,(ncol(x)-2)).
compute nval = ncol(design(mdvar)).
compute fvmin = cmin(fvar).

```

```

compute fvmax = cmax(fvar).
do if (mmodval = 999).
compute mnmd = csum(mdvar)/n.
compute tmp = make(n,1,mnmd).
compute sdmd = sqrt(csum(((mdvar-
tmp)**2))/(n-1)).
compute probeval = {mnmd-sdmd; mnmd;
mnmd+sdmd}.
do if (probeval(1,1) < mdmin).
compute lowwarn = 1.
compute probeval(1,1)=mdmin.
end if.
do if (probeval(nrow(probeval),1) > mdmax).
compute highwarn = 1.
compute probeval(nrow(probeval),1)=mdmax.
end if.
do if (quantile = 1).
compute tmp = mdvar.
compute tmp(GRADE(mdvar(:,1)),:)= mdvar.
compute mdvar = tmp.
compute
probeval={mdvar(trunc(0.1*n),1);mdvar(trunc(0
.25*n),1);mdvar(trunc(0.5*n),1);mdvar(trunc(0.7
5*n),1);mdvar(trunc(0.9*n),1)}.
end if.
end if.
do if (nval = 2).
compute probeval = make(2,1,0).
compute probeval(1,1)=cmin(mdvar).
loop i = 1 to n.
do if (mdvar(i,1) <> probeval(1,1)).
compute probeval(2,1) = mdvar(i,1).
BREAK.
end if.
end loop.
end if.
do if (mmodval <> 999).
compute probeval = mmodval.
end if.
compute outp = make(nrow(probeval),7,0).

```

```

do if (mcfoc > 0 or mcmmod > 0).
compute focvals=make(1,ncol(x)+1,1).
do if (mcfoc > 0).
print/title = 'Conditional effect of X on Y at
values of the moderator:'.
print/title = ' '/space=0.
compute rnn2=mdtr.
compute matt=make(nx,6,0).
end if.
loop jj=1 to nrow(probeval).
do if (mcfoc > 0).
loop ii=1 to nx.
compute g1=b((ncol(x)-(2*nx)+ii),1).
compute g3=b((ncol(x)-nx+ii),1).
compute vg1=varb((ncol(x)-(2*nx)+ii),(ncol(x)-
(2*nx)+ii)).
compute vg3=varb((ncol(x)-nx+ii),(ncol(x)-
nx+ii)).
compute covg1g3=varb((ncol(x)-
(2*nx)+ii),(ncol(x)-nx+ii)).
compute x2 = probeval(jj,1).
compute w1 = g1+g3*x2.
compute varw1 =
vg1+(2*x2*covg1g3)+((x2*x2)*vg3).
compute sew1 = sqrt(varw1).
compute t1 = w1/sew1.
do if (ovals <> 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-tcdf(abs(t1), dfres)).
compute cnms = {'Coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)= {w1,sew1,t1,p,llci,ulci}.
else if (ovals = 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-cdfnorm(abs(t1))).
compute cnms = {'Coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)= {w1,sew1,t1,p,llci,ulci}.

```

```

end if.
end loop.
compute rnms=t(xname).
compute mdvalpr=probeval(jj,1).
print mdvalpr/title = 'Moderator
value: '/rnames=rnn2/format= F10.4 /space=0.
print matt/title='
'/cnames=cnms/rnames=rnms/format= F10.4
/space=0.
compute xprob=x(:,(ncol(x)-(2*nx)))-mdvalpr.
loop kk = 1 to nx.
compute xprob={xprob,
(xprob(:,1)&*x(:,(ncol(x)-(2*nx)+kk)))}.
end loop.
compute xprob={x(:,1:((ncol(x)-(2*nx))-
1)),xprob}.
do if (ovals <> 2).
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute residc = y-(xprob*bmultc).
compute ssresidc=csum(residc&*residc).
compute r2c = r2-(1-(ssresidc/sstotal)).
compute fcha2=(dfres*r2c)/(nx*(1-r2)).
do if (hc3 = 1).
loop kk = 1 to nx.
compute xprob={xprob, x(:,(ncol(x)-
(2*nx)+kk))}.
end loop.
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute k3 = nrow(bmultc).
compute xhc=xprob.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)=
xhc(i3,:)*inv(t(xprob)*xprob)*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).

```

```

end loop.
compute
varbc=(inv(t(xprob)*xprob)*t(xhc)*xhc*inv(t(xprob)*xprob)).
compute lmat=make((nrow(bmultc)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha2 =
(t(t(lmat)*bmultc)*inv(t(lmat)*varbc*lmat)*((t(lmat)*bmultc)))/nx).
end if.
compute pvalr2cc=1-fcdf(fcha2,nx,dfres).
compute rcha2 = {r2c, fcha2, nx,
dfres,pvalr2cc}.
print rcha2/title = 'Test of equality of conditional
means at this value of the moderator'/clabels
'R2-chng' 'F' 'df1' 'df2' 'p'/format F10.4.
else if (ovals = 2).
compute btmp=b.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=xprob.
compute bt1 = make(ncol(xlp),1,0).

.
compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.

```

```

end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL4.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(lldiff,nx).
compute rcha2 = {LLdiff, nx,pchi}.
print rcha2/title = 'Test of equality of log odds
conditioned on this moderator value'/clabels
'Chi-sq' 'df' 'p'/format F10.4.
end if.
end if.
compute
tttt=make(nrow(dummat),1,probeval(jj,1)).
compute tttt={tttt,dummat(:,2:ncol(dummat))}.
loop kkk=1 to nx.
compute tttt={tttt,tttt(:,1)*tttt(:,1+kkk)}.
end loop.
compute ones=make(nrow(dummat),1,1).
do if (ncol(x) > (2*nx+2)).

```



```

compute
covnmnmat=make(nrow(tttt),ncol(covmns),0).
loop kkk=1 to nrow(tttt).
compute covnmnmat(kkk,:)=covmns.
end loop.
compute tttt={covnmnmat,tttt}.
else.
compute tttt={ones,tttt}.
end if.
compute focvals={focvals;dummat(:,1),tttt}.
do if (mcfoc > 0).
compute yhat={dummat(:,1),(tttt*b)}.
compute cmnms={fciv, 'yhat'}.
do if (ovals <> 2).
print yhat/title = 'Estimated conditional means
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (ovals = 2).
print yhat/title = 'Estimated conditional log odds
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (jj <> nrow(probeval)).
print/title='-----'/space=0.
end if.
end if.
end loop.
compute focvals=focvals(2:nrow(focvals),:).
compute yhat=focvals(:,2:ncol(focvals))*b.
compute
focvals={focvals(:,1),focvals(:,(ncol(focvals)-
(2*nx))),yhat}.
compute cnms={fciv,mdtr,'yhat'}.
do if (mcmod > 0).
compute cnms={mdtr,fciv,'yhat'}.
end if.
end if.
do if (mcmod > 0).
compute outp=make((nx+1),7,0).

```

```

compute bcatm={b((ncol(x)-
(2*nx)),1);b((ncol(x)-nx+1):ncol(x),1)}.
compute outp(:,1)=dummat(:,1).
compute bcatcov=varb((ncol(x)-
nx):ncol(x),(ncol(x)-nx):ncol(x)).
compute bcatcov(1,1)=varb((ncol(x)-
(2*nx)),(ncol(x)-(2*nx))).
compute
bcatcov(2:nrow(bcatcov),1)=varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx)).
compute
bcatcov(1,2:nrow(bcatcov))=t(varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx))).
loop i = 1 to nrow(dummat).
compute
catmval={1,dummat(i,2:ncol(dummat))}.
compute condeff=catmval*bcatm.
compute
condse=sqrt(catmval*bcatcov*t(catmval)).
compute outp(i,2:3)={condeff,condse}.
end loop.
compute outp(:,4)=outp(:,2)&/outp(:,3).
compute outp(:,5) = 2*(1-tcdf(abs(outp(:,4)),
dfres)).
compute outp(:,6) = (outp(:,2)-
sqrt(tval)&*outp(:,3)).
compute outp(:,7) =
(outp(:,2)+sqrt(tval)&*outp(:,3)).
do if (ovals <> 2).
compute cnmms = {xname2(1,1), 'coeff', 'se',
't', 'p', 'LLCI', 'ULCI'}.
end if.
do if (ovals = 2).
compute outp(:,5) = 2*(1-
cdfnorm(abs(outp(:,4)))).
compute cnmms = {xname2(1,1), 'coeff', 'se',
'Z', 'p', 'LLCI', 'ULCI'}.
end if.

```

```

print outp/title = 'Conditional Effect of Focal
Predictor in Groups Defined by the Moderator
Variable:.'/cnames=cnmms/format = F10.4.
end if.
do if (nval > 2 and (mmodval = 999) and
mcmmod = 0).
do if (quantile <> 1).
print/title = 'Moderator values are the sample
mean and plus/minus one SD from mean'.
else.
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
end if.
do if (highwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD above the mean was'.
print/title = 'replaced with the maximum
because one SD above the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
do if (lowwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD below the mean was'.
print/title = 'replaced with the minimum
because one SD below the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
end if.
do if (nval = 2 and (mmodval = 999)).
print/title = 'The moderator variable is
dichotomous'.
end if.
do if (mmodval <> 999 and mcmmod = 0 and
((mmodval < mdmin) or (mmodval > mdmax))).
print/title = 'Warning: Moderator value specified
is outside of the range of the data'.
end if.

```

```

do if (plot = 1).
do if (mcfoc = 0 and mcmmod=0).
compute cnms = {t(nms((ncol(x)-2):(ncol(x)-
1),1)), 'yhat'}.
end if.
do if (ovals = 2).
compute prob =
exp(focvals(:,3))/(1+exp(focvals(:,3))).
compute focvals = {focvals, prob}.
compute cnms = {cnms, 'prob'}.
end if.
print/title =
*****
*****!

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (ovals = 2).
do if (mcfoc=0).
print/title = 'DATA LIST FREE/Eco_Innovation
Pro_Env_Strgy logodds prob.' /space=1.
end if.
do if (mcmmod=0).
print/title = 'DATA LIST FREE/Pro_Env_Strgy
Eco_Innovation logodds prob.' /space=1.
end if.

```

```

else if (ovals <> 2).
do if (mcfoc=0).
print/title = 'DATA LIST FREE/Eco_Innovation
Pro_Env_Strgy Operational_Performance.'
/space=1.
end if.
do if (mcmod=0).
print/title = 'DATA LIST FREE/Pro_Env_Strgy
Eco_Innovation Operational_Performance.'
/space=1.
end if.
end if.
print/title = 'BEGIN DATA.' /space=0.
print focvals/title = ' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (ovals <> 2 and mcmod=0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Operational_Performance BY
Pro_Env_Strgy.' /space=0.
end if.
do if (ovals <> 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH Operational_Performance BY
Eco_Innovation.' /space=0.
end if.
do if (ovals = 2 and mcmod=0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Pro_Env_Strgy.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Pro_Env_Strgy.' /space=0.
end if.
do if (ovals = 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH logodds BY Eco_Innovation.' /space=0.

```

```

print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH prob BY Eco_Innovation.' /space=0.
end if.
do if (ncovs > 0).
print/title = 'NOTE: For data above, covariates
are set to their sample means.'.
end if.
end if.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
end if.
END IF.
end if.
do if (bad > 0).
compute note(notes,1) = 9.
compute notes = notes + 1.
end if.
print/title = '***** ANALYSIS
NOTES AND WARNINGS
*****'.
loop i = 1 to errs.
do if (runerrs(i,1) = 1).
print/title = 'ERROR: One of your declared
mediators is dichotomous. This procedure
can"t be used.'.
end if.
do if (runerrs(i,1) = 2).
print/title = 'ERROR: For model 6, this
procedure limits the number of mediators to
four.'.
end if.
do if (runerrs(i,1) = 3).

```

```
print/title = 'ERROR: For models 1, 2, and 3,
only a single variable can be listed in the M
list.'.
end if.
do if (runerrs(i,1) = 4).
print/title = 'ERROR: You requested a model
involving W but did not provide a valid W
variable name.'.
end if.
do if (runerrs(i,1) = 5).
print/title = 'ERROR: You requested a model
involving Z but did not provide a valid Z
variable name.'.
end if.
do if (runerrs(i,1) = 6).
print/title = 'ERROR: You requested a model
involving Q but did not provide a valid Q
variable name.'.
end if.
do if (runerrs(i,1) = 7).
print/title = 'ERROR: You requested a model
involving V but did not provide a valid V
variable name.'.
end if.
do if (runerrs(i,1) = 8).
print/title = 'ERROR: You specified a W
variable for a model that does not need it'.
end if.
do if (runerrs(i,1) = 9).
print/title = 'ERROR: You specified a Z variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 10).
print/title = 'ERROR: You specified a Q variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 11).
print/title = 'ERROR: You specified a V variable
for a model that does not need it.'.
end if.
```

```

do if (runerrs(i,1) = 12).
print/title = 'ERROR: The variable specified for
W has already been assigned.'.
end if.
do if (runerrs(i,1) = 13).
print/title = 'ERROR: The variable specified for
Z has already been assigned.'.
end if.
do if (runerrs(i,1) = 14).
print/title = 'ERROR: The variable specified for
Q has already been assigned.'.
end if.
do if (runerrs(i,1) = 15).
print/title = 'ERROR: The variable specified for
V has already been assigned.'.
end if.
do if (runerrs(i,1) = 16).
print/title = 'ERROR: You did not provide a
valid Y variable name.'.
end if.
do if (runerrs(i,1) = 17).
print/title = 'ERROR: The variable specified for
Y has already been assigned.'.
end if.
do if (runerrs(i,1) = 18).
print/title = 'ERROR: Model 6 requires more
than one mediator.'.
end if.
do if (runerrs(i,1) = 19).
print/title = 'ERROR: You have not specified a
valid model number.'.
end if.
do if (runerrs(i,1) = 20).
print/title = 'ERROR: At least one and only one
variable must be listed for X.'.
end if.
do if (runerrs(i,1) = 21).
print/title = 'ERROR: At least one and only one
variable must be listed for Y.'.
end if.

```



```
do if (runerrs(i,1) = 22).
print/title = 'ERROR: Iteration didn"t converge
to a solution. Interpret results with caution.'.
end if.
do if (runerrs(i,1) = 23).
print/title = 'ERROR: You specified a clustering
variable that does not exist in your variable
list.'.
end if.
do if (runerrs(i,1) = 24).
print/title = 'ERROR: You specified a clustering
variable that has already been assigned.'.
end if.
do if (runerrs(i,1) = 25).
print/title = 'ERROR: One or more of your M
variables is not listed in the variables list.'.
end if.
do if (runerrs(i,1) = 26).
print/title = 'ERROR: A maximum of 20 cluster
units is allowed. Use multilevel modeling
instead.'.
end if.
do if (runerrs(i,1) = 27).
print/title = 'ERROR: One of the variables in
your model is a constant.'.
end if.
do if (runerrs(i,1) = 28).
print/title = 'ERROR: Dichotomous Y is not
permitted with WS option.'.
end if.
do if (runerrs(i,1) = 29).
print/title = 'ERROR: Insufficient number of
variables in vars= list when using WS option.'.
end if.
do if (runerrs(i,1) = 30).
print/title = 'ERROR: Too many variables in
vars= list when using WS option. Covariates
not allowed.'.
end if.
do if (runerrs(i,1) = 31).
```

```

print/title = 'ERROR: mmodval and wmodval
can"t both be set to zero with contrast option.'.
end if.
do if (runerrs(i,1) = 32).
print/title = 'ERROR: You did not provide a
valid X variable name.'.
end if.
do if (runerrs(i,1) = 33).
print/title = 'ERROR: PROCESS requires all
variable names to be eight characters or
fewer.'.
print/title = '      Please shorten variable names
and reexecute.'/space=0.
end if.
do if (runerrs(i,1)=34).
print/title = 'ERROR: X and M cannot both be
specified as multicategorical.'.
end if.
do if (runerrs(i,1)=35).
print/title = 'ERROR: Multicategorical variable
specification available only for models 1 and
4.'.
end if.
do if (runerrs(i,1)=36).
print/title = 'ERROR: Categorical variables
cannot have more than 10 categories.'.
end if.
do if (runerrs(i,1)=37).
print/title = 'ERROR: Each group must have at
least two cases.'.
end if.
do if (runerrs(i,1)=38).
print/title = 'ERROR: Cluster option not
available with multicategorical variables.'.
end if.
do if (runerrs(i,1)=39).
print/title = 'ERROR: A maximum of 10
variables is allowed in the M= list.'.
end if.
do if (runerrs(i,1) = 41).

```

```

print/title = 'ERROR: MCM option not permitted
in model 4.'.
end if.
end loop.
do if (errs = 0).
do if (boot > 1 or mc > 0).
do if (bconoff = 1 and boot > 1).
print boot/title = 'Number of bootstrap samples
for bias corrected bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (bconoff = 0 and boot > 1).
print boot/title = 'Number of bootstrap samples
for percentile bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (mc > 1).
print mc/title = 'Number of samples for Monte
Carlo confidence intervals:'/format = F8.0.
end if.
do if (booterr = 1).
compute badend = badend(1,2:ncol(badend)).
print badend/title = 'WARNING: Bootstrap CI
endpoints below not trustworthy. Decrease
confidence or increase bootstraps'/format =
F10.4.
end if.
end if.
print conf/title = 'Level of confidence for all
confidence intervals in output:'/format = F8.2.
do if ((center = 1 or ws=1) and (ncol(centvar) >
1)).
compute centvar = centvar(1,2:ncol(centvar)).
print centvar/title = 'NOTE: The following
variables were mean centered prior to
analysis:'/format = a8.
end if.
loop i = 1 to notes.
do if (note(i,1) = 1).

```

```
print/title = 'NOTE: Confidence level restricted
to between 50 and 99.9999%. 95% confidence
is provided in output'.
end if.
do if (note(i,1) = 2).
print/title = 'NOTE: Effect size measures for
indirect effects are not available for models
with dichotomous outcomes'.
end if.
do if (note(i,1) = 3).
print/title = 'NOTE: All standard errors for
continuous outcome models are based on the
HC3 estimator'.
end if.
do if (note(i,1) = 6).
print/title = 'NOTE: The number of bootstrap
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 7).
print/title = 'NOTE: The Johnson-Neyman
method is available only for Models 1 and 3'.
end if.
do if (note(i,1) = 8).
print/title = 'NOTE: The Johnson-Neyman
method cannot be used with a dichotomous
moderator'.
end if.
do if (note(i,1)=9).
print bad/title = 'NOTE: Some bootstrap
samples had to be replaced. The number of
such replacements was:'.
end if.
do if (note(i,1) = 11).
print nmiss/title = 'NOTE: Some cases were
deleted due to missing data. The number of
such cases was:'.
end if.
do if (note(i,1) = 12).
```

```
print /title = 'NOTE: Monte Carlo method
available only for models 4 and 5.
Bootstrapping was used instead.'.
end if.
do if (note(i,1) = 13).
print/title = 'NOTE: The number of Monte Carlo
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 19).
print/title = 'NOTE: Effect sizes not available for
within-subject analyses.'.
end if.
do if (note(i,1) = 16).
print/title = 'NOTE: Normal theory tests not
available for within-subject analyses.'.
end if.
do if (note(i,1) = 17).
print/title = 'NOTE: Monte Carlo confidence
intervals not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 18 and warnrep=0).
print/title = 'WARNING: You have requested
OLS estimation with a dichotomous criterion.'.
print/title = 'Interpret model coefficients and
inferential statistics with caution.'/space=0.
compute warnrep=1.
end if.
do if (note(i,1) = 20).
print/title = 'NOTE: Saving of bootstrap
estimates not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 22).
print/title = 'NOTE: Effect size option with
covariates requires covariates in models of M
and Y.'.
end if.
do if (note(i,1) = 23).
```

```

print/title = 'NOTE: Johnson-Neyman method
not available with multicategorical X'.
end if.
do if (note(i,1) = 24).
print/title = 'NOTE: MMODVAL option not
available with a multicategorical moderator'.
end if.
do if (note(i,1) = 25).
print/title = 'NOTE: Omnibus test for relative
indirect effects is not available with the MC
option.'.
end if.
do if (note(i,1) = 26).
print/title = 'NOTE: CONTRAST option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 27).
print/title = 'NOTE: EFFSIZE option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 28).
print/title = 'NOTE: NORMAL option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 31).
print/title = 'NOTE: The TOTAL option is not
available when Y is dichotomous.'.
end if.
do if (note(i,1) = 32).
print/title = 'NOTE: Kappa-squared is disabled
from output as of version 2.16.'.
end if.
do if (note(i,1) = 33).
print/title = 'NOTE: COVMY option ignored in
models 1, 2, and 3.'.
end if.
end loop.
end if.
end matrix.

```

Elapsed Time

00:00:04.58

[DataSet1] C:\Users\Rizwan\Desktop\SPSS Data\Data-sav (CSR & Marketing).sav

Run MATRIX procedure:

***** PROCESS Procedure for SPSS Release 2.16.3 *****

Written by Andrew F. Hayes, Ph.D. www.afhayes.com
Documentation available in Hayes (2013). www.guilford.com/p/hayes3

Model = 4
Y = Operatio
X = Pro_Env_
M = Eco_Inno

Sample size
856

Outcome: Eco_Inno

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|-----------|--------|----------|-------|
| .7582 | .5748 | .4904 | 1154.6120 | 1.0000 | 854.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|-------|-------|---------|-------|-------|--------|
| constant | .9317 | .0866 | 10.7643 | .0000 | .7618 | 1.1016 |
| Pro_Env_ | .7270 | .0214 | 33.9796 | .0000 | .6850 | .7690 |

Outcome: Operatio

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|-----------|--------|----------|-------|
| .9214 | .8490 | .1812 | 2397.4380 | 2.0000 | 853.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|-------|-------|---------|-------|--------|-------|
| constant | .0739 | .0561 | 1.3171 | .1882 | -.0362 | .1839 |
| Eco_Inno | .2538 | .0208 | 12.2015 | .0000 | .2130 | .2946 |
| Pro_Env_ | .7020 | .0199 | 35.1938 | .0000 | .6628 | .7411 |

***** DIRECT AND INDIRECT EFFECTS *****

Direct effect of X on Y

| Effect | SE | t | p | LLCI | ULCI |
|--------|-------|---------|-------|-------|-------|
| .7020 | .0199 | 35.1938 | .0000 | .6628 | .7411 |

Indirect effect of X on Y

| | Effect | Boot SE | BootLLCI | BootULCI |
|----------|--------|---------|----------|----------|
| Eco_Inno | .1845 | .0148 | .1556 | .2135 |

Normal theory tests for indirect effect

| Effect | se | Z | p |
|--------|-------|---------|-------|
| .1845 | .0161 | 11.4792 | .0000 |

***** ANALYSIS NOTES AND WARNINGS *****

Number of bootstrap samples for bias corrected bootstrap confidence intervals:
5000

Level of confidence for all confidence intervals in output:
95.00

----- END MATRIX -----

restore.

Matrix

| Notes | | |
|----------------|--------------------------------|----------|
| Output Created | 20-OCT-2020 15:57:38 | |
| Comments | | |
| Input | Active Dataset | DataSet1 |
| | Filter | <none> |
| | Weight | <none> |
| | Split File | <none> |
| | N of Rows in Working Data File | 856 |

Syntax

```
matrix.  
get dat/file = */variables =  
Operational_Performance Plan_Orga_Pract  
Eco_Innovation /names = vnames/missing =  
9999.  
compute ninit = nrow(dat).  
get dat/file = */variables =  
Operational_Performance Plan_Orga_Pract  
Eco_Innovation /names = vnames/missing =  
omit.  
get tmp/file = */variables =  
Operational_Performance /names =  
yname/missing = omit.  
get tmp2/file = */variables = Plan_Orga_Pract  
/names = xname/missing = omit.  
get tmp/file = */variables = Eco_Innovation  
/names = mnames/missing = omit.  
get tmp/file = */variables = w999999t z999999t  
v999999t q999999t.  
compute wname=tmp(1,1).  
do if (wname = ' ').  
compute wname = 'xxx'.  
end if.  
compute zname=tmp(1,2).  
do if (zname = ' ').  
compute zname = 'xxx'.  
end if.  
compute vname=tmp(1,3).  
do if (vname = ' ').  
compute vname = 'xxx'.  
end if.  
compute qname=tmp(1,4).  
do if (qname = ' ').  
compute qname = 'xxx'.  
end if.  
compute n = nrow(dat).  
compute p0=-.322232431088.  
compute p1 = -1.  
compute p2 = -.342242088547.  
compute p3 = -.0204231210245.
```

```

compute p4 = -.0000453642210148.
compute q0 = .0993484626060.
compute q1 = .588581570495.
compute q2 = .531103462366.
compute q3 = .103537752850.
compute q4 = .0038560700634.
compute badend = 0.
compute priorlo = -9999999.
compute priorhi = 9999999.
compute criterr = 0.
compute cluster = 0.
compute clsdmy = 0.
compute jndich = 0.
compute wvdich=0.
compute zqdich=0.
compute mod74dic=0.
compute warnrep=0.
compute contrast = trunc( 0 ).
do if (contrast > 2 or contrast < 0).
compute contrast = 0.
end if.
compute booterr = 0.
compute effsize = ( 0 = 1).
compute note = make(10,1,0).
compute notes = 1.
compute iterr = 0.
compute clsmtn = 0.
compute quantile = ( 0 = 1).
compute mmodval= 999.
compute jn = ( 0 = 1).
compute covcoeff=( 0 =1).
compute center = ( 0 = 1).
compute detail = ( 1 = 1).
compute coeffci = ( 1 = 1).
compute conf = 95.
compute longname = ( 1 =1).
compute bconoff=( 0 <> 1).
compute covmy = trunc( 0 ).
do if (covmy < 0 or covmy > 2).
compute covmy = 0.

```

```

end if.
do if (trunc(conf) >= 100 or (trunc(conf) <=
50)).
compute conf = 95.
compute note(notes,1) = 1.
compute notes = notes + 1.
end if.
do if (n < ninit).
compute nmiss = ninit-n.
compute note(notes,1) = 11.
compute notes = notes + 1.
end if.
compute errs = 0.
compute quantd = {0,0,0,0,0,0}.
compute quantc = {0,0,0,0,0,0}.
compute mcheck = 0.
compute ttt = 0.
compute plot = trunc( 0 ).
compute runerrs = make(50,1,0).
compute model = trunc( 4 ).
compute normal = 1.
compute ws= ( 0 =1).
compute olsdichm= ( 0 =1).
compute olsdichy= ( 0 =1).
compute mcx=trunc( 0 ).
compute mcm=trunc( 0 ).
do if (ws = 1).
do if (effsize = 1).
compute note(notes,1) = 19.
compute notes = notes + 1.
compute effsize=0.
end if.
do if (normal=1).
compute note(notes,1) = 16.
compute notes = notes + 1.
compute normal=0.
end if.
end if.
do if (model <> 4).
compute ws = 0.

```

```

end if.
do if (model > 0 and model < 4 and covmy <>
0).
compute covmy=0.
compute note(notes,1) = 33.
compute notes = notes + 1.
end if.
print/space=0.
do if (jn = 1 and model <> 1 and model <> 3).
compute note(notes,1) = 7.
compute notes = notes + 1.
end if.
do if ((model > 76) or (model < 1)).
compute model = 77.
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 19.
end if.
compute itprob=0.
do if (mcx < 0).
compute mcx=0.
end if.
do if (mcx > 4).
compute mcx=0.
end if.
do if (mcm < 0).
compute mcm=0.
end if.
do if (mcm > 4).
compute mcm=0.
end if.
do if (mcx > 0 and mcm > 0).
compute errs=errs+1.
compute runerrs(errs,1)=34.
compute criterr=1.
end if.
do if ((model <> 1 and model <> 4) and (mcx >
0 or mcm > 0)).
compute errs=errs+1.
compute runerrs(errs,1)=35.

```

```
compute criterr=1.  
end if.  
do if ((model = 4) and (mcm > 0)).  
compute errs=errs+1.  
compute runerrs(errs,1)=41.  
compute criterr=1.  
end if.  
do if ((mcx > 0) and jn <> 0).  
compute jn=0.  
compute note(notes,1) = 23.  
compute notes = notes + 1.  
end if.  
do if (mmodval <> 999 and mcm > 0).  
compute note(notes,1) = 24.  
compute notes = notes + 1.  
end if.  
compute toteff = 0.  
compute toteff = (( 0 = 1)*( 4 = 4 or 4 = 6)).  
compute varorder = 2.  
compute hc3 = ( 0 <> 0).  
compute cname = 'xxx'.  
compute centvar = {'xxx'}.  
compute nmeds = ncol(mnames).  
do if (longname=0).
```

compute criterr = 1.

compute errs = errs+1.

compute runerrs(errs,1) = 33.

1

2

3

end if.

compute modelm =

```
{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,2;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,3;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,5;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,6;  
1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,7;  
1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,8;  
1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,9;  
1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,3,10;  
1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,5,11;  
1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,0,5,12;  
1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5,13;  
0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,2,14;  
0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,2,15;  
0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,4,16;  
0,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,4,17;
```


0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,6,18;
0,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,6,19;
0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,6,20;
1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,7,21;
1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,7,22;
1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,23;
1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,0,24; 1,1,1,1

,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,25;
1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,0,0,26;
1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,0,0,27;
1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,7,28;
1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,0,7,29;
1,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,30;
1,1,0,1,0,0,1,1,0,1,0,0,0,0,0,0,0,31;
1,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,0,32;
1,1,1,1,0,0,1,1,1,1,0,0,0,0,0,0,0,33;
1,1,1,1,0,0,1,0,0,1,0,0,0,0,0,0,0,34;
1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,35;
1,0,0,1,1,0,0,0,0,0,1,1,0,0,0,0,0,0,36;
1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,37;
1,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,38;
1,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,39;
1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,0,40;
1,0,0,1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,41;
1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,42;
1,0,0,1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,43;
1,0,0,1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,44;
1,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,45;
1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,46;
1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,47;
1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,48;
1,1,0,1,1,0,1,1,0,0,0,

0,0,0,0,0,0,0,0,49;
1,1,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,50;
1,1,1,1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,51;
1,1,1,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,52;
1,1,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,53;
1,1,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,54;

1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,55;
1,1,1,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,56;
1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,57;
1,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,58;
1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,59;
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,60;
1,1,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,61;
1,1,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,62;
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,0,0,0,63;
1,0,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,64;
1,0,0,1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,65;
1,0,0,1,0,0,0,0,0,0,1,0,0,1,0,0,0,0,0,66;
1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,67;
1,1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,68;
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,69;
1,0,0,1,0,0,0,0,0,0,0,0,0,1,1,0,0,0,0,70;
1,0,0,1,0,0,1,0,0,1,0,0,1,1,1,0,0,0,71;
1,1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,1,1,0,72;
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,1,1,0,0

,73; 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,74;
1,1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,75;
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,1,0,0,76;
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,77}.

compute wm = modelm(model, 1).
compute zm = modelm(model, 2).
compute wzm = modelm(model, 3).
compute vy = modelm(model, 4).
compute qy = modelm(model, 5).
compute vqy = modelm(model, 6).
compute wy = modelm(model, 7).
compute zy = modelm(model, 8).
compute wzy = modelm(model, 9).
compute vxy = modelm(model, 10).
compute qxy = modelm(model, 11).
compute vqxy = modelm(model, 12).
compute wmy = modelm(model, 13).
compute wvmy = modelm(model, 14).
compute wvxy = modelm(model, 15).
compute zmy = modelm(model, 16).

```

compute wzmy = modelm(model,17).
compute xmy = modelm(model,18).
compute imm=modelm(model,19).
do if (model = 74).
compute vdich=0.
end if.
do if (ncol(xname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 20.
compute criterr = 1.
end if.
do if (ncol(mnames) > 10).
compute errs = errs+1.
compute runerrs(errs,1) = 39.
compute criterr = 1.
end if.
do if (ncol(yname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 21.
compute criterr = 1.
end if.
do if (( 999 =0) and ( 999 =0) and (model=3 or
model=2) and (contrast=1)).
compute errs = errs+1.
compute runerrs(errs,1) = 31.
compute criterr = 1.
end if.
compute xlist = (wm or zm or wzm or wy or zy
or wzy or vxy or qxy or vqxy or wvxy or xmy).
compute mlist = (vy or qy or vqy or zmy or wmy
or wzmy or xmy or (model < 4)).
compute bad=0.
do if (criterr = 0).
compute werr = 0.
compute verr = 0.
compute qerr = 0.
compute zerr = 0.
compute yerr = 1.
compute xerr = 1.

```

```

compute wlist = (wm or wzm or wy or wzy or
wm or wvmy or wvmy or wvxy or wzmy).
do if (wlist = 1 and wname = 'xxx').
compute werr = 1.
compute wlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (wlist = 1 and ((wname = qname) or
(wname = vname) or (wname = zname) or
(wname = xname) or (wname = yname))).
compute werr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 12).
end if.
compute zlist = (zm or wzm or zy or wzy or
zmy or wzmy).
do if (zlist = 1 and zname = 'xxx').
compute zerr = 1.
compute zlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (zlist = 1 and ((zname = qname) or
(zname = vname) or (zname = wname) or
(zname = xname) or (zname = yname))).
compute zerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 13.
end if.
compute qlist = (qy or vqy or qxy or vqxy).
do if (qlist = 1 and qname = 'xxx').
compute qerr = 1.
compute qlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (qlist = 1 and ((qname = zname) or
(qname = vname) or (qname = wname) or
(qname = xname) or (qname = yname))).

```

```

compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 14.
end if.
compute vlist = (vy or vqy or vxy or vqxy or
wvmy or wvxy).
do if (vlist = 1 and vname = 'xxx').
compute verr = 1.
compute vlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (vlist = 1 and ((vname = zname) or
(vname = qname) or (vname = wname) or
(vname = xname) or (vname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 15.
end if.
do if (wlist = 0 and wname <> 'xxx').
compute werr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 8.
end if.
do if (zlist = 0 and zname <> 'xxx').
compute zerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 9.
end if.
do if (qlist = 0 and qname <> 'xxx').
compute qerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 10.
end if.
do if (vlist = 0 and vname <> 'xxx').
compute verr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 11.
end if.
do if (hc3 = 1).

```

```

compute note(notes,1) = 3.
compute notes = notes+1.
end if.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute cons = make(n,1,1).
compute temp = (n*sscp(dat))-
(t(csum(dat))*(csum(dat))).
compute temp = temp/(n*(n-1)).
compute temp = csum(((diag(temp))=0)).
compute temp2=1.
do if (ws = 1).
compute temp2 = (n*sscp(tmp2))-
(t(csum(tmp2))*(csum(tmp2))).
compute temp2 = temp2/(n*(n-1)).
end if.
do if ((temp > 0 and ws = 0) or (temp > 0 and
ws = 1 and temp2 > 0)).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 27.
end if.
compute nmeds = ncol(mnames).
compute mcmats=ident(nmeds*2).
compute mccoef=make(nmeds*2,1,0).
compute sobel = make(nmeds,4,-999).
do if (model = 6 and nmeds > 4).
compute errs = errs+1.
compute runerrs(errs,1)=2.
end if.
do if (model < 4 and nmeds > 1).
compute errs = errs+1.
compute runerrs(errs,1) = 3.
end if.
compute nmods = (model = 74).
compute bad = 0.
compute intcnt = 1.

```

```

compute modvals = 0.
compute modvalsd = 0.
compute yintemp = {'int_1', 'int_2', 'int_3',
'int_4', 'int_5', 'int_6', 'int_7', 'int_8', 'int_9',
'int_10', 'int_11', 'int_12', 'int_13', 'int_14',
'int_15'}.
compute yintemp = {yintemp, 'int_16', 'int_17',
'int_18', 'int_19', 'int_20', 'int_21', 'int_22',
'int_23', 'int_24', 'int_25', 'int_26', 'int_27',
'int_28'}.
compute yintemp = {yintemp, 'int_29', 'int_30',
'int_31', 'int_32', 'int_33', 'int_34', 'int_35',
'int_36', 'int_37', 'int_38', 'int_39', 'int_40'}.
compute yintemp = {yintemp, 'int_41', 'int_42',
'int_43', 'int_44', 'int_45', 'int_46', 'int_47',
'int_48', 'int_49', 'int_50', 'int_51', 'int_52'}.
compute yintemp = {yintemp, 'int_53', 'int_54',
'int_55', 'int_56', 'int_57', 'int_58', 'int_59',
'int_60', 'int_61', 'int_62', 'int_63', 'int_64'}.
compute
cntname={'(C1)';'(C2)';'(C3)';'(C4)';'(C5)';'(C6)';(
C7)';'(C8)';'(C9)';'(C10)';'(C11)';'(C12)';'(C13)';(
C14)';'(C15)';'(C16)';'(C17)'}.
compute
cntname={cntname;'(C18)';'(C19)';'(C20)';'(C21
)';'(C22)';'(C23)';'(C24)';'(C25)';'(C26)';'(C27)';(
C28)';'(C29)';'(C30)';'(C31)'}.
compute
cntname={cntname;'(C32)';'(C33)';'(C34)';'(C35
)';'(C36)';'(C37)';'(C38)';'(C39)';'(C40)';'(C41)';(
C42)';'(C43)';'(C44)';'(C45)'}.
compute
cntname={cntname;'(C46)';'(C47)';'(C48)';'(C49
)';'(C50)';'(C51)';'(C52)';'(C53)';'(C54)';'(C55)';(
C56)';'(C57)';'(C58)';'(C59)'}.
compute
cntname={cntname;'(C60)';'(C61)';'(C62)';'(C63
)';'(C64)';'(C65)';'(C66)';'(C67)';'(C68)';'(C69)';(
C70)';'(C71)';'(C72)';'(C73)'}.

```

```

compute
cntname={cntname;'(C74)';'(C75)';'(C76)';'(C77
)';'(C78)';'(C79)';'(C80)';'(C81)';'(C82)';'(C83)';'(
C84)';'(C85)';'(C86)';'(C87)}.
compute
cntname={cntname;'(C88)';'(C89)';'(C90)';'(C91
)';'(C92)';'(C93)';'(C94)';'(C95)';'(C96)';'(C97)';'(
C98)';'(C99)';'(C100)';'(C101)}.
compute
cntname={cntname;'(C102)';'(C103)';'(C104)';'(
C105)}.
compute apathnam={'a path'; 'a1 path'; 'a2
path'; 'a3 path'; 'a4 path'; 'a5 path'; 'a6 path';
'a7 path'; 'a8 path'; 'a9 path'; 'a10 path'}.
compute bpathnam={'b path'; 'b1 path'; 'b2
path'; 'b3 path'; 'b4path'; 'b5path'; 'b6 path'; 'b7
path'; 'b8 path'; 'b9 path'; 'b10 path'}.
compute modvnm = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute modvnm2 = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute mlab = { ' M1 ='; ' M2 ='; ' M3 ='; '
M4 ='; ' M5 ='; ' M6 ='; ' M7 ='; ' M8 ='; '
M9 ='; ' M10 ='}.
compute m = make(n,nmeds,1).
compute mmat = make(16,nmeds,0).
compute ymat = make(8,nmeds,0).
compute deco = make(10,1,0).
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
compute iterate = abs(trunc( 10000 )).
compute converge = abs( 0.00000001 ).
compute boot = abs(trunc( 5000 )).
do if (ws=1 and 0 > 0).
compute note(notes,1)=17.
compute notes=notes+1.
end if.
compute mc=(abs(trunc( 0 ))*(1-ws)).

```



```

do if (mc > 0 and model > 5).
do if (boot = 0).
compute boot=mc.
end if.
compute mc=0.
compute note(notes,1)=12.
compute notes = notes+1.
end if.
compute adjust = 0.
do if (mc > 0 and (model > 3 and model < 6)).
compute boot=0.
compute bconoff=0.
end if.
do if (boot > 0 and mc > 0).
compute mc = 0.
end if.
compute savboot = 0.
compute saveboot = ( 0 = 1).
do if (ws=1 and saveboot=1).
compute saveboot=0.
compute note(notes,1) = 20.
compute notes = notes + 1.
compute normal=0.
end if.
do if (saveboot =1 and boot > 0 and model >
3).
compute savboot = 1.
end if.
do if (boot <> 0 or mc <> 0).
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
loop.
compute cilow = rnd(bootsz*(1-(conf/100))/2).
compute cihigh =
trunc((bootsz*(conf/100)+(bootsz*(1-
(conf/100))/2)))+1.
do if (cilow < 1 or cihigh > bootsz).

```

```

compute
bootz=trunc((bootz+1000)/1000)*1000.
compute adjust = 1.
end if.
end loop if (cilow gt 0 and cihigh le bootz).
compute boot=bootz.
do if (mc > 0).
compute mc = bootz.
end if.
do if (boot > 0 and mc > 0 and (model > 3 and
model < 6)).
compute boot=0.
end if.
do if (adjust = 1 and boot > 0).
compute note(notes,1)=6.
compute notes = notes+1.
end if.
do if (adjust = 1 and mc > 0).
compute note(notes,1)=13.
compute notes = notes+1.
end if.
end if.
do if (model = 6 and nmeds > 1).
compute mmpaths =
make((nmeds+2),(nmeds+2),0).
do if (nmeds = 2).
compute indboot = make(boot+1, 3, 999).
else if (nmeds = 3).
compute indboot = make(boot+1, 7, 999).
else if (nmeds = 4).
compute indboot = make(boot+1, 15, 999).
end if.
compute indlbl = {'Total:','Ind1 :','Ind2 :','Ind3
:','Ind4 :','Ind5 :','Ind6 :','Ind7 :','Ind8 :','Ind9
:','Ind10:','Ind11:','Ind12:','Ind13:','Ind14:','
Ind15:'}.
compute indlbl2 = {'Ind1','Ind2','Ind3','Ind4';
'Ind5','Ind6','Ind7','Ind8','Ind9','Ind10';
'Ind11','Ind12','Ind13','Ind14','Ind15'}.
compute indces = make(boot+1, 4, 999).

```

```

end if.
do if (model < 4).
compute boot = 0.
compute cmat = make(10,1,0).
compute zmat = make(10,1,0).
end if.
compute nvarch = make(1,ncol(dat),0).
compute wmatch = 0.
compute zmatch = 0.
compute vmatch = 0.
compute qmatch = 0.
compute mmatch=0.
compute minprobe=0.
compute maxprobe=0.
loop i = 1 to ncol(vnames).
do if (vnames(:,i)=yname).
compute y = dat(:,i).
compute nvarch(1,i)=1.
compute yerr = 0.
do if ((yname = xname) or (yname = wname) or
(yname = zname) or (yname = vname) or
(yname = qname)).
compute errs = errs+1.
compute runerrs(errs,1)=17.
end if.
end if.
do if (vnames(:,i)=xname).
compute x = dat(:,i).
compute nvarch(1,i)=1.
compute xdich = 1.
compute xerr=0.
loop jj = 1 to n.
do if ((x(jj,1) <> cmin(x)) and (x(jj,1) <>
cmax(x))).
compute xdich = 0.
break.
end if.
end loop.
compute xmean = csum(x)/n.

```

```

do if (center = 1 and (model < 4 or xlist > 0)
and mcx = 0).
compute meanvec = make(n,1, xmean).
compute x = x-meanvec.
compute centvar = {centvar, xname}.
end if.
compute xmean = csum(x)/n.
compute tmp = x-(cons*xmean).
compute xsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
do if (xdich = 0).
compute quantc(1,6) = 1.
compute matx = {xmean-xsd; xmean;
xmean+xsd}.
do if ( 999 = 999 and quantile = 0).
do if ((matx(1,1) < cmin(x)) and model = 74).
compute matx(1,1)=cmin(x).
compute minprobe=1.
end if.
do if ((matx(3,1) > cmax(x)) and model = 74).
compute matx(3,1)=cmax(x).
compute maxprobe=1.
end if.
end if.
do if (quantile = 1).
compute quantd(1,6) = 1.
compute quantc(1,6) = 0.
compute tmp = x.
compute tmp(GRADE(x)) = x.
compute matx =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (xdich = 1).
compute matx = {cmin(x); cmax(x)}.
do if (model=74).
compute matx=cmin(x).
compute mod74dic=1.
end if.

```

```

end if.
do if ( 999 <> 999).
compute matx = 999.
compute quantd(1,6) = 0.
compute quantc(1,6) = 0.
end if.
end if.
do if (werr = 0 and wlist = 1).
do if (vnames(:,i)=wname).
compute werr = 0.
compute wmatch = 1.
compute w = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(w)/n).
compute w = w-meanvec.
compute centvar = {centvar, wname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute wmean = csum(w)/n.
compute tmp = w-(cons*wmean).
compute wsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute wdich = 1.
loop jj = 1 to n.
do if ((w(jj,1) <> cmin(w)) and (w(jj,1) <>
cmax(w))).
compute wdich = 0.
break.
end if.
end loop.
do if (model = 3).
compute jndich=wdich.
compute jnmin=cmin(w).
compute jnmax=cmax(w).
end if.
do if (wdich = 0).
compute matw = {wmean-wsd; wmean;
wmean+wsd}.
do if ( 999 = 999 and quantile = 0).
do if (matw(1,1) < cmin(w)).

```

```

compute matw(1,1)=cmin(w).
compute minprobe=1.
end if.
do if (matw(3,1) > cmax(w)).
compute matw(3,1)=cmax(w).
compute maxprobe=1.
end if.
end if.
compute quantc(1,1) = 1.
do if (quantile = 1).
compute quantd(1,1) = 1.
compute quantc(1,1) = 0.
compute tmp = w.
compute tmp(GRADE(w)) = w.
compute matw =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (wdich = 1).
compute matw = {cmin(w); cmax(w)}.
compute wvdich=1.
compute cmaxw=cmax(w).
compute cminw=cmin(w).
end if.
do if ( 999 <> 999).
compute matw = 999.
compute quantd(1,1) = 0.
compute quantc(1,1) = 0.
end if.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm(1,1)=wname.
compute modmatp(1,1) = 1.
end if.
end if.
do if (zerr = 0 and zlist = 1).
do if (vnames(:,i)=zname).
compute zerr = 0.

```

```

compute zmatch = 1.
compute z = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(z)/n).
compute z = z-meanvec.
compute centvar = {centvar, zname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute zmean = csum(z)/n.
compute tmp = z-(cons*zmean).
compute zsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute zdich = 1.
loop jj = 1 to n.
do if ((z(jj,1) <> cmin(z)) and (z(jj,1) <>
cmax(z))).
compute zdich = 0.
break.
end if.
end loop.
do if (zdich = 0).
compute matz = {zmean-zsd; zmean;
zmean+zsd}.
do if ( 999 = 999 and quantile = 0).
do if (matz(1,1) < cmin(z)).
compute matz(1,1)=cmin(z).
compute minprobe=1.
end if.
do if (matz(3,1) > cmax(z)).
compute matz(3,1)=cmax(z).
compute maxprobe=1.
end if.
end if.
compute quantc(1,2) = 1.
do if (quantile = 1).
compute quantd(1,2) = 1.
compute quantc(1,2) = 0.
compute tmp = z.
compute tmp(GRADE(z)) = z.

```

```

compute matz = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (zdich = 1).
compute matz = {cmin(z); cmax(z)}.
compute cmaxz=cmax(z).
compute cminz=cmin(z).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matz = 999.
compute quantd(1,2) = 0.
compute quantc(1,2) = 0.
end if.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm(1,2)=zname.
compute modmatp(1,2) = 1.
end if.
end if.
do if (verr = 0 and vlist = 1).
do if (vnames(:,i)=vname).
compute verr = 0.
compute vmatch = 1.
compute v = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(v)/n).
compute v = v-meanvec.
compute centvar = {centvar, vname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute vmean = csum(v)/n.
compute tmp = v-(cons*vmean).
compute vsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute vdich = 1.
loop jj = 1 to n.

```



```

do if ((v(jj,1) <> cmin(v)) and (v(jj,1) <>
cmax(v))).
compute vdich = 0.
break.
end if.
end loop.
do if (vdich = 0).
compute matv = {vmean-vsd; vmean;
vmean+vsd}.
do if ( 999 = 999 and quantile = 0).
do if (matv(1,1) < cmin(v)).
compute matv(1,1)=cmin(v).
compute minprobe=1.
end if.
do if (matv(3,1) > cmax(v)).
compute matv(3,1)=cmax(v).
compute maxprobe=1.
end if.
end if.
compute quantc(1,3) = 1.
do if (quantile = 1).
compute quantd(1,3) = 1.
compute quantc(1,3) = 0.
compute tmp = v.
compute tmp(GRADE(v)) = v.
compute matv = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (vdich = 1).
compute matv = {cmin(v); cmax(v)}.
compute wvdich=1.
compute cmaxv=cmax(v).
compute cminv=cmin(v).
end if.
do if ( 999 <> 999).
compute matv = 999.
compute quantd(1,3) = 0.
compute quantc(1,3) = 0.

```

```

end if.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm(1,3)=vname.
compute modmatp(1,3) = 1.
end if.
end if.
do if (qerr = 0 and qlist = 1).
do if (vnames(:,i)=qname).
compute qerr = 0.
compute qmatch = 1.
compute q = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(q)/n).
compute q = q-meanvec.
compute centvar = {centvar, qname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute qmean = csum(q)/n.
compute tmp = q-(cons*qmean).
compute qsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute qdich = 1.
loop jj = 1 to n.
do if ((q(jj,1) <> cmin(q)) and (q(jj,1) <>
cmax(q))).
compute qdich = 0.
break.
end if.
end loop.
do if (qdich = 0).
compute matq = {qmean-qsd; qmean;
qmean+qsd}.
do if ( 999 = 999 and quantile = 0).
do if (matq(1,1) < cmin(q)).
compute matq(1,1)=cmin(q).
compute minprobe=1.
end if.
do if (matq(3,1) > cmax(q)).
compute matq(3,1)=cmax(q).

```

```

compute maxprobe=1.
end if.
end if.
compute quantc(1,4) = 1.
do if (quantile = 1).
compute quantd(1,4) = 1.
compute quantc(1,4) = 0.
compute tmp = q.
compute tmp(GRADE(q)) = q.
compute matq = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (qdich = 1).
compute matq = {cmin(q); cmax(q)}.
compute cmaxq=cmax(q).
compute cminq=cmin(q).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matq = 999.
compute quantd(1,4) = 0.
compute quantc(1,4) = 0.
end if.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm(1,4)=qname.
compute modmatp(1,4) = 1.
end if.
end if.
do if (vnames(:,i)=cname).
compute cld = dat(:,i).
compute cvname = vnames(:,i).
compute nvarch(1,i)=1.
compute clsmtch = 1.
end if.
loop j = 1 to ncol(mnames).
do if (vnames(:,i)=mnames(1,j)).
compute mmatch = mmatch + 1.

```

```

compute m(:,j)=dat(:,i).
do if (center = 1 and nvarch(1,i) = 0 and mlist >
0 and mcm = 0).
compute meanvec = make(n,1,csum(m(:,j))/n).
compute m(:,j) = m(:,j)-meanvec.
compute centvar = {centvar, mnames(1,j)}.
end if.
compute nvarch(1,i)=1.
compute dichm = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute dichm = 0.
break.
end if.
end loop.
do if ((dichm = 1) and (olsdichm = 1)).
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichm = 1 and model > 3 and mcheck = 0
and oldsdichm = 0).
compute errs= errs+1.
compute runerrs(errs,1) = 1.
compute mcheck = 1.
end if.
do if ((model <= 3) and (ncol(mnames)=1)).
compute nmods = nmods + 1.
compute mmean = csum(m(:,j))/n.
compute tmp = m(:,j)-(cons*mmean).
compute msd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute mdich = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute mdich = 0.
break.
end if.
end loop.
do if (model = 1).

```

```

compute jndich=mdich.
compute jnmin=cmin(m(:,j)).
compute jnmax=cmax(m(:,j)).
end if.
do if (mdich = 0).
compute matm = {mmean-msd; mmean;
mmean+msd}.
do if ( 999 = 999 and quantile = 0).
do if (matm(1,1) < cmin(m(:,j))).
compute matm(1,1)=cmin(m(:,j)).
compute minprobe=1.
end if.
do if (matm(3,1) > cmax(m(:,j))).
compute matm(3,1)=cmax(m(:,j)).
compute maxprobe=1.
end if.
end if.
compute quantc(1,5) = 1.
do if (quantile = 1).
compute quantd(1,5) = 1.
compute quantc(1,5) = 0.
compute tmp = m(:,j).
compute tmp(GRADE(m(:,j))) = m(:,j).
compute matm = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (mdich = 1).
compute matm = {cmin(m); cmax(m)}.
end if.
do if ( 999 <> 999).
compute matm = 999.
compute quantd(1,5) = 0.
compute quantc(1,5) = 0.
end if.
compute modmatv(1,5)=nrow(matm).
compute modmat((1:nrow(matm)), 5) = matm.
compute modvnm(1,5)=mnames(1,j).
compute modmatp(1,5) = 1.

```

```

end if.
end if.
end loop.
end loop.
do if (minprobe=1).
compute note(notes,1) = 14.
compute notes = notes + 1.
end if.
do if (maxprobe=1).
compute note(notes,1) = 15.
compute notes = notes + 1.
end if.
do if (cname <> 'xxx' and clsmatch = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 23.
end if.
do if (cname <> 'xxx').
do if ((cname = zname) or (cname = vname)
or (cname = wname) or (cname = xname) or
(cname = yname) or (cname = qname)).
compute errs = errs+1.
compute runerrs(errs,1) = 24.
end if.
end if.
do if (wlist = 1 and werr = 0 and wmatch = 0).
compute werr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (zlist = 1 and zerr = 0 and zmatch = 0).
compute zerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (qlist = 1 and qerr = 0 and qmatch = 0).
compute qerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (vlist = 1 and verr = 0 and vmatch = 0).

```

```

compute verr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (yerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 16.
end if.
do if (xerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 32.
end if.
do if (model = 6 and nmeds < 2).
compute errs = errs+1.
compute runerrs(errs,1) = 18.
end if.
do if (mmatch < ncol(mnames)).
compute errs=errs+1.
compute runerrs(errs,1) = 25.
end if.
do if (cname <> 'xxx').
compute cld = design(cld).
compute cluster = ncol(cld).
compute cld = cld(:,2:ncol(cld)).
compute clsdmy = ncol(cld).
do if (clsdmy > 19).
compute errs = errs+1.
compute runerrs(errs,1) = 26.
end if.
do if (mcx > 0 or mcm > 0).
compute errs = errs+1.
compute runerrs(errs,1) = 38.
end if.
end if.
compute dichy = 1.
loop jj = 1 to n.
do if ((y(jj,1) <> cmin(y)) and (y(jj,1) <>
cmax(y))).
compute dichy = 0.
break.

```

```

end if.
end loop.
do if ((dichy = 1) and (olsdichy = 1)).
compute dichy=0.
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichy = 1).
do if (toteff=1).
compute toteff=0.
compute note(notes,1) = 31.
compute notes = notes + 1.
end if.
compute jncrit=xp2*xp2.
do if (ws = 1).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 28.
end if.
end if.
compute ncovs = ncol(dat)-rsum(nvarch).
do if (effsize = 1 and covmy <> 0 and ncovs >
0 and model > 3 and model < 7).
compute note(notes,1) = 22.
compute notes = notes + 1.
compute effsize = 0.
end if.
do if (ws = 1 and ncovs < nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 29.
end if.
do if (ws = 1 and ncovs > nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 30.
end if.
end if.
do if (errs = 0).
do if (rsum(quantd) > 0).

```



```

compute note(notes,1) = 4.
compute notes = notes+1.
end if.
do if (rsum(quantc) > 0).
compute note(notes,1) = 5.
compute notes = notes+1.
end if.
do if (ncovs > 0).
compute c = make(n,ncovs,0).
compute cnames = {'x'}.
compute j = 1.
loop i = 1 to ncol(vnames).
do if (nvarch(1,i)) = 0.
compute c(:,j) = dat(:,i).
compute nvarch(1,i)=1.
compute j=j+1.
compute cnames = {cnames, vnames(:,i)}.
end if.
end loop.
compute cnames = cnames(1,2:ncol(cnames)).
do if (ws = 1).
compute covmean=csum(c)/n.
loop i = 1 to ncovs.
compute meanvec = make(n,1, covmean(1,i)).
compute c(:,i) = c(:,i)-meanvec.
end loop.
compute centvar = {centvar, cnames}.
compute x=c(:,1:ncovs).
do if (ncovs=nmads).
compute ncovs=0.
end if.
end if.
end if.
compute names = {yname, xname, mnames,
wname, zname, vname, qname}.
do if (ncovs > 0).
compute names = {names, cnames}.
end if.
do if (dichy = 1 and effsize = 1).
compute note(notes,1) = 2.

```

```

compute notes = notes+1.
end if.
do if (model > 3 and model < 6).
compute indeff=make(nmeds,1,0).
compute indboot=make(boot+1,nmeds,999).
do if (mc > 0).
compute indboot=make(mc+1,nmeds,999).
end if.
do if (effsize = 1 and dichy = 0).
compute rmeff=make(boot+1,nmeds+1,999).
compute abpseff=make(boot+1,nmeds+1,999).
compute abcseff=make(boot+1,nmeds+1,999).
compute pmeff=make(boot+1,nmeds+1,999).
compute r245 = make(boot+1,1,999).
compute kappa2 = make(boot+1,1,999).
end if.
end if.
do if (model = 6 and effsize = 1 and dichy = 0).
compute
rmeff=make(boot+1,ncol(indboot),999).
compute
abpseff=make(boot+1,ncol(indboot),999).
compute
abcseff=make(boot+1,ncol(indboot),999).
compute
pmeff=make(boot+1,ncol(indboot),999).
end if.
do if (nmods > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm(1,tmp) = modvnm(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:nmods).
compute modvnm=modvnm(:,1:nmods).
compute modmatv=modmatv(:,1:nmods).

```

```

loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvals =
make((modmatv(1,1)*modprod(1,1)),
nmods,0).
loop i = 1 to nmods.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvals)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvals(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
do if (model = 74).
compute modvals=matx.
compute modvnm=xname.
end if.
compute vmat = make(8,nrow(modvals),0).
compute vmat(1,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute vmat(5,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute indeff=make(nrow(modvals),1,0).
do if (model <> 5).
compute indboot = make(((boot+1)*nmeds),
nrow(modvals), -99999999).
compute indbootp =
make(boot+1,(nmeds*nmods), -99999999).
do if (imm > 4 and imm < 8).
do if (imm = 7).

```

```

compute immmv1=matw.
compute immmv2=matv.
end if.
do if (imm = 5).
compute immmv1=matw.
compute immmv2=matz.
end if.
do if (imm = 6).
compute immmv1=matv.
compute immmv2=matq.
end if.
compute
indboop1=make((boot+1),(nmeds*(nrow(immmv2))),-9999999).
compute
indboop2=make((boot+1),(nmeds*(nrow(immmv1))),-9999999).
compute onesmv1=make(1,nrow(immmv1),1).
compute onesmv2=make(1,nrow(immmv2),1).
compute
nimvs=nrow(immmv1)+nrow(immmv2).
compute cmmval=0.
compute cmmlbs1={'xx'}.
compute cmmlbs2={'xx'}.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv2}.
compute
cmmlbt=make(nrow(immmv2),1,mnames(1,i)).
compute cmmlbs1={cmmlbs1;cmmlbt}.
end loop.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv1}.
compute
cmmlbt=make(nrow(immmv1),1,mnames(1,i)).
compute cmmlbs2={cmmlbs2;cmmlbt}.
end loop.
compute cmmval=cmmval(2:nrow(cmmval),:).
end if.
end if.
end if.

```

```

do if (nmods> 0).
loop i = 1 to ncol(modvals).
do if (modvnm(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm(1,i)=qname).
compute qcol = i.
end if.
end loop.
end if.
do if (dichy = 1).
compute omx = cmax(y).
compute omn = cmin(y).
compute y = (y = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute data = {cons,y,m,x}.
compute datamed = data.
compute datayed = data.
compute datanm = {'constant'; yname;
t(mnames); xname}.
compute datanmm = {'constant'; yname;
t(mnames); xname}.
compute datanmy = {'constant'; yname;
t(mnames); xname}.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
do if (model < 4 and errs = 0).
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' ', ' '}.
compute datayed = {datayed, x&*m}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).

```

```

compute vmat(1,i) = 1.
compute vmat(2,i) = modvals(i,1).
end loop.
compute mmat = make(16,nmeds,1).
end if.
do if (model = 2 or model = 3).
compute int1 = x&*w.
compute datayed = {datayed, w, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,2).
compute vmat(3,i) = modvals(i,1).
compute vmat(4,i) = modvals(i,1)*modvals(i,2).
end loop.
end if.
do if (model = 3).
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*m.
compute int2 = x&*w&*m.
compute datayed = {datayed, int1, int2}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' X', wname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (model = 4 or model = 5).
compute vmat = make(8,1,1).
end if.
compute yintkey2 = yintkey.
do if (wm = 1).
compute int1 = x&*w.

```

```

compute datamed = {datamed, w,int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmm = {datanmm; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,wcol).
end loop.
do if (zm = 1).
compute int1 = x&*z.
compute datamed = {datamed, z, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(3,i) = modvals(i,zcol).
end loop.
end if.
do if (wzm = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*z.
compute int2 = x&*w&*z.
compute datamed = {datamed, int1, int2}.
loop i = 1 to nrow(modvals).
compute vmat(4,i) =
modvals(i,wcol)*modvals(i,zcol).
end loop.
end if.

```

```

end if.
compute mdatacol=ncol(datamed).
compute mintkey = yintkey.
compute yintkey = {' ',' ',' ',' ',' ',' '}.
compute medints = intcnt-1.
do if (vy = 1 or xmy = 1).
compute mp = 1.
do if (xmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,1).
end loop.
end if.
do if (vy = 1).
compute datayed = {datayed, v}.
compute datanmy = {datanmy; vname}.
compute mmods = 1.
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,vcol).
end loop.
do if (qy = 1).
compute mp = 2.
compute datayed = {datayed,q}.
compute datanmy = {datanmy; qname}.
compute mmods = 2.
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,qcol).
end loop.
end if.
do if (vqy = 1).
compute mp = 3.
compute datayed = {datayed, v&*q}.
compute mmods = 3.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,vcol)*modvals(i,qcol).
end loop.
end if.
end if.
compute mints = make(n,(nmeds*mp),0).
loop i = 0 to (nmeds-1).

```



```

do if (i = 0 and vqy = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
vname, ' X', qname, '', ''}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, '', ''}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (xmy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))&*x.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', xname, '', ''}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (qy = 1).
compute mints(:,((i*mp)+2))=m(:,(i+1))&*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', qname, '', ''}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqy = 1).
compute mints(:,((i*mp)+3))=m(:,(i+1))&*v&*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end loop.

```

```

compute datayed = {datayed, mints}.
end if.
compute mp = 1.
do if (wvmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,wcol)*modvals(i,vcol).
end loop.
end if.
compute mints2 = make(n,(nmeds*mp),0).
do if (wmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,wcol).
end loop.
do if (wy = 0 and model > 3).
compute datayed = {datayed, w}.
compute datanmy = {datanmy; wname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wvmy = 1).
compute datayed = {datayed,w&*v}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints2(:,((i*mp)+1))= m(:,(i+1))&*w.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wvmy = 1).
compute mints2(:,((i*mp)+2))=
m(:,(i+1))&*w&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', vname}.

```

```

compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints2}.
end if.
compute mp = 1.
do if (zmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,zcol).
end loop.
do if (wzmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,zcol)&*modvals(i,wcol).
end loop.
end if.
end if.
do if (zmy = 1).
compute mints3 = make(n,(nmeds*mp),0).
do if (zy = 0).
compute datayed = {datayed, z}.
compute datanmy = {datanmy; zname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wzmy = 1 and wzy = 0).
compute datayed = {datayed,w&*z}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints3(:,((i*mp)+1))= m(:,(i+1))&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.

```

```

compute intcnt = intcnt+1.
do if (wzmy = 1).
compute mints3(:,(i*mp)+2))=
m(:,(i+1))*w*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', zname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints3}.
end if.
compute decoc = 1.
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
do if (wy = 1 and model > 3).
compute datayed = {datayed, w, x*w}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm2(1,1)=wname.
compute modmatp(1,1) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (zy = 1).
compute datayed = {datayed,z,x*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm2(1,2)=zname.
compute modmatp(1,2) = 1.

```

```

compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ',', ' '}.
compute datanmy = {datanmy; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (wzy = 1).
compute datayed = {datayed,w&*z,x&*w&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ',', ' '}.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vxy = 1).
compute datayed = {datayed, x&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm2(1,3)=vname.
compute modmatp(1,3) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, ',', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (qxy = 1).
compute datayed = {datayed, x&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.

```

```

compute modvnm2(1,4)=qname.
compute modmatp(1,4) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqxy = 1).
compute datayed = {datayed, x&*v&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, 'X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end if.
do if (wvxy = 1).
compute datayed = {datayed, x&*w&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, 'X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute modvalsd = 0.
compute ttt = rsum(modmatp).
do if (rsum(modmatp) > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm2(1,tmp) = modvnm2(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.

```

```

end loop.
compute modmat=modmat(:,1:ttt).
compute modvnm2=modvnm2(:,1:ttt).
compute modmatv=modmatv(:,1:ttt).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvalsd =
make((modmatv(1,1)*modprod(1,1)), ttt,0).
loop i = 1 to ttt.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvalsd)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvalsd(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
end if.
do if (ttt > 0).
loop i = 1 to ncol(modvalsd).
do if (modvnm2(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm2(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm2(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm2(1,i)=qname).
compute qcol = i.

```

```

end if.
end loop.
compute directv = make(nrow(modvalsd),1,1).
do if (wy = 1).
compute directv ={directv, modvalsd(:,wcol)}.
end if.
do if (zy = 1).
compute directv ={directv, modvalsd(:,zcol)}.
end if.
do if (wzy = 1).
compute directv ={directv,
(modvalsd(:,wcol)&*modvalsd(:,zcol))}.
end if.
do if (vxy = 1).
compute directv ={directv, modvalsd(:,vcol)}.
end if.
do if (qxy = 1).
compute directv ={directv, modvalsd(:,qcol)}.
end if.
do if (vqxy = 1).
compute directv ={directv,
(modvalsd(:,vcol)&*modvalsd(:,qcol))}.
end if.
do if (wvxy = 1).
compute directv ={directv,
(modvalsd(:,vcol)&*modvalsd(:,wcol))}.
end if.
end if.
compute ydatacol=ncol(datayed).
do if (ncovs > 0).
do if (covmy <> 2).
compute datamed = {datamed,c}.
end if.
do if (covmy <> 1).
compute datayed = {datayed,c}.
end if.
do if (covmy = 1 and model = 4 and mcx > 0).
compute datayed = {datayed,c}.
end if.
compute covmeans = csum(c)/n.

```



```

end if.
do if (cluster > 0).
compute datamed = {datamed,cld}.
compute datayed = {datayed, cld}.
compute cldmeans = csum(cld)/n.
end if.
compute mst = 3.
compute mnd = mst+nmeds-1.
compute ydatacol=ncol(datayed).
compute mdatacol=ncol(datamed).
do if (ncovs > 0).
compute datanmy = {datanmy; t(cnames)}.
do if (model > 3).
compute datanmm = {datanmm; t(cnames)}.
end if.
end if.
compute datanmy = {'constant';
datanmy(3:nrow(datanmy),1)}.
do if (model > 3).
compute datanmm = {'constant';
datanmm(3:nrow(datanmm),1)}.
end if.
compute amm = make(2,1,0).
compute abmm = make(2,1,0).
compute mnv = csum(datayed(:,2)/n).
compute mnv = make(n,1,mnv).
compute ssty = csum((datayed(:,2)-mnv)**2).
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
do if (ws=1).
compute stddevm=diag(sigma).
compute
stddevm=sqrt(stddevm(3:(nmeds+2),1)).
end if.
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute r2xy =
(sigma(2,(3+nmeds)))/(stddevy*stddevx)**2.

```

```

compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))&**2.
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).
do if (model = 4 and nmeds = 1 and cluster = 0
and ncovs = 0).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
compute datatm = datamed.
compute dataty = datayed.
compute mdlInms2 = { '4' ; yname; xname}.
compute mdlInms = {'Model ='; ' Y ='; ' X ='
}.
do if (ws = 1).
compute mdlInms2 = { '4' ; yname}.
compute mdlInms = {'Model ='; ' Y ='}.
end if.

```

```

loop i = 1 to ncol(mnames).
compute mdlInms2 = {mdlInms2; mnames(1,i)}.
do if (i = 1 and ncol(mnames) = 1).
compute mdlInms = {mdlInms; ' M = '}.
else.
compute mdlInms = {mdlInms; mlab(i,1)}.
end if.
end loop.
do if (wname <> 'xxx').
compute mdlInms2 = {mdlInms2; wname}.
compute mdlInms = {mdlInms; ' W = '}.
end if.
do if (zname <> 'xxx').
compute mdlInms2 = {mdlInms2; zname}.
compute mdlInms = {mdlInms; ' Z = '}.
end if.
do if (vname <> 'xxx').
compute mdlInms2 = {mdlInms2; vname}.
compute mdlInms = {mdlInms; ' V = '}.
end if.
do if (qname <> 'xxx').
compute mdlInms2 = {mdlInms2; qname}.
compute mdlInms = {mdlInms; ' Q = '}.
end if.
do if (jn = 1 and model = 1 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
do if (jn = 1 and model = 3 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
compute mdpbe=0.
compute medte2=0.
do if ( 4 =4 and mcx > 0).
compute medte2=1.
end if.
do if (((mcx > 0) or (mcm > 0)) AND model =
1).
compute mdpbe=1.

```

```

compute dd={datayed(:,4),datayed(:,3)}.
do if (ncovs > 0).
compute dd={datayed(:,6:(5+ncovs)),dd}.
end if.
compute dd={datayed(:,2),dd}.
end if.
compute yes = ((nrow(modvals)=1) and
(contrast = 1) and (model = 3 or model = 2)).
compute yes2 = ((nrow(modvals)=1) and
(contrast = 1) and (model = 2)).
print/title = '***** PROCESS Procedure
for SPSS Release 2.16.3 *****'.
print/title = '      Written by Andrew F. Hayes,
Ph.D.    www.afhayes.com'.
print/title = '  Documentation available in
Hayes (2013).
www.guilford.com/p/hayes3/space=0.
print mdlnms2/title =
*****
*****'/rnames = mdlnms/format =
a8.
do if (ncovs > 0).
print cnames/title = 'Statistical Controls:'/'rlabels
= 'CONTROL='/'format a8.
end if.
print n/title = 'Sample size'/format F10.0.
do if (cluster > 0).
print cluster/rnames = cvname/title =
'Clustering Variable and Number of Clusters'.
end if.
do if (model > 3 and ( 'random' <> 'random')).
compute seedt= 'random'.
print seedt/title='Custom'/format=A12/rlabel =
'Seed:'.
end if.
do if (mdpbe=0 and medte2 = 0).
loop bt = 1 to (boot+1).
do if (bt = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)),-999).

```

```

end if.
compute bootcoef = 0.
do if (bt > 1).
loop.
compute v=trunc(uniform(n,1)*n)+1.
compute datayed = dataty(v,:).
do if (ws=1).
compute
bmeans=csum(datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed)))/n.
compute cntmat=make(n,nmeds,1).
compute bmeans2=bmeans.
do if (nmeds > 1).
compute bmeans2=mdia(bmeans).
end if.
compute cntmat=cntmat*bmeans2.
compute datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed))=datayed(:,(ncol(data
yed)-nmeds+1):ncol(datayed))-cntmat.
end if.
compute detcheck=det(t(datayed)*datayed).
compute rk=(detcheck <> 0).
do if (model > 3).
compute datamed = datatm(v,:).
compute detcheck=det(t(datamed)*datamed).
do if (rk=1).
compute rk=(detcheck <> 0).
end if.
end if.
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
compute temp=diag(sigma).
compute bad = bad+(1-rk).
compute false = 1.
end loop if (rk = 1).
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).

```

```

compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).
do if (model = 4 and nmeds = 1 and ncovs = 0
and cluster = 0).
compute r2xy =
(sigma(2,4)/(stddevy*stddevx))&**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))&**2.
compute sstot = sigma(2,2)*(n-1).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
end if.
do if (model > 3).
loop im = 1 to nmeds.
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.

```

```

compute xnm = {'constant';
datanmm((2+nmeds):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
do if (model = 6).
do if (im = 1).
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
end if.
do if (im > 1).
compute xm={cons, datamed(:,3:(im+1)),
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant'; datanmm(2:im,1);
datanmm((mnd):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
compute
mmpaths((im+1),(2:im))=t(coeff(2:im,1)).
end if.
end if.
do if (ws = 1).
compute
coeff(1,1)=csum(datamed(:,(2+im)))/n.
end if.
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
do if (bt = 1).
compute resid=datamed(:,(2+im))-xm*coeff.
compute sse =cssq(resid).
compute mse = sse/(n-ncol(xm)).
compute mnv = csum(data(:,(2+im)))/n.
compute mnv = make(n,1,mnv).
compute sstm = csum((data(:,(2+im))-
mnv)**2).

```

```

compute k3 = nrow(coeff).
do if (hc3 = 1).
compute h = xm(:,1).
loop i3=1 to n.
compute h(i3,1)= xm(i3,:)*invXtX*t(xm(i3,:)).
end loop.
loop i3=1 to k3.
compute xm(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xm(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xm(:,i3) = sqrt(mse)&*xm(:,i3).
end loop.
end if.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute hccov=invXtX*t(xm)*xm*invXtX.
compute mcmats(im,im)=hccov(2,2).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*hccov*lmat)*((t(lmat)
)*coeff))/dfnum.
compute coeff = coeff(1:(nrow(coeff)-
clsdmy),1).
compute mccoef(im,1)=coeff(2,1).
compute standerr =
sqrt(diag(invXtX*t(xm)*xm*invXtX)).
compute standerr = standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1).
compute standerr(1,1)=stddevm(im,1)/sqrt(n).
end if.
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-
ncol(xm)))).
compute temp=(n-ncol(xm)).
do if (ws=1).

```



```

compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute temp=(n-1).
end if.
compute xd = abs(xp2).
compute temp = (temp* (exp((temp-
(5/6))*((xd/(temp-(2/3)+(.11/temp)))*(xd/(temp-
(2/3)+(.11/temp)))))-1)).
compute temp1 = coeff-
sqrt(abs(temp))*standerr.
compute temp2 =
coeff+sqrt(abs(temp))*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (ws=1).
compute op=op(1,:).
compute xnmn =
apathnam(((nmeds>1)+1+(im-1)),1).
end if.
compute sobel(im,1) = coeff(2,1).
compute sobel(im,2)=standerr(2,1).
compute temp = mnames(1,im).
compute r2full = 1-(sse/sstm).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute summ = {sqrt(r2full), r2full, mse,
Fratio, dfnum, dfden, pfr}.
do if (detail = 1).
print temp/title =
*****
*****'/rlabels = 'Outcome:'/format
= A8.
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels 'R',
'R-sq', 'MSE','F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
print op/title = 'Model'/rnames = xnmn/clabels
'coeff', 'se', 't', 'p', 'LLCI', 'ULCI'/format = F10.4.
do if (covcoeff=1 and ws <> 1).

```

```

compute
hccovtmp=hccov(1:nrow(op),1:nrow(op)).
compute cnamestp=t(xmnm).
print hccovtmp/title='Covariance matrix of
regression parameter
estimates'/rnames=xmnm/cnames=cnamestp/f
ormat= F10.4.
end if.
do if (nmods > 0 and nrow(mintkey) > 1).
print mintkey/title = 'Product terms key:.'/format
= A8.
end if.
do if (ws = 1).
compute wsdf=n-1.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
end if.
end if.
compute ymat(1,im) = coeff(2,1).
do if (ws = 1).
compute ymat(1,im)=coeff(1,1).
end if.
do if (wm = 1).
compute ymat(2,im) = coeff(4,1).
do if (zm = 1).
compute ymat(3,im) = coeff(6,1).
do if (wzm = 1).
compute ymat(4, im) = coeff(8,1).
end if.
end if.
end if.
do if (model = 6).
compute mmpaths((im+1),1)=coeff((im+1),1).
end if.
end loop.
end if.
loop totlp = 1 to (1+(toteff*(bt = 1))).
do if (toteff = 1 and totlp =2).
compute xy =
{cons,datayed(:,(3+nmeds):ydatacol)}.

```

```

else.
compute xy={cons, datayed(:,3:ydatacol)}.
end if.
do if (dichy = 1).
compute pt2 =
make(nrow(datayed(:,2)),1,(csum(datayed(:,2))
/n)).
compute LL3 = datayed(:,2)*ln(pt2)+(1-
datayed(:,2))*ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xy),1,0).
compute LL1 = 0.
compute xy22=xy.
loop jjj = 1 to iterate.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.
compute coeff =
bt1+inv(t(xy22)*xy)*t(xy)*(datayed(:,2)-pt1).
compute pt1 = 1/(1+exp(-(xy*coeff))).
compute itprob = csum((pt1 <
.00000000000001) or (pt1 >
.99999999999999)).
do if (itprob = 0).
compute LL = datayed(:,2)*ln(pt1)+(1-
datayed(:,2))*ln(1-pt1).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
break.
end if.
compute bt1 = coeff.
compute LL1 = LL2.
end loop.
do if (jjj >= iterate and iterr = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
compute iterr = 1.
end if.

```

```

loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)&*pt1&*(1-pt1).
end loop.
compute covmat = inv(t(xy22)*xy).
release xy22.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
end if.
do if (dichy = 0).
compute invXtX = inv(t(xy)*xy).
compute coeff =invXtX*t(xy)*datayed(:,2).
do if (effsize=1 and model > 3 and model < 7
and (ncovs > 0 or clsdmy > 0)).
compute xysdy =
{cons,datayed(:,(4+nmeds):ydatacol)}.
compute coeffsd
=inv(t(xysdy)*xysdy)*t(xysdy)*datayed(:,2).
compute resid=datayed(:,2)-(xysdy*coeffsd).
compute sdycov =sqrt(csum(resid&*resid)/(n-
ncol(xysdy))).
compute xvaron=datayed(:,(3+nmeds)).
compute
coeffx2=inv(t(xysdy)*xysdy)*t(xysdy)*xvaron.
compute residx2=xvaron-(xysdy*coeffx2).
compute sdxcov
=sqrt(csum(residx2&*residx2)/(n-ncol(xysdy))).
end if.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4 and bt > 1).
compute resid=datayed(:,2)-xy*coeff.
compute sse = cssq(resid).
compute r2full = 1-(sse/sstot).

```

```

end if.
do if (bt = 1).
compute resid=data(:,2)-xy*coeff.
compute k3 = nrow(coeff).
compute sse = cssq(resid).
compute mse = sse/(n-ncol(xy)).
compute
hcnott=coeff&/(sqrt(diag(mse*invXtX))).
do if (hc3 = 1).
compute h = xy(:,1).
loop i3=1 to n.
compute h(i3,1)= xy(i3,:)*invXtX*t(xy(i3,:)).
end loop.
loop i3=1 to k3.
compute xy(:,i3) = (resid(:,ncol(resid))&/(1-
h))*xy(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xy(:,i3) = sqrt(mse)*xy(:,i3).
end loop.
end if.
compute covmat = (invXtX*t(xy)*xy*invXtX).
end if.
end if.
do if (bt = 1).
do if (model = 2).
compute xy2={cons, datayed(:,3:ydatacol)}.
compute temp = ncol(xy2).
do if (temp > 6).
compute xy3=xy2(:,7:temp)}.
end if.
compute xy2={xy2(:,1:3), xy2(:,5)}.
do if (temp > 6).
compute xy2={xy2, xy3}.
release xy3.
end if.
compute invXtX = inv(t(xy2)*xy2).
compute coeff2 =invXtX*t(xy2)*datayed(:,2).

```

```

compute ssem2=cssq(datayed(:,2)-xy2*coeff2).
release xy2.
end if.
compute standerr = sqrt(diag(covmat)).
do if (totlp =1).
compute
mcmats((nmeds+1):ncol(mcmats),(nmeds+1):n
col(mcmats))=covmat(2:(1+nmeds),2:(1+nmed
s)).
end if.
compute standerr=standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1 and totp=2).
compute standerr(1,1)=stddevy/sqrt(n).
end if.
compute coeffplt = coeff.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*coeff)))/dfnum).
compute coeff=coeff(1:(nrow(coeff)-clsdmy),1).
do if (totlp=1).
compute
mccoeff((nmeds+1):nrow(mccoeff))=coeff(2:(1+
nmeds),1).
end if.
compute bbbb=coeff(2,1).
do if totp = 1.
compute deco(1,1)=2+nmeds.
compute deco = deco(1:decoc,1).
compute covdirt = make((nrow(covmat)-
clsdmy),(ncol(covmat)-clsdmy),0).
compute covdirt = covmat(deco,:).
compute covdir =
make(nrow(covdirt),ncol(covdirt),0).
compute covdir = covdirt(:,t(deco)).
compute deco=coeff(deco,1).

```

```

do if (ttt > 0).
compute sedir =
sqrt(diag(directv*covdir*t(directv))).
compute directv = directv*deco.
end if.
compute sobel(:,3)=coeff(2:(1+nmeds),1).
compute sobel(:,4) = standerr(2:(1+nmeds),1).
compute sobel2 = sobel&*sobel.
do if (varorder <> 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)&*sobel2(:,4)+sobel2(:,3)&*sob
el2(:,2)).
end if.
do if (varorder = 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)&*sobel2(:,4)+sobel2(:,3)&*sob
el2(:,2)+sobel2(:,2)&*sobel2(:,4)).
end if.
compute sobel(:,1) = sobel(:,1)&*sobel(:,3).
compute sobel(:,3) = sobel(:,1)&/sobel(:,2).
compute sobel(:,4) = 2*(1-
cdfnorm(abs(sobel(:,3)))).
end if.
do if (dichy = 0).
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
do if (ws= 1 and totlp=2).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute dfden=n-1.
end if.
compute cnms = {'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute op = {coeff, standerr, tratio, p}.
end if.
do if (dichy = 1).
compute tratio = (coeff&/standerr).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute wald = tratio&*tratio.
compute cnms = {'coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.

```

```

compute temp=coeff-abs(xp2)*standerr.
compute op = {coeff, standerr, tratio, p, temp}.
compute temp=coeff+abs(xp2)*standerr.
compute op = {op, temp}.
end if.
do if (detail = 1).
do if (totlp = 2).
print yname/title = '*****'
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:.'/format = A8.
end if.
do if (totlp <> 2).
print yname/title =
'*****'
*****'/rlabels = 'Outcome:.'/format
= A8.
end if.
end if.
do if (dichy = 1 and bt = 1 and totp = 1).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:.'/cnames = nmsd/format = F9.2.
end if.
do if (dichy = 0).
compute r2full = 1-(sse/ssty).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute jndf=dfden.
compute wsdf=dfden.
compute xd = abs(xp2).
compute jncrit = (dfden* (exp((dfden-
(5/6))*((xd/(dfden-
(2/3)+(.11/dfden)))*(xd/(dfden-
(2/3)+(.11/dfden)))))-1)).
compute summ = {sqrt(r2full), r2full, mse,
fratio, dfnum, dfden, pfr}.
compute temp1=coeff-sqrt(jncrit)*standerr.
compute temp2=coeff+sqrt(jncrit)*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.

```



```

do if (detail = 1).
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels =
'R', 'R-sq', 'MSE' 'F', 'df1', 'df2', 'p'/format =
F10.4.
end if.
end if.
end if.
do if (dichy = 1).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(coeff)-
1)).
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
do if (detail = 1).
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format =
F10.4.
end if.
end if.
do if (totlp = 2).
compute datanmy={'constant';
datanmy((nmeds+2):nrow(datanmy),1)}.
end if.
do if (detail = 1).
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
do if (ws=1).
compute op2=op(1:(1+nmeds),:).
compute datanmy2 = {'c" path'; bpathnam(
(1+(nmeds>1)): ((1+(nmeds>1))+(nmeds-
1)),1)}.
do if (totlp=2).
compute datanmy2={'c path'}.
compute op2=op(1,:).

```

```

compute wsdf=n-1.
end if.
print op2/title = 'Model'/rnames =
datanmy2/cnames = cnms/format = F10.4.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
do if (ws <> 1).
print op/title = 'Model'/rnames =
datanmy/cnames = cnms/format = F10.4.
do if (covcoeff=1).
compute
covmattp=covmat(1:nrow(op),1:nrow(op)).
compute cnamestp=t(datanmy).
print covmattp/title='Covariance matrix of
regression parameter
estimates'/rnames=datanmy/cnames=cnamest
p/format= F10.4.
end if.
end if.
end if.
do if (ttt = 0 and totlp = 1).
compute deco = op((nmeds+2),:).
do if (ws = 1).
compute deco=op(1,:).
end if.
end if.
do if (ttt = 0 and totlp = 2).
compute decotot = op(2,:).
do if (ws = 1).
compute decotot=op(1,:).
end if.
end if.
do if (nmods > 0 and model > 4 and detail = 1
and nrow(yintkey) > 1)).
print yintkey/title = 'Product terms key:'/format
= A8.
end if.
do if (nmods > 0 and model < 4 and detail = 1).
print yintkey2/title = 'Product terms key:'/format
= A8.

```

```

do if ((model = 1 or model = 2) and dichy = 0).
compute temp={((op(4,3)**2)*(1-r2full))/dfden,
op(4,3)**2, 1, dfden, op(4,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(4,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(2,1).
do if (model = 2).
compute temp={temp;((op(6,3)**2)*(1-
r2full))/dfden, op(6,3)**2, 1, dfden, op(6,4)}.
do if (hc3=1).
compute temp(2,1)=((hcnot(6,1)**2)*(1-
r2full))/dfden.
end if.
compute frat2=(dfden*(r2full-(1-
(ssem2/ssty))))/(2*(1-r2full)).
do if (hc3=1).
compute lmat=make(nrow(coeffplt),2,0).
compute lmat(4,1)=1.
compute lmat(6,2)=1.
compute frat2 =
(t(t(lmat)*coeffplt)*inv(t(lmat)*covmat*lmat)*((t(l
mat)*coeffplt))/2).
end if.
compute temp={temp;(r2full-(1-
(ssem2/ssty))),frat2,2,dfden,1-
fcdf(frat2,2,dfden)}.
compute rnms={rnms; yintkey2(3,1),'Both'}.
end if.
print temp/title = 'R-square increase due to
interaction(s):'/rnames=rnms/clabels = 'R2-
chng', 'F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (model = 3 and dichy = 0).
compute temp={((op(8,3)**2)*(1-r2full))/dfden,
op(8,3)**2, dfden, op(8,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(8,1)**2)*(1-
r2full))/dfden.

```

```

end if.
compute rnms=yintkey2(5,1).
print temp/title = 'R-square increase due to
three-way interaction:.'/rnames=rnms/clabels =
'R2-chng', 'F(1,df2)', 'df2', 'p'/format = F10.4.
end if.
end if.
end if.
do if (model = 6 and totlp = 1).
compute
mmpaths(nrow(mmpaths),1)=coeff(nrow(mmpa
ths),1).
compute
mmpaths(nrow(mmpaths),(2:(nmeds+1)))=t(co
eff(2:(nmeds+1),1)).
end if.
do if (totlp = 1).
loop im = 1 to nmeds.
do if (model < 4).
compute ymat(1,im) = coeff(3,1)*(1-yes).
compute ymat(2,im) = coeff(4,1).
compute cmat(1,im) = covmat(3,3).
compute cmat(2,im) = covmat(4,4).
compute cmat(5,im) = covmat(3,4).
compute jnb1=coeff(3,1).
compute jnb3=coeff(4,1).
compute jnsb1=covmat(3,3).
compute jnsb3=covmat(4,4).
compute jnsb1b3=covmat(3,4).
do if (model = 2 or model = 3).
compute ymat(3,im) = coeff(6,1).
compute cmat(3,im) = covmat(6,6).
compute cmat(6,im) = covmat(3,6).
compute cmat(8,im) = covmat(4,6).
end if.
do if (model = 3).
compute ymat(4,im) = coeff(8,1).
compute cmat(4,im) = covmat(8,8).
compute cmat(7,im) = covmat(3,8).
compute cmat(9,im) = covmat(4,8).

```

```

compute cmat(10,im) = covmat(6,8).
compute jnb1=coeff(4,1).
compute jnb3=coeff(8,1).
compute jnsb1=covmat(4,4).
compute jnsb3=covmat(8,8).
compute jnsb1b3=covmat(4,8).
end if.
end if.
do if (model > 3).
compute ymat(5,im) = coeff((1+im),1).
end if.
do if (xmy = 1).
compute ymat(6,im) = coeff((2+nmeds+im),1).
end if.
do if (vy = 1).
compute ymat(6,im) = coeff((3+nmeds+im),1).
end if.
do if (qy = 1 and vy = 1).
compute ymat(6,im) = coeff((5+nmeds+((im-1)*2)),1).
compute ymat(7,im) = coeff((6+nmeds+((im-1)*2)),1).
end if.
do if (vqy = 1).
compute ymat(6,im) = coeff((6+nmeds+((im-1)*3)),1).
compute ymat(7,im) = coeff((7+nmeds+((im-1)*3)),1).
compute ymat(8,im) = coeff((8+nmeds+((im-1)*3)),1).
end if.
do if (wmy = 1).
compute ymat(7,im) = coeff((3+nmeds+im-wy),1).
end if.
do if (wmy = 1 and vy = 1).
compute ymat(7,im) =coeff((4+(nmeds*2)+im-wy),1).
end if.
do if (wmy = 1 and vy = 1 and wvmy = 1).

```

```

compute ymat(7,im) =
coeff((6+(nmeds*2)+((im-1)*2)-wy),1).
compute ymat(8,im) =
coeff((7+(nmeds*2)+((im-1)*2)-wy),1).
end if.
do if (wmy = 1 and zmy = 1).
compute ymat(6,im) = coeff((6-(wzy*3)+(wzm-
1)+(nmeds*2)+((im-1)*2)*wzm)+((im-1)*(1-
wzm))-((zy-wzy)*2),1).
compute ymat(7,im) = coeff((3-
zy+im+nmeds),1).
do if (wzmy = 1).
compute ymat(8,im) = coeff((7-
(wzy*3)+(nmeds*2)+((im-1)*2)),1).
end if.
end if.
do if (nmods > 0 and model <> 5).
loop indlp = 1 to nrow(modvals).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp)).
do if (model > 6).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp))*csum(ym
at(5:8,im)*vmat(5:8,indlp)).
end if.
end loop.
compute indboot((bt+(im-
1)*(boot+1)),:)=t(indeff).
do if (imm = 1 or imm = 3).
compute
indbootp(bt,im)=ymat(2,im)*ymat(5,im).
do if (imm=3).
compute
indbootp(bt,(im+nmeds))=ymat(3,im)*ymat(5,i
m).
do if (zdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxz-cminz).
end if.

```

```

end if.
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
end if.
end if.
do if ((imm= 2 or imm = 4) or model = 74).
compute
indbootp(bt,im)=ymat(1,im)*ymat(6,im).
do if (imm=4).
compute
indbootp(bt,(im+nmeds))=ymat(1,im)*ymat(7,i
m).
do if (qdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxq-cminq).
end if.
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
end if.
end if.
do if (imm=7).
compute
indbootp(bt,im)=ymat(2,im)*ymat(6,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(2,im)*y
mat(6,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(6,im)*onesmv1)+(ymat(2,im)*y
mat(6,im)*t(immmv1)).
do if (wdich=1).

```

```

compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxv-cminv).
end if.
end if.
do if (imm=5).
compute
indbootp(bt,im)=ymat(4,im)*ymat(5,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(4,im)*y
mat(5,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(3,im)*ymat(5,im)*onesmv1)+(ymat(4,im)*y
mat(5,im)*t(immmv1)).
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).

```



```

end if.
do if (zdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxz-
cminz).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxz-cminz).
end if.
end if.
do if (imm=6).
compute
indbootp(bt,im)=ymat(1,im)*ymat(8,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(1,im)*ymat(6,im)*onesmv2)+(ymat(1,im)*y
mat(8,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(7,im)*onesmv1)+(ymat(1,im)*y
mat(8,im)*t(immmv1)).
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxv-cminv).
end if.
do if (qdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxq-
cminq).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-

```

```

1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxq-cminq).
end if.
end if.
do if ((model = 58 or model = 59) and
wvdich=1).
compute
indbootp(bt,im)=(ymat(1,im)*ymat(7,im)*(cmax
w-cminw))+(ymat(2,im)*ymat(5,im)*(cmaxw-
cminw))+(ymat(2,im)*ymat(7,im)*((cmaxw*cma
xw)-(cminw*cminw))).
end if.
end if.
do if (model = 4 or model = 5).
compute
indboot(bt,im)=csum(ymat(1:4,im)&*vmat(1:4,1
))*csum(ymat(5:8,im)&*vmat(5:8,1)).
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.000000000000001.
end if.
compute sumind=rsum(indboot(bt,:)).
do if (im=nmeds).
compute
pmeff(bt,2:(im+1))=indboot(bt,:)/(sumind+coeff(
(2+nmeds),1)).
end if.
compute
rmeff(bt,(im+1))=indboot(bt,im)/coeff((2+nmeds
),1).
compute
abpseff(bt,(im+1))=indboot(bt,im)/stddevy.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*stddevx.
do if (ncovs > 0 or clsdmy > 0).
compute
abpseff(bt,(im+1))=indboot(bt,im)/sdy cov.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*sdx cov.
end if.

```

```

do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,1) = r2my-(r2full-r2xy).
compute abmmr = 1.
do if (indboot(bt,im) < 0).
compute abmmr = cmin(abmm).
end if.
do if (indboot(bt,im) > 0).
compute abmmr = cmax(abmm).
end if.
compute kappa2(bt,1)=indboot(bt,im)/abmmr.
compute tmp = indboot(bt,im)/abmmr.
end if.
end if.
end if.
do if (model = 6).
do if (nmeds = 2).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(4,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3).
compute
indboot(bt,3)=mmpaths(3,1)*mmpaths(4,3).
end if.
do if (nmeds = 3).
compute indboot(bt,1) =
mmpaths(2,1)*mmpaths(5,2).
compute indboot(bt,2) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3).
compute indboot(bt,3) =
mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4).
compute indboot(bt,4) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4).
compute indboot(bt,5) =
mmpaths(3,1)*mmpaths(5,3).
compute indboot(bt,6) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4).

```

```

compute indboot(bt,7) =
mmpaths(4,1)*mmpaths(5,4).
end if.
do if (nmeds = 4).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(6,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(6,3).
compute
indboot(bt,3)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(6,4).
compute
indboot(bt,4)=mmpaths(2,1)*mmpaths(5,2)*mmpaths(6,5).
compute
indboot(bt,5)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(6,4).
compute
indboot(bt,6)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3)*mmpaths(6,5).
compute
indboot(bt,7)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,8)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,9)=mmpaths(3,1)*mmpaths(6,3).
compute
indboot(bt,10)=mmpaths(3,1)*mmpaths(4,3)*mmpaths(6,4).
compute indboot(bt,11)
=mmpaths(3,1)*mmpaths(5,3)*mmpaths(6,5).
compute indboot(bt,12) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,13) =
mmpaths(4,1)*mmpaths(6,4).

```

```

compute indboot(bt,14) =
mmpaths(4,1)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,15) =
mmpaths(5,1)*mmpaths(6,5).
end if.
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.00000000000001.
end if.
compute
pmeff(bt,:)=indboot(bt,:)/(rsum(indboot(bt,:))+m
mpaths(nrow(mmpaths),1)).
compute rmeff(bt,:) =
indboot(bt,:)/mmpaths(nrow(mmpaths),1).
compute abpseff(bt,:)=indboot(bt,:)/stddevy.
compute
abcseff(bt,:)=(stddevx*indboot(bt,:))/stddevy.
do if (ncovs > 0 or clsdmy > 0).
compute abpseff(bt,:)=indboot(bt,:)/sdy cov.
compute
abcseff(bt,:)=(sdx cov*indboot(bt,:))/sdy cov.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,:) = r2my-(r2full-r2xy).
end if.
end if.
end if.
end loop.
end if.
end loop.
do if (savboot = 1 and bt > 1).
compute bootstrp((bt-1),:)=bootcoef.
end if.
end loop.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.

```

```

release datayed, dat, datamed, data, y, x,
datatm, dataty, mints2, cons, mnv, tmp.
do if (mc > 0).
compute x1 = sqrt(-
2*ln(uniform(mc,nrow(mcmats))))&*cos((2*3.14
159265358979)*uniform(mc,nrow(mcmats))).
compute x1=x1*chol(mcmats).
loop i=1 to nrow(x1).
compute x1(i,:)=x1(i,:)+t(mccoeff).
end loop.
loop i = 1 to nmeds.
compute
indboot(2:nrow(indboot),i)=x1(:,i)&*x1(:,(i+nme
ds)).
end loop.
end if.
do if (ttt = 0 and model > 3).
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
end if.
do if (model <> 74).
do if (dichy = 0).
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 't', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 't', 'p', 'LLCI', 'ULCI'/format =
F10.4.
else.
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 'Z', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.

```

```

print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 'Z', 'p', 'LLCI', 'ULCI' /format =
F10.4.
end if.
end if.
end if.
do if (ttt > 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
compute clbs = {modvnm2, 'Effect', 'SE', 't', 'p',
'LLCI', 'ULCI'}.
compute tratio = (directv&/sedir).
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
compute outp = {modvalsd, directv, sedir,
tratio, p}.
do if (dichy = 0).
compute temp1=directv-sqrt(jncrit)*sedir.
compute temp2=directv+sqrt(jncrit)*sedir.
compute outp = {outp, temp1, temp2}.
end if.
do if (dichy = 1).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute temp = directv-abs(xp2)*sedir.
compute outp = {outp, temp}.
compute temp = directv+abs(xp2)*sedir.
compute outp = {outp, temp}.
compute clbs = {modvnm2, 'Effect', 'SE', 'Z', 'p',
'LLCI', 'ULCI'}.
end if.
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
print outp/title = 'Conditional direct effect(s) of
X on Y at values of the moderator(s):'/format =
F10.4 /cnames = clbs.
end if.
do if (nmods > 0 and model <> 5).
do if (model < 4).

```

```

print/title =
*****
*****!

compute zmat(1,1) = 1*(1-yes).
compute cfse = make(nrow(modvals),1,0).
loop #m = 1 to nrow(modvals).
do if (model = 1).
compute zmat(2,1)=modvals(#m,1)**2.
compute zmat(5,1)=2*modvals(#m,1).
end if.
do if (model = 2 or model = 3).
compute zmat(2,1)=modvals(#m,2)**2.
compute zmat(3,1)=modvals(#m,1)**2.
compute
zmat(4,1)=(modvals(#m,1)**2)*(modvals(#m,2)
**2)*(1-yes2).
compute zmat(5,1)=2*modvals(#m,2)*(1-yes).
compute zmat(6,1)=2*modvals(#m,1)*(1-yes).
compute
zmat(7,1)=2*modvals(#m,1)*modvals(#m,2)*(1-
yes).
compute
zmat(8,1)=2*modvals(#m,1)*modvals(#m,2).
compute
zmat(9,1)=2*modvals(#m,1)*(modvals(#m,2)**
2)*(1-yes2).
compute
zmat(10,1)=2*(modvals(#m,1)**2)*modvals(#m
,2)*(1-yes2).
end if.
compute cfse(#m,1)=sqrt(csum(zmat&*cmat)).
end loop.
end if.
do if (nmods > 0).
compute clbs = {modvnm, 'Effect'}.
loop im = 1 to nmeds.
compute obs = t(indboot(1+(im-1)*(boot+1),:)).
compute outp = {modvals, obs}.
do if (model < 4).
compute tstat = obs&/cfse.

```



```

do if (dichy = 0).
compute pval = 2*(1-tcdf(abs(tstat), (n-
ncol(xy)))).
compute temp=obs-sqrt(jncrit)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp=obs+sqrt(jncrit)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 't', 'p', 'LLCI', 'ULCI'}.
compute jnclbs=clbs.
end if.
do if (dichy = 1).
compute pval = 2*(1-cdfnorm(abs(tstat))).
compute temp = obs-abs(xp2)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp = obs+abs(xp2)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute jnclbs=clbs.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indboot((1+(im-
1)*(boot+1)),:).
compute indboot2 = indboot((2+(im-
1)*(boot+1)):(1+(im-1)*(boot+1)+boot),:).
compute mnind = t(csum(indboot2)/boot).
compute stdind=t(sqrt((cssq(indboot2)-
((csum(indboot2)**2)/boot))/(boot-1))).
compute llci=make(1,ncol(indboot2),-999).
compute ulci=make(1,ncol(indboot2),-999).
loop #e = 1 to ncol(indboot2).
.
compute temp = indboot2(:,#e).
compute temp(GRADE( indboot2(:,#e) )) =
indboot2(:,#e).
compute badlo = 0.
compute badhi = 0.

```

```

do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.

```

```

do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {modvals, obs, stdind, t(llci),
t(ulci)}.
compute clbs = {modvnm, 'Effect', 'Boot SE',
'BootLLCI', 'BootULCI'}.
end if.
compute mtemp = mnames(1, im).
compute rlbs =
make(nrow(modvals),1,mnames(1,im)).
do if (model < 4).
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
do if (yes=0).
print outp/title = 'Conditional effect of X on Y at
values of the moderator(s):'/cnames =
clbs/format = F10.4.
else.
compute outp=outp(:,3:ncol(outp)).
compute clbs=clbs(:,3:ncol(clbs)).
compute clbs(1,1)='Contrast'.
print outp/title = 'Contrast of conditional effects
of X on Y'/cnames = clbs/format = F10.4.
end if.
end if.
do if (model > 5 and mod74dic <> 1).
do if (im = 1).
print/title = 'Conditional indirect effect(s) of X on
Y at values of the moderator(s):'.
end if.
print outp/title = 'Mediator'/rnames =
rlbs/cnames = clbs/format = F10.4.

```

```

else if (mod74dic = 1 and model = 74).
do if (im = 1).
print/title = 'Indirect effect(s) of X on Y:'.
compute clbs3=clbs(1,2:ncol(clbs)).
end if.
compute outp3=outp(1,2:ncol(outp)).
print outp3/title = 'Mediator'/rnames =
rlbs/cnames = clbs3/format = F10.4.
end if.
end loop.
loop i = notes to 1 by -1.
do if (note(i,1) = 4 and yes=0).
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 5 and yes=0).
print/title = 'Values for quantitative moderators
are the mean and plus/minus one SD from
mean.'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 14 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    below the mean was replaced
with the minimum because one SD below the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.
do if (note(i,1) = 15 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    above the mean was replaced
with the maximum because one SD above the
mean'/space=0.

```

```

print/title = '      is outside of the range of the
data.'/space=0.
end if.
end loop.
do if (model = 3 and yes=0).
compute jnvals=make(nrow(matw),7,0).
compute jnvals(:,1)=matw.
compute jnvals(:,2)=jnb1+jnb3*jnvals(:,1).
compute
jnvals(:,3)=sqrt(jnsb1+2*jnvals(:,1)*jnsb1b3+(jn
vals(:,1)*jnvals(:,1))*jnsb3).
compute jnvals(:,4)=jnvals(:,2)/jnvals(:,3).
do if (dichy = 0).
compute jnvals(:,5)=2*(1-tcdf(abs(jnvals(:,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(:,5)=2*(1-
cdfnorm(abs(jnvals(:,4)))).
end if.
compute jnvals(:,6)=jnvals(:,2)-
sqrt(jncrit)*jnvals(:,3).
compute
jnvals(:,7)=jnvals(:,2)+sqrt(jncrit)*jnvals(:,3).
compute clbs={clbs(:,1),clbs(:,3:ncol(clbs))}.
do if (coeffci = 0).
compute jnvals =jnvals(:,1:(ncol(jnvals)-2)).
end if.
print jnvals/title = 'Conditional effect of X*M
interaction at values of W:.'/cnames
=clbs/format = F10.4.
end if.
do if (jn = 1 and (model = 1 or model = 3) and
jndich = 0 and yes=0).
print/title = '***** JOHNSON-
NEYMAN TECHNIQUE
*****'.
compute ajn =(jncrit*jnsb3)-(jnb3*jnb3).
compute bjn = 2*((jncrit*jnsb1b3)-(jnb1*jnb3)).
compute cjn = (jncrit*jnsb1)-(jnb1*jnb1).

```

```

compute radarg = (bjn*bjn)-(4*ajn*cjn).
compute den = 2*ajn.
compute nrts = 0.
do if (radarg >= 0 and den <> 0).
compute x21 = (-bjn+sqrt(radarg))/den.
compute x22 = (-bjn-sqrt(radarg))/den.
compute roots = 0.
do if (x21 >= jnmin and x21 <= jnmax).
compute nrts = 1.
compute roots = {roots; x21}.
end if.
do if (x22 >= jnmin and x22 <= jnmax).
compute nrts = nrts + 1.
compute roots = {roots; x22}.
end if.
compute roots={roots,make(nrow(roots),2,0)}.
compute modtemp=m.
do if (model=3).
compute modtemp=w.
end if.
do if (nrts > 0).
compute roots = roots(2:nrow(roots),1:3).
compute roots(1,2)=(csum(modtemp <
roots(1,1))/n)*100.
compute roots(1,3)=(csum(modtemp >
roots(1,1))/n)*100.
do if (nrow(roots)=2).
compute roots(2,2)=(csum(modtemp <
roots(2,1))/n)*100.
compute roots(2,3)=(csum(modtemp >
roots(2,1))/n)*100.
end if.
print roots/title = 'Moderator value(s) defining
Johnson-Neyman significance
region(s)'/clabels = 'Value', '% below', '%
above'/format = F10.4.
compute jnvals=make((21+nrts),7,0).
loop i= 0 to 20.
compute jnvals((i+1),1)=jnmin+(i*((jnmax-
jnmin)/20)).

```

```

end loop.
loop i = 1 to nrts.
loop j = 2 to nrow(jnvals).
do if ((roots(i,1) > jnvals((j-1),1)) and (roots(i,1)
< jnvals(j,1))).
compute
jnvals((j+1):(21+i),1)=jnvals(j:(20+i),1).
compute jnvals(j,1)=roots(i,1).
end if.
end loop.
end loop.
loop i = 1 to nrow(jnvals).
compute jnvals(i,2)=jnb1+jnb3*jnvals(i,1).
compute
jnvals(i,3)=sqrt(jnsb1+2*jnvals(i,1)*jnsb1b3+(jn
vals(i,1)*jnvals(i,1))*jnsb3).
compute jnvals(i,4)=jnvals(i,2)/jnvals(i,3).
do if (dichy = 0).
compute jnvals(i,5)=2*(1-tcdf(abs(jnvals(i,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(i,5)=2*(1-
cdfnorm(abs(jnvals(i,4)))).
end if.
compute jnvals(i,6)=jnvals(i,2)-
sqrt(jncrit)*jnvals(i,3).
compute
jnvals(i,7)=jnvals(i,2)+sqrt(jncrit)*jnvals(i,3).
end loop.
do if (model = 1).
print jnvals/title = 'Conditional effect of X on Y
at values of the moderator (M)'/cnames
=jnclbs/format = F10.4.
end if.
do if (model = 3).
compute
jnclbs={jnclbs(:,1),jnclbs(:,3:ncol(jnclbs))}.

```

```

print jnvals/title = 'Conditional effect of X*M on
Y at values of the moderator (W)'/cnames
=jnclbs/format = F10.4.
end if.
end if.
do if (nrts = 0).
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
else.
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
end if.
end if.
do if (model < 4 and (plot = 1)).
compute dataplot =
make((nrow(modvals)*nrow(matx)),(ncol(modv
als)+1),0).
compute tmp = 1.
loop i = 1 to nrow(modvals).
loop j = 1 to nrow(matx).
compute dataplot(tmp,:)={matx(j,1),
modvals(i,:)}.
compute tmp=tmp+1.
end loop.
end loop.
compute dataplot = {dataplot,
make(nrow(dataplot),(1+dichy),0)}.
compute dataplo2 = make(nrow(dataplot),1,1).
do if (model = 1).
compute dataplo2 = {dataplo2, dataplot(:,2),
dataplot(:,1), (dataplot(:,1)*dataplot(:,2))}.
end if.
do if (model = 2 or model = 3).
compute dataplo2 = {dataplo2, dataplot(:,3),
dataplot(:,1), (dataplot(:,1)*dataplot(:,3)),
dataplot(:,2), (dataplot(:,1)*dataplot(:,2))}.

```



```

do if (model = 3).
compute dataplo2 = {dataplo2,
(dataplot(:,2)*dataplot(:,3)),
(dataplot(:,1)*dataplot(:,2)*dataplot(:,3))}.
end if.
end if.
loop i = 1 to nrow(dataplot).
compute tmp=dataplo2(i,:).
do if (ncovs > 0).
compute tmp = {tmp, covmeans}.
end if.
do if (cluster > 0).
compute tmp = {tmp, cldmeans}.
end if.
compute dataplot(i,(ncol(dataplot)-
(dichy)))=tmp*coeffplt.
do if (dichy = 1).
compute
dataplot(i,(ncol(dataplot)))=exp(tmp*coeffplt)/(1
+exp(tmp*coeffplt)).
end if.
end loop.
compute clbs = {xname, modvnm, 'yhat'}.
do if (dichy = 1).
compute clbs = {xname, modvnm, 'ln(odds)',
'prob'}.
end if.
print/title =
*****
*****!

```

```

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (model = 1).
do if (dichy = 0).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Eco_Innovation
Operational_Performance.' /space=1.
else if (dichy = 1).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Eco_Innovation
logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH Operational_Performance BY
Eco_Innovation.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Operational_Performance BY
Plan_Orga_Pract.' /space=0.
end if.
end if.

```

```

do if (dichy = 1).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH logodds BY Eco_Innovation.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH prob BY Eco_Innovation.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Plan_Orga_Pract.'
/space=0.
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Plan_Orga_Pract.' /space=0.
end if.
end if.
end if.
do if (model = 2 or model = 3).
do if (dichy = 0).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Eco_Innovation
Operational_Performance.' /space=1.
end if.
do if (dichy = 1).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Eco_Innovation
logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH Operational_Performance BY
Eco_Innovation/PANEL ROWVAR=' /space=0.

```

```

end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Operational_Performance BY
Plan_Orga_Pract/PANEL ROWVAR=.'
/space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH logodds BY Eco_Innovation/PANEL
ROWVAR=.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH prob BY Eco_Innovation/PANEL
ROWVAR=.' /space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Plan_Orga_Pract/PANEL
ROWVAR=.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Plan_Orga_Pract/PANEL
ROWVAR=.' /space=0.
end if.
end if.
end if.
do if (ncovs > 0).
print/title = '* Estimates are based on setting
covariates to their sample means.'.
end if.
end if.
end if.

```

```

do if ((imm > 0) or (model = 74 and
mod74dic=0) or ((model = 58 or model = 59)
and wvdich=1))).
do if (imm > 4 and imm < 8).
compute
indbootp={indbootp(:,1:nmeds),indboop1,indbo
op2}.
end if.
compute obsprod = t(indbootp(1,:)).
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indbootp(1,:).
compute indbootp = indbootp(2:(boot+1),:).
compute mnindp = t(csum(indbootp)/boot).
compute stdindp=t(sqrt((cssq(indbootp)-
((csum(indbootp)**2)/boot))/(boot-1))).
compute llcip=make(1,ncol(indbootp),-999).
compute ulcip=make(1,ncol(indbootp),-999).
loop #e = 1 to ncol(indbootp).

.
compute temp = indbootp(:,#e).
compute temp(GRADE( indbootp(:,#e) )) =
indbootp(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).

```

```

compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcip(1,#e)=llcit.
compute ulcip(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {obsprod, stdindp, t(llcip),
t(ulcip)}.
compute clbs = {'Effect', 'SE(Boot)', 'BootLLCI',
'BootULCI'}.
do if (model = 8 or model = 12).

```

```

print/title='-----'.
compute outpimm=outp(1:nmeds,:).
print/title = 'Indirect effect of highest order
product.'/space=0.
print outpimm/title = 'Mediator'/cnames =
clbs/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
compute clbs3 = {'Index', 'SE(Boot)',
'BootLLCI', 'BootULCI'}.
do if (imm < 5).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator.'/format=A8.
end if.
compute outpimm=outp((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
do if (imm > 4 and imm < 8).

```

```

print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'.

compute outpimm=outp(1:nmeds,:).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
compute
outpimm=outp((nmeds+1):nrow(outp),:).
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.

loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),clbs3}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3(1,1)=modvnm(1,1).
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.

```



```

print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
end if.
do if (boot = 0).
do if (model = 8 or model = 12).
print/title='-----'.
compute obsimm=obsprod(1:nmeds,:).
print/title = 'Indirect effect of highest order
product.'/space=0.
print obsimm/title = 'Mediator'/clabels =
'Effect'/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
do if ((imm > 0 and imm < 5) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=obsprod((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
end loop.
end if.

```

```

do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****!
.
compute outpimm=obsprod(1:nmeds,1).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
compute
outpimm=obsprod((nmeds+1):nrow(obsprod),:)
.
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),'Index'}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3={modvnm(1,1),'Index'}.
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator: '/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
end if.
end if.

```

```

end if.
compute conmake=0.
compute concols = 0.
do if ((model > 3 and model < 7) and (contrast
> 0) and nmods = 0 and nmeds > 1 ).
compute concols =
((ncol(indboot)*(ncol(indboot)-1))/2).
compute indcon=make(nrow(indboot),concols,-
999).
compute conkey = { ' ', ' ', ' ' }.
compute temp=1.
compute conmake=1.
loop i = 1 to (ncol(indboot)-1).
loop j = (i+1) to (ncol(indboot)).
compute indcon(:,temp)=indboot(:,i)-
indboot(:,j).
do if (contrast = 2).
compute indcon(:,temp)=abs(indboot(:,i))-
abs(indboot(:,j)).
end if.
do if (model <= 6).
compute conkey={conkey; mnames(1,i),
'minus', mnames(1,j)}.
end if.
do if (model = 6).
compute conkey={conkey; indlbl2(i,1), 'minus',
indlbl2(j,1)}.
end if.
compute temp=temp+1.
end loop.
end loop.
end if.
do if (model = 4 or model = 5).
compute clbs = {'Effect'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obs = {csum(obs); obs}.
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.

```

```

compute rlbs = {rlbs;
cntname(1:ncol(indcon),1)}.
end if.
compute outp = obs.
compute outp2=outp.
do if (effsize = 1 and dichy = 0).
compute
pmeff(:,1)=rsum(pmeff(:,2:ncol(pmeff))).
compute
rmeff(:,1)=rsum(rmeff(:,2:ncol(rmeff))).
compute
abpseff(:,1)=rsum(abpseff(:,2:ncol(abpseff))).
compute
abcseff(:,1)=rsum(abcseff(:,2:ncol(abcseff))).
compute eff = {pmeff, rmeff, abpseff, abcseff}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute eff = {eff, r245}.
compute r245obs = {r245(1,1);r245(1,1)}.
compute kappa2ob = kappa2(1,1).
end if.
compute pmobs = t(pmeff(1,1:(nmeds+1))).
compute rmobs = t(rmeff(1,1:(nmeds+1))).
compute psobs = t(abpseff(1,1:(nmeds+1))).
compute csobs = t(abcseff(1,1:(nmeds+1))).
do if (contrast = 0).
compute outp2 = {obs, psobs, csobs, pmobs,
rmobs}.
end if.
do if (contrast > 0).
compute obs2=obs(1:nrow(psobs),:).
compute outp2 = {obs2, psobs, csobs, pmobs,
rmobs}.
end if.
compute clbs = {'ab', 'ab_ps', 'ab_cs', 'ab/c',
'ab/c'}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute outp2 = {outp2, r245obs}.
compute clbs = {clbs, 'R-sq_med'}.

```

```

do if (boot = 0 and mc = 0).
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
do if (boot = 0 and mc = 0).
do if (nmeds = 1).
compute outp2 = outp2(2,:).
compute rlbs = rlbs(2,1).
end if.
do if (model=5).
compute outp2=outp2(:,1:3).
end if.
print outp2/title = 'Indirect effect(s) of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1 and nmeds >
1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0 or mc > 0).
compute indboot = {rsum(indboot), indboot}.
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
compute ones = make(bootSZ,1,1).
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(bootSZ+1),:).
compute mnind = t(csum(indboot)/bootSZ).

```

```

compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/bootSZ))/(bootSZ-1))).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).

.

compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.

```

```

end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llici(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
do if (effsize = 1 and dichy = 0 and boot > 0).
compute estmte=eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).
.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.

```

```

compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).

```



```

compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
end if.
end if.
do if (boot > 0 or mc > 0).
compute outp = {obs, stdind, t(llci), t(ulci)}.
do if (nmeds = 1).
compute outp = outp(2,:).
compute rlbs = rlbs(2,1).
end if.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
do if (mc > 0).
compute clbs = {'Effect', 'MC SE', 'MC LLCI',
'MC ULCI'}.
end if.
print outp/title = 'Indirect effect of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (dichy = 0 and effsize = 1 and boot > 0).
compute outp = {psobs,
stdindf((2*(nmeds+1)+1):(3*(nmeds+1)),1),
t(llcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))),
t(ulcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Partially standardized indirect
effect of X on Y'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute outp = {csobs,
stdindf((3*(nmeds+1)+1):(4*(nmeds+1)),1),
t(llcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))),
t(ulcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.

```

```

print outp/title = 'Completely standardized
indirect effect of X on Y'/rnames = rlbs/cnames
= clbs/format = F10.4.
compute outp = {pmobs,
stdindf(1:(nmeds+1),1),t(llcif(1,1:(nmeds+1))),
t(ulcif(1,1:(nmeds+1))) }.
do if (model = 4).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
compute outp = {rmobs, stdindf(((nmeds+1)+1)
:(2*(nmeds+1)) ,1),
t(llcif(1,((nmeds+1)+1):(2*(nmeds+1)))),
t(ulcif(1,((nmeds+1)+1):(2*(nmeds+1))))}.
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
do if (nmeds = 1 and cluster = 0 and ncovs = 0
and model = 4).
compute r245obs = r245obs(1,1).
compute outp = {r245obs,
stdindf((4*(nmeds+1)+1):(4*(nmeds+1)+1),1),
t(llcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1))),
t(ulcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1)))}.
print outp/title = 'R-squared mediation effect
size (R-sq_med)'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
end if.
do if (normal = 1).

```

```

do if (nmeds = 1).
print sobel/title = 'Normal theory tests for
indirect effect'/clabels = 'Effect', 'se', 'Z',
'p'/format = F10.4.
end if.
do if (nmeds > 1).
compute rlbs2 = rlbs(2:nrow(rlbs),1).
print sobel/title = 'Normal theory tests for
specific indirect effects'/rnames = rlbs2/clabels
= 'Effect', 'se', 'Z', 'p'/format = F10.4.
end if.
end if.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
do if (model = 6).
compute clbs = {'eff'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obsfsum=csum(obs).
compute obs = {obsfsum; obs}.
compute indlbl = indlbl(1:nrow(obs),1).
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute indlbl = {indlbl;
cntname(1:ncol(indcon),1)}.
end if.
compute obs2=obs.
do if (boot = 0).
do if (dichy = 0 and effsize = 1).
compute clbs = {'eff', 'eff_ps', 'eff_cs', 'eff/c',
'eff/c'}.
do if (ncovs = 0).

```

```

compute obs = {obs, obs/stddevy,
(obs*stddevx/stddevy), obs/ctot,
obs/mmpaths(nrow(mmpaths),1)}.
end if.
do if (ncovs > 0 or clsdmy > 0).
compute obs = {obs, obs/sdycov,
(obs*sdxcov/sdycov),
obs/(obsfsum+mmpaths(nrow(mmpaths),1)),
obs/mmpaths(nrow(mmpaths),1)}.
end if.
compute obs2=obs.
do if (contrast > 0).
compute obs2 = obs(1:(nrow(obs)-concols),:).
end if.
end if.
print obs2/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute indboot = {rsum(indboot), indboot}.
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(boot+1),:).
compute mnind = t(csum(indboot)/boot).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/boot))/(boot-1))).
compute temp = nrow(indboot).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).

```

```

loop #e = 1 to ncol(indboot).

.

compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.

```

```

do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute obs = {obs, stdind, t(llci), t(ulci)}.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
print obs/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (effsize = 1 and dichy = 0).
compute indboot=indboot(:,1:(ncol(indboot)-
concols)).
compute eff = {rsum(abpseff), abpseff,
rsum(abcseff), abcseff, rsum(pmeff), pmeff,
rsum(rmeff), rmeff}.
compute effobs = eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).
.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).

```

```

compute badlo = 0.
compute badhi = 0.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(effobs(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.

```

```

compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute temp2 = stdindf(1:ncol(indboot),1).
compute temp3 = effobs(:,1:ncol(indboot)).
compute templo = llcif(1,1:ncol(indboot)).
compute temphi = ulcif(1,1:ncol(indboot)).
compute outp = {t(temp3), temp2,t(templo),
t(temphi)}.
print outp/title = 'Partially standardized indirect
effect of X on Y'/cnames = clbs/rnames =
indlbl/format = F10.4.
compute temp2 =
stdindf((ncol(indboot)+1):(2*ncol(indboot)),1).
compute temp3 =
effobs(:,(ncol(indboot)+1):(2*ncol(indboot))).
compute templo =
llcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute temphi =
ulcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templo),
t(temphi)}.
print outp/title = 'Completely standardized
indirect effect of X on Y'/cnames = clbs/rnames
= indlbl/format = F10.4.
compute temp2 =
stdindf((2*(ncol(indboot))+1):(3*ncol(indboot)),1
).
compute temp3 =
effobs(:,(2*(ncol(indboot))+1):(3*ncol(indboot)))
.

```



```

compute templow =
llcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute temphi =
ulcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
compute temp =
eff(:,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temp2 =
stdindf((3*(ncol(indboot))+1):(4*ncol(indboot)),1
).
compute temp3 =
effobs(:,(3*(ncol(indboot))+1):(4*ncol(indboot)))
.
compute templow =
llcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temphi =
ulcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
end if.
end if.
do if (nmeds = 2).
compute effkey = {xname, '->', mnames(1,1), '-
>', yname, '', ''}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, '', ''}.
compute effkey = {indlbl(2:4,1), effkey}.
end if.
do if (nmeds = 3).
compute effkey = {xname, '->', mnames(1,1), '-
>', yname, '', '', ''}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, '',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, '',' ',' ',' '}.
compute effkey = {indbl(2:8,1), effkey}.
end if.
do if (nmeds = 4).
compute effkey = {xname, '->' , mnames(1,1), '-
>', yname, '',' ',' ',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,4), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname, '',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,4), '->', yname, '',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, '',' '}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', mnames(1,4), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,4), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', mnames(1,4), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,4), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {indlbl(2:16,1), effkey}.
end if.
print effkey/title = 'Indirect effect key'/format =
A8.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
end if.
DO IF (medte2=1).
compute
name=t(mdlnms2(2:nrow(mdlnms2),1)).
compute tmpdat=datayed(:,2:ncol(datayed)).
compute dd=tmpdat(:,1).

```

```

compute
dd={dd,tmpdat(:,(2+nmeds)),tmpdat(:,2:(1+nmeds))}.
compute stratify=0.
do if (ncovs > 0).
compute cname = cnames.
compute
dd={dd,tmpdat(:,(3+nmeds):(2+nmeds+ncovs))}.
end if.
compute method = 1.
compute samples = boot.
do if (mc > 0).
compute samples = mc.
compute method=2.
end if.
compute omnibus = 1.
compute booterr = 0.
do if (mcx < 0 or mcx > 4).
compute mcx = 1.
end if.
compute ovals = ncol(design(dd(:,1))).
compute nte = notes.
compute criterr = 0.
compute xskip = 0.
compute dumok = 0.
compute badend = 0.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute priorlo = -9999999.
compute priorhi = 9999999.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute reps = samples.
do if (contrast = 1).
compute note(nte,1) = 26.
compute nte=nte+1.

```

```

end if.
do if (effsize = 1).
compute note(nte,1) = 27.
compute nte=nte+1.
end if.
do if (normal = 1).
compute note(nte,1) = 28.
compute nte=nte+1.
end if.
do if (method = 2).
do if (omnibus = 1).
compute note(nte,1) = 25.
compute nte=nte+1.
end if.
compute samples = 0.
do if (reps < 1000).
compute reps = 1000.
end if.
end if.
do if (method = 1).
compute samples = reps.
end if.
compute temp = ncol(dd).
compute n = nrow(dd).
compute ones = make(n,1,1).
compute temp=dd(:,2).
compute nxd = ncol(temp).
compute nx = nxd.
do if (nx = 1 and mcx > 0).
compute temp = dd.
compute temp(GRADE(dd(:,2)),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,2)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute mnvls = cmin(t(nnvls)).
do if (mnvls < 2).
compute errs = errs + 1.
compute runerrs(errs,1) = 37.
compute criterr = 1.

```

```

end if.
do if (nvls > 9).
compute errs = errs+1.
compute runerrs(errs,1) = 36.
compute criterr = 1.
end if.
do if (criterr = 0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,2).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,2) <> nnvls((temp-1),1)).
compute nnvls(temp,1)=dd(i,2).
compute temp = temp+1.
end if.
end loop.
do if (mcx > 0).
compute x = dummy(:,2:ncol(dummy)).
compute nx = ncol(x).
compute minus1 = make(1,ncol(x),-1).
compute xdes=make((nx+1),3,0).
compute xdes(1,1)=dd(1,2).
compute xdes(1,2)=1.
compute temp = 2.
loop k = 2 to n.
do if (dd(k,2) <> dd((k-1),2)).
compute xdes(temp,2) = k.
compute xdes(temp,1) = dd(k,2).
compute xdes((temp-1),3) = k-1.
compute temp=temp+1.
end if.
end loop.
compute xdes((temp-1),3)=n.
compute xdes = {xdes, (xdes(:,3)-xdes(:,2)+1)}.
do if (stratify <> 0).
compute note(nte,1) = 7.
compute nte=nte+1.
end if.
do if (mcx = 4).

```

```

loop k = 1 to n.
do if (rsum(x(k,:)) = 0).
compute x(k,:) = minus1.
end if.
end loop.
end if.
do if (mcx = 2 or mcx = 3).
compute note(nte,1)=4.
compute nte = nte+1.
loop k = 1 to n.
do if (rsum(x(k,:)) > 0).
loop i = 1 to ncol(x).
do if (x(k,i) = 0).
compute x(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcx = 3).
compute conmat1={-8,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1; 0,0,0,0,0,-
3,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.
loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,:)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute x(k,:)=conmat1((rsum(x(k,:))+1),:).
end loop.
end if.
end if.
compute xdata = x.
compute xskip = 1.
compute xname = ddd(1,1:nx).
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).

```

```

compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcx = 4).
compute dummat(1,:) = minus1.
end if.
do if (mcx = 2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcx = 3).
compute dummat=conmat1.
end if.
compute dummat={nnvls, dummat}.
compute xname2={name(1,2), xname}.
end if.
end if.
end if.
compute nm=nmeds.
compute r2b = make(nm,1,-999).
compute indeff = make((nx*nm),1,-999).
compute cov =
make((nx*nm+nm),(nx*nm+nm),0).
compute mns=make(nrow(cov),1,0).
compute ny = ncol(dd)-nm-nxd-ncovs.
compute y = dd(:,1).
compute ydata = y.
compute yname = name(1,1).
do if (xskip = 0).
compute x = dd(:,2:(1+nxd)).
compute xdata = x.
compute xname = name(1,2:(1+nxd)).
compute indlbs = t(xname).
end if.
compute m = dd(:,(2+nxd):(1+nxd+nm)).
compute mdata = m.

```



```

compute mlab = { ' M1 = ';' ' M2 = ';' ' M3 = ';' '
M4 = ';' ' M5 = ';' ' M6 = ';' ' M7 = ';' ' M8 = ';' '
M9 = ';' ' M10 = '}.
compute mlab = {mlab; ' M11 = ';' ' M12 = ';' '
M13 = ';' ' M14 = ';' ' M15 = '}.
compute mname =
name(1,(2+nxd):(1+nxd+nm)).
do if (ncovs > 0).
compute c =
dd(:,(2+nxd+nm):(1+nxd+nm+ncovs)).
compute x = {x, c}.
compute xdata = x.
compute cdata = c.
compute xname = {xname, cname}.
end if.
compute nms={yname; t(mname);
t(xname(1,1:(ncol(xname)-ncovs)))}.
do if (dumok = 1).
compute nms={yname; t(mname); name(1,2)}.
end if.
do if (nm < 16).
do if (dumok = 1).
print dummat/title = 'Coding of categorical X
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
compute
boots=make((samples+1),((nx+1)*nm),-999).
do if (errs = 0 and criterr = 0).
loop k = 1 to (samples+1).
do if (k = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef=0.
do if (k > 1).
compute u=trunc(uniform(n,1)*n)+1.
compute xdata = x(u,:).
compute mdata = m(u,:).

```

```

compute ydata = y(u,:).
do if (ncovs > 0).
compute cdata = c(u,:).
end if.
end if.
loop i = 1 to nm.
compute start = 1+(i-1)*nx.
compute xmat = {ones, xdata}.
do if (covmy=2).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = mdata(:,i).
compute amat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*amat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-(((1-r2)*(n-1))/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute r2cha = r2.
compute adjr2cha = adjr2.
do if (ncovs > 0 and covmy <> 2).
compute xmat = {ones, cdata}.
compute atmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*atmat)**2).
compute r2step1 = (sstotal-ssresid2)/sstotal.
compute r2cha = r2-r2step1.
compute adjr2cha = adjr2- (1-(((1-r2step1)*(n-
1))/(n-ncol(xmat)))).
compute xmat = {ones, xdata}.
end if.
compute bootcoef={bootcoef,t(amat)}.
compute r2b(i,1)=adjr2cha.
do if (k = 1).

```

```

compute hcinvtX=inv(t( xmat )' xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,1)*hcinvtX*t( xmat
(i3,1)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvtX*t( xmat )' xmat
*hcinvtX
.
compute tratio = amat&/sqrt(diag(covmat)).
compute rnms = mname(1,i).
print rnms/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.
compute lmat = ident(nrow(amat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*amat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*amat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/'format = F10.4
/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'.

```

```

compute op = {amat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*(xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute cov(start:(start+(nx-
1)),start:(start+(nx-
1)))=covmat(2:(ncol(covmat)-(ncovs*(covmy <>
2))),2:(ncol(covmat)-(ncovs*(covmy <> 2)))).
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns(start:(start+(nx-
1)),1)=amat(2:(1+nx)).
end loop.
compute xmat = {ones,mdata,xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (k = 1).
print yname/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.

```

```

do if (ovals = 2).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:'/cnames = nmsd/format = F9.2.
end if.
end if.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(yamat)/n)).
compute LL3 = yamat*&ln(pt2)+(1-yamat)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= yamat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).

```

```

compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute bootcoef={bootcoef,t(bmat)}.
compute LL1 = LL2.
compute covmat=varb.
end if.
do if (ovals <> 2).
compute bmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*bmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-(((1-r2)*(n-1))/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute bootcoef={bootcoef,t(bmat)}.
end if.
do if (k = 1).
do if (ovals = 2).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.

```

```

compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).
compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms2/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute
opdirsum=outp((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
end if.
do if (ovals <> 2).
.
compute hcinvXtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtXt*( xmat
(i3,:)).

```

```

end loop.
loop i3=1 to k3.
compute xmat(:,i3) = ( resid(:,ncol( resid
))&/(1-h))&* xmat(:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat(:,i3) = sqrt( mse )&* xmat(:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(bmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/(ncol(xmat)-1).
compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat((2+nmeds):(2+nmeds+(nx-1)),:)=
lmat2.
compute fratiodr =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/nx.
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute df2=(n-ncol(xmat)).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), df2, pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = bmat&/sqrt(diag(covmat)).
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute op = {bmat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat)))) }.

```



```

compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp))))*(xp2/(temp-
(2/3)+(.11/temp)))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms2/format =
F10.4.
compute
opdirsum=op((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
compute xmat = {ones,mdata}.
do if (ncovs > 0).
compute xmat = {xmat, cdata}.
end if.
compute btmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*btmat)**2).
compute r2direct = r2-((sstotal-
ssresid2)/sstotal).
compute pfr = 1-fcdf(fratiodr,(nx),(n-1-nx-nm-
ncovs)).
compute r2direct = {r2direct, fratiodr, nx, (n-1-
nx-nm-ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms2/cnames=rnms2/form
at= F10.4.
end if.
end if.

```

```

do if (k = 1 and toteff = 1).
print yname/title = '*****'
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:.'/format = A8.
compute xmat = {ones, xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(ymat)/n)).
compute LL3 = ymat*&ln(pt2)+(1-ymat)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= ymat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.

```

```

compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute LL1 = LL2.
compute covmat=varb.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(bmat)-
1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms = {'Constant'; t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).

```

```

compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute optotsum=outp(2:(1+nx),:).
end if.
do if (ovals <> 2).
compute cmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*cmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).

.
compute hcinvXtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
)))/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).

```

```

loop i3=1 to k3.
compute xmat(:,i3) = sqrt( mse )&* xmat(:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(cmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = cmatrix/sqrt(diag(covmat)).
compute op = {cmatrix, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute optotsum=op(2:(1+nx),:).
compute ssresid2 = ssresid.
compute r2total = r2.

```

```

compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat(2:(1+nx),:) = lmat2.
compute fratio3 =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/nx.
do if (ncovs > 0).
compute xmat = {ones, cdata}.
compute ctmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*ctmat)**2).
compute r2total = r2-((sstotal-ssresid2)/sstotal).
end if.
compute pfr = 1-fcdf(fratio3,(nx),(n-1-nx-
ncovs)).
compute r2tot2 = {r2total, fratio3, nx, (n-1-nx-
ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns((nrow(mns)-
nm+1):nrow(mns),1)=bmat(2:(1+nm),1).
compute tmp = 1.
loop i = 1 to (nm*nx).
compute
boots(k,i)=mns(i,1)*mns(((nm*nx)+tmp),1).
do if ((i/nx)=trunc(i/nx)).
compute
boots(k,((nm*nx)+tmp))=r2b(tmp,1)*mns(((nm*
nx)+tmp),1).
compute tmp = tmp+1.
end if.
end loop.
do if (savboot = 1 and k > 1).

```

```

compute bootstrp((k-1),:)=bootcoef.
end if.
end loop.
do if (method = 1).
compute llci=make(1,ncol(boots),-999).
compute ulci=make(1,ncol(boots),-999).
compute indeff = t(boots(1,1:(nm*nx))).
compute indeff22=t(boots(1,:)).
compute omni = t(boots(1,((nm*nx)+1) :
ncol(boots))).
do if (nrow(boots) > 1).
compute boots = boots(2:nrow(boots),:).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
loop i = 1 to ncol(boots).
.
compute temp = boots(:,i).
compute temp(GRADE( boots(:,i) )) = boots(:,i).
compute badlo = 0.
compute badhi = 0.
do if ( indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(indeff22(i,1)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2)))+1.

```

```

do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,i)=llcit.
compute ulci(1,i)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute llci = t(llci).
compute ulci= t(ulci).
compute llci={llci,ulci}.
end if.
end if.
do if (method = 2).
compute tmp=1.
loop i = 1 to (nm*nx).
compute
indeff(i,1)=mns(i,1)&*mns(((nm*nx)+tmp),1).

```



```

do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = sqrt(-
2*ln(uniform(reps,nrow(cov))))&*cos((2*3.1415
9265358979)*uniform(reps,nrow(cov))).
compute boots=boots*chol(cov).
loop i = 1 to ncol(boots).
compute boots(:,i)=boots(:,i)+mns(i,1).
end loop.
compute tmp=1.
loop i = 1 to (nm*nx).
compute
boots(:,i)=boots(:,i)&*boots(:,((nm*nx)+tmp)).
compute temp = boots(:,i).
compute temp(GRADE(boots(:,i))) = boots(:,i).
compute boots(:,i) = temp.
do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = boots(:,1:(nm*nx)).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
compute llci =
t(((boots(cilow,:));(boots(cihigh,:)))).
end if.
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
compute rnmssum=rnms(2:(1+nx),1).
do if (ovals <> 2).
print optotsum/title = 'Relative total effects of X
of Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.

```

```

print r2tot2/title = 'Omnibus test of total effect of
X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print optotsum/title = 'Relative total effects of X
on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
end if.
compute
rnmssum=rnms2((2+nmeds):(1+nmeds+nx),1).
do if (ovals <> 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.
print r2direct/title = 'Omnibus test of direct
effect of X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
do if (omnibus = 1 and (method = 1 or method
= 3)).
compute indlbs = {indlbs; 'Omnibus'}.
end if.
loop i = 1 to nm.
compute op = mname(1,i).
print op/title = 'Relative indirect effect(s) of X on
Y through:.'/format = A8/space=0.
do if (method = 2).
compute clbs = {'Effect', 'SE MC', 'MC LLCI',
'MC ULCI'}.
compute op = {indeff, bootse(1:(nm*nx)), lici}.
end if.

```

```

do if (method = 1).
compute clbs = {'Effect', 'SE(boot)', 'LLCI',
'ULCI'}.
compute op = indeff.
do if (samples > 0).
compute op = {indeff, bootse(1:(nm*nx),:),
llci(1:(nm*nx),:)}).
end if.
end if.
compute temp = op((((i-1)*nx)+1):(nx*i),:).
do if ((method = 1) and omnibus = 1).
do if (samples = 0).
compute temp={temp;omni(i,1)}.
end if.
do if (samples > 0).
compute temp = {temp; omni(i,1),
bootse(((nm*nx)+i),:), llci(((nm*nx)+i),:)}).
end if.
end if.
print temp/title = '/rnames = indlbs/cnames =
clbs/format = F10.4 /space=0.
print/title = '-----'.
end loop.
end if.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
END IF.
DO IF (mdpbe=1).
compute nm={yname,xname,mnames}.
do if (ncovs > 0).

```

```

compute
nm={yname,cnames,xname,mnames}.
end if.

compute tpx=xname.
compute x = dd(:,2:ncol(dd)).
compute tpy = yname.
compute tempy= dd(:,1).
compute n = nrow(dd).
compute nx=1.
compute criterr=0.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute ddd1 = {'int_1', 'int_2', 'int_3', 'int_4',
'int_5', 'int_6', 'int_7', 'int_8', 'int_9'}.
compute dumok = 0.
compute mcfoc=mcx.
compute mcmmod=mcm.
compute mcloc=(mcfoc>0).
compute con = make(n,1,1).
compute ncovs=ncol(x)-2.
do if (mcfoc > 0 or mcmmod > 0).
compute temp = dd.
compute temp(GRADE(dd(:,(ncol(dd)-
mcloc))),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,ncol(dd)-
mcloc)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute toosmall=rsum(nnvls < 2).
compute mnvls = cmin(t(nnvls)).
do if ((rsum(nnvls < 2)) > 0).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=37.
end if.
do if (nvls > 10).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=36.

```

```

end if.
do if (criterr=0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,ncol(dd)-mcloc).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,ncol(dd)-mcloc) <> nnvls((temp-
1),1)).
compute nnvls(temp,1)=dd(i,ncol(dd)-mcloc).
compute temp = temp+1.
end if.
end loop.
compute dummy = dummy(:,2:ncol(dummy)).
do if (mcfoc=4 or mcmmod=4).
compute minus1=make(1,ncol(dummy),-1).
loop k = 1 to n.
do if (rsum(dummy(k,:)) = 0).
compute dummy(k,:)=minus1.
end if.
end loop.
end if.
do if (mcfoc = 2 or mcfoc = 3) or (mcmmod = 2 or
mcmmod = 3)).
loop k = 1 to n.
do if (rsum(dummy(k,:)) > 0).
loop i = 1 to ncol(dummy).
do if (dummy(k,i) = 0).
compute dummy(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcfoc = 3 or mcmmod=3).
compute conmat1={-8,1,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1,1; 0,0,0,0,0,-
3,1,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.

```

```

loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,:)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute
dummy(k,:)=conmat1((rsum(dummy(k,:))+1),:).
end loop.
end if.
end if.
compute nx = ncol(dummy).
compute xname = ddd(1,1:nx).
compute xdata = dummy.
compute xname = ddd(1,1:nx).
compute xname2={nm(1,(ncol(nm)-mcloc)),
xname}.
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcfoc = 2 or mcmmod=2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcfoc = 3 or mcmmod=3).
compute dummat=conmat1.
end if.
do if (mcfoc = 4 or mcmmod = 4).
compute dummat(1,:)=minus1.
end if.
compute dummat={nnvls, dummat}.
end if.
end if.
do if (criterr=0).
compute y = dd(:,1).
compute ovals = ncol(design(dd(:,1))).

```

```

compute itprob = 0.
do if (ovals = 2).
compute omx = cmax(y(:,1)).
compute omn = cmin(y(:,1)).
compute y(:,1) = (y(:,1) = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute sstotal = csum((y-(csum(y)/n))&**2).
compute outv = t(nm(1,1)).
compute mdtr = nm(1,ncol(dd)).
compute fciv = nm(1,(ncol(dd)-1)).
do if (mcfoc <> 0 or mcmod <> 0).
compute inter=make(n,nx,0).
loop i = 1 to nx.
compute inter(:,i)=dummy(:,i)&*dd(:,ncol(dd)-
(1-mcloc)).
end loop.
do if (ncovs = 0).
compute x={con,dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
do if (ncovs > 0).
compute
x={con,dd(:,2:(1+ncovs)),dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={t(nm(1,2:(1+ncovs)));
nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
end if.
compute xmns = csum(x)&/n.
compute focvals = {1,0,0}.
compute highwarn = 0.
compute lowwarn = 0.

```

```

do if (ncol(x) > (2*nx+2)).
compute covmns = {1, xmns(1,2:(ncol(x)-1-
(2*nx)))}.
compute focvals = {covmns,0,0}.
end if.
compute dfres = n-ncol(x).
do if (ovals = 2).
compute pt2 = make(n,1,(csum(y)/n)).
compute LL3 = y*&ln(pt2)+(1-y)&ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(x),1,0).
compute LL1 = 0.

.

compute pt1lp= pt1.
compute xlp= x.
compute ylp= y.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).

```



```

compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
else if (ovals <> 2).
compute invXtX=inv(t(x)*x).
compute b = invXtX*t(x)*y.
compute resid = y-(x*b).
compute ssresid=csum(resid&*resid).
compute msresid=ssresid/dfres.
compute varb = msresid*invXtX.
compute
seint=sqrt(varb(nrow(varb),nrow(varb))).
compute k3 = nrow(b).
do if (hc3 = 1).
compute xhc=x.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)= xhc(i3,:)*invXtX*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).
end loop.
compute varb=(invXtX*t(xhc)*xhc*invXtX).
end if.
compute seb = sqrt(diag(varb)).
compute r2 = 1-(ssresid/sstotal).

```

```

do if (mcfoc <> 0 or mcmod <> 0).
compute ytp=y.
compute xtp=x(:,1:(ncol(x)-nx)).
compute btp = inv(t(xtp)*xtp)*t(xtp)*ytp.
compute residt = ytp-(xtp*btp).
compute ssresidt=csum(residt*residt).
compute r2noint = 1-(ssresidt/sstotal).
end if.
compute pr = ncol(x)-1.
compute lmat = ident(nrow(b)).
compute lmat = lmat(:,2:ncol(lmat)).
compute f =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
pr).
compute pf = 1-fcdf(f,pr,dfres).
compute pf = {sqrt(r2),r2,msresid,f,pr,dfres,pf}.
compute xd=abs(xp2).
end if.
do if (itprob = 0).
compute tstat = b&/seb.
do if (ovals <> 2).
compute p = 2*(1-tcdf(abs(tstat), dfres)).
else if (ovals = 2).
compute p = 2*(1-cdfnorm(abs(tstat))).
end if.
compute outp = {b,seb,tstat,p}.
compute nms = {'constant'; nms; 'interact'}.
do if (ovals <> 2).
compute tval = (dfres* (exp((dfres-
(5/6))*((xd/(dfres-(2/3)+(.11/dfres)))*(xd/(dfres-
(2/3)+(.11/dfres)))))-1)).
compute outp={outp,(b-
sqrt(tval)&*seb),(b+sqrt(tval)&*seb)}.
end if.
compute bb = tval.
do if (dumok = 1).
do if (mcfoc > 0).
print dummat/title = 'Coding of categorical X
variable for analysis:'/cnames = xname2/format
= F5.2.

```

```

else if (mcmod > 0).
print dummat/title = 'Coding of categorical M
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
print yname/title =
*****
*****'/rlabels = 'Outcome: '/format
= A8.
do if (ovals = 2).
compute nmsd = {outv, 'Analysis'}.
print rcd/title = 'Coding of binary Y for
analysis: '/cnames = nmsd/format = F10.4.
compute outp = {outp,(outp(:,1)-
sqrt(tval)&*outp(:,2)),(outp(:,1)+sqrt(tval)&*outp
(:,2))}.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
print outp/title = 'Logistic Regression
Model'/rnames = nms/clabels 'Coeff' 'se' 'Z' 'p'
'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
compute varbtmp=varb.
compute btmp=b.

```

```

compute LL2f=LL2.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=x(:,1:(ncol(x)-nx)).
compute bt1 = make(ncol(xlp),1,0).

.

compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(y-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)*&ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.

```

```

compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL2f.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(LLdiff,nx).
compute rcha={LLdiff,nx,pchi}.
end if.
do if (ovals <> 2).
print pf/title = 'Model Summary'/clabels = 'R' 'R-
sq' 'MSE' 'F' 'df1' 'df2' 'p'/format F10.4.
compute lmat=make(nrow(b),1,0).
compute lmat(nrow(lmat),1)=1.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
1).
compute
rcha=((b(nrow(b),1)/seint)*(b(nrow(b),1)/seint))*
(1-r2)/dfres.
compute rcha = {rcha, fcha, 1, dfres,
outp((pr+1),4)}.
print outp/title = 'Model'/rnames = nms/clabels
'coeff' 'se' 't' 'p' 'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
do if (mcfoc > 0 or mcmod > 0).
compute rcha=r2-r2noint.
compute lmat=make((nrow(b)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
nx).
compute pvalr2c=1-fcdf(fcha,nx,dfres).

```

```

compute rcha = {rcha, fcha, nx, dfres,pvalr2c}.
end if.
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute intkey = {'a', 'b', 'c', 'd', 'e'}.
loop i = 1 to nx.
compute intkey={intkey; ddd1(1,i), ' : ', ddd(1,i),
' X ', nm(1,ncol(dd)-(1-mcloc))}.
end loop.
compute intkey=intkey(2:nrow(intkey),:).
print intkey/title='Product terms key:.'/format =
a8.
end if.
do if (ovals <> 2).
print rcha/title = 'R-square increase due to
interaction:.'/clabels 'R2-chng' 'F' 'df1' 'df2'
'p'/format F10.4.
else if (ovals = 2).
print rcha/title = 'Likelihood ratio test for
interaction:.'/clabels 'Chi-sq' 'df' 'p'/format F10.4.
end if.
print/title =
*****
*****'.
compute mdvar = x(:,(ncol(x)-1)).
do if (mcfoc = 0 and mcmmod = 0).
compute g1 = b((ncol(x)-2),1).
compute g3 = b(ncol(x),1).
compute vg1 = varb((ncol(x)-2),(ncol(x)-2)).
compute vg3 = varb(ncol(x),ncol(x)).
compute covg1g3 = varb((ncol(x)-2), ncol(x)).
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute mdvar = x(:,(ncol(x)-(2*nx))).
end if.
compute mdmin = cmin(mdvar).
compute mdmax = cmax(mdvar).
compute fvar = x(:,(ncol(x)-2)).
compute nval = ncol(design(mdvar)).
compute fvmin = cmin(fvar).

```

```

compute fvmax = cmax(fvar).
do if (mmodval = 999).
compute mnmd = csum(mdvar)/n.
compute tmp = make(n,1,mnmd).
compute sdmd = sqrt(csum(((mdvar-
tmp)**2))/(n-1)).
compute probeval = {mnmd-sdmd; mnmd;
mnmd+sdmd}.
do if (probeval(1,1) < mdmin).
compute lowwarn = 1.
compute probeval(1,1)=mdmin.
end if.
do if (probeval(nrow(probeval),1) > mdmax).
compute highwarn = 1.
compute probeval(nrow(probeval),1)=mdmax.
end if.
do if (quantile = 1).
compute tmp = mdvar.
compute tmp(GRADE(mdvar(:,1)),:)= mdvar.
compute mdvar = tmp.
compute
probeval={mdvar(trunc(0.1*n),1);mdvar(trunc(0
.25*n),1);mdvar(trunc(0.5*n),1);mdvar(trunc(0.7
5*n),1);mdvar(trunc(0.9*n),1)}.
end if.
end if.
do if (nval = 2).
compute probeval = make(2,1,0).
compute probeval(1,1)=cmin(mdvar).
loop i = 1 to n.
do if (mdvar(i,1) <> probeval(1,1)).
compute probeval(2,1) = mdvar(i,1).
BREAK.
end if.
end loop.
end if.
do if (mmodval <> 999).
compute probeval = mmodval.
end if.
compute outp = make(nrow(probeval),7,0).

```

```

do if (mcfoc > 0 or mcmod > 0).
compute focvals=make(1,ncol(x)+1,1).
do if (mcfoc > 0).
print/title = 'Conditional effect of X on Y at
values of the moderator:'.
print/title = ' '/space=0.
compute rnn2=mdtr.
compute matt=make(nx,6,0).
end if.
loop jj=1 to nrow(probeval).
do if (mcfoc > 0).
loop ii=1 to nx.
compute g1=b((ncol(x)-(2*nx)+ii),1).
compute g3=b((ncol(x)-nx+ii),1).
compute vg1=varb((ncol(x)-(2*nx)+ii),(ncol(x)-
(2*nx)+ii)).
compute vg3=varb((ncol(x)-nx+ii),(ncol(x)-
nx+ii)).
compute covg1g3=varb((ncol(x)-
(2*nx)+ii),(ncol(x)-nx+ii)).
compute x2 = probeval(jj,1).
compute w1 = g1+g3*x2.
compute varw1 =
vg1+(2*x2*covg1g3)+((x2*x2)*vg3).
compute sew1 = sqrt(varw1).
compute t1 = w1/sew1.
do if (ovals <> 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-tcdf(abs(t1), dfres)).
compute cnms = {'Coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)= {w1,sew1,t1,p,llci,ulci}.
else if (ovals = 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-cdfnorm(abs(t1))).
compute cnms = {'Coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)= {w1,sew1,t1,p,llci,ulci}.

```



```

end if.
end loop.
compute rnms=t(xname).
compute mdvalpr=probeval(jj,1).
print mdvalpr/title = 'Moderator
value: '/rnames=rnn2/format= F10.4 /space=0.
print matt/title='
'/cnames=cnms/rnames=rnms/format= F10.4
/space=0.
compute xprob=x(:,(ncol(x)-(2*nx)))-mdvalpr.
loop kk = 1 to nx.
compute xprob={xprob,
(xprob(:,1)&*x(:,(ncol(x)-(2*nx)+kk)))}.
end loop.
compute xprob={x(:,1:((ncol(x)-(2*nx))-
1)),xprob}.
do if (ovals <> 2).
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute residc = y-(xprob*bmultc).
compute ssresidc=csum(residc&*residc).
compute r2c = r2-(1-(ssresidc/sstotal)).
compute fcha2=(dfres*r2c)/(nx*(1-r2)).
do if (hc3 = 1).
loop kk = 1 to nx.
compute xprob={xprob, x(:,(ncol(x)-
(2*nx)+kk))}.
end loop.
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute k3 = nrow(bmultc).
compute xhc=xprob.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)=
xhc(i3,:)*inv(t(xprob)*xprob)*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).

```

```

end loop.
compute
varbc=(inv(t(xprob)*xprob)*t(xhc)*xhc*inv(t(xprob)*xprob)).
compute lmat=make((nrow(bmultc)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha2 =
(t(t(lmat)*bmultc)*inv(t(lmat)*varbc*lmat)*((t(lmat)*bmultc)))/nx).
end if.
compute pvalr2cc=1-fcdf(fcha2,nx,dfres).
compute rcha2 = {r2c, fcha2, nx,
dfres,pvalr2cc}.
print rcha2/title = 'Test of equality of conditional
means at this value of the moderator'/clabels
'R2-chng' 'F' 'df1' 'df2' 'p'/format F10.4.
else if (ovals = 2).
compute btmp=b.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=xprob.
compute bt1 = make(ncol(xlp),1,0).

.
compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.

```

```

end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL4.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(lldiff,nx).
compute rcha2 = {LLdiff, nx,pchi}.
print rcha2/title = 'Test of equality of log odds
conditioned on this moderator value'/clabels
'Chi-sq' 'df' 'p'/format F10.4.
end if.
end if.
compute
tttt=make(nrow(dummat),1,probeval(jj,1)).
compute tttt={tttt,dummat(:,2:ncol(dummat))}.
loop kkk=1 to nx.
compute tttt={tttt,tttt(:,1)*tttt(:,1+kkk)}.
end loop.
compute ones=make(nrow(dummat),1,1).
do if (ncol(x) > (2*nx+2)).

```

```

compute
covmnmat=make(nrow(tttt),ncol(covmns),0).
loop kkk=1 to nrow(tttt).
compute covmnmat(kkk,:)=covmns.
end loop.
compute tttt={covmnmat,tttt}.
else.
compute tttt={ones,tttt}.
end if.
compute focvals={focvals;dummat(:,1),tttt}.
do if (mcfoc > 0).
compute yhat={dummat(:,1),(tttt*b)}.
compute cmnms={fciv, 'yhat'}.
do if (ovals <> 2).
print yhat/title = 'Estimated conditional means
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (ovals = 2).
print yhat/title = 'Estimated conditional log odds
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (jj <> nrow(probeval)).
print/title='-----'/space=0.
end if.
end if.
end loop.
compute focvals=focvals(2:nrow(focvals),:).
compute yhat=focvals(:,2:ncol(focvals))*b.
compute
focvals={focvals(:,1),focvals(:,(ncol(focvals)-
(2*nx))),yhat}.
compute cnms={fciv,mdtr,'yhat'}.
do if (mcmod > 0).
compute cnms={mdtr,fciv,'yhat'}.
end if.
end if.
do if (mcmod > 0).
compute outp=make((nx+1),7,0).

```

```

compute bcatm={b((ncol(x)-
(2*nx)),1);b((ncol(x)-nx+1):ncol(x),1)}.
compute outp(:,1)=dummat(:,1).
compute bcatcov=varb((ncol(x)-
nx):ncol(x),(ncol(x)-nx):ncol(x)).
compute bcatcov(1,1)=varb((ncol(x)-
(2*nx)),(ncol(x)-(2*nx))).
compute
bcatcov(2:nrow(bcatcov),1)=varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx)).
compute
bcatcov(1,2:nrow(bcatcov))=t(varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx))).
loop i = 1 to nrow(dummat).
compute
catmval={1,dummat(i,2:ncol(dummat))}.
compute condeff=catmval*bcatm.
compute
condse=sqrt(catmval*bcatcov*t(catmval)).
compute outp(i,2:3)={condeff,condse}.
end loop.
compute outp(:,4)=outp(:,2)&/outp(:,3).
compute outp(:,5) = 2*(1-tcdf(abs(outp(:,4))),
dfres)).
compute outp(:,6) = (outp(:,2)-
sqrt(tval)&*outp(:,3)).
compute outp(:,7) =
(outp(:,2)+sqrt(tval)&*outp(:,3)).
do if (ovals <> 2).
compute cnmms = {xname2(1,1), 'coeff', 'se',
't', 'p', 'LLCI', 'ULCI'}.
end if.
do if (ovals = 2).
compute outp(:,5) = 2*(1-
cdfnorm(abs(outp(:,4)))).
compute cnmms = {xname2(1,1), 'coeff', 'se',
'Z', 'p', 'LLCI', 'ULCI'}.
end if.

```

```

print outp/title = 'Conditional Effect of Focal
Predictor in Groups Defined by the Moderator
Variable:.'/cnames=cnmms/format = F10.4.
end if.
do if (nval > 2 and (mmodval = 999) and
mcmmod = 0).
do if (quantile <> 1).
print/title = 'Moderator values are the sample
mean and plus/minus one SD from mean'.
else.
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
end if.
do if (highwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD above the mean was'.
print/title = 'replaced with the maximum
because one SD above the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
do if (lowwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD below the mean was'.
print/title = 'replaced with the minimum
because one SD below the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
end if.
do if (nval = 2 and (mmodval = 999)).
print/title = 'The moderator variable is
dichotomous'.
end if.
do if (mmodval <> 999 and mcmmod = 0 and
((mmodval < mdmin) or (mmodval > mdmax))).
print/title = 'Warning: Moderator value specified
is outside of the range of the data'.
end if.

```

```

do if (plot = 1).
do if (mcfoc = 0 and mcmmod=0).
compute cnms = {t(nms((ncol(x)-2):(ncol(x)-
1),1)), 'yhat'}.
end if.
do if (ovals = 2).
compute prob =
exp(focvals(:,3))/(1+exp(focvals(:,3))).
compute focvals = {focvals, prob}.
compute cnms = {cnms, 'prob'}.
end if.
print/title =
*****
*****!

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (ovals = 2).
do if (mcfoc=0).
print/title = 'DATA LIST FREE/Eco_Innovation
Plan_Orga_Pract logodds prob.' /space=1.
end if.
do if (mcmmod=0).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Eco_Innovation
logodds prob.' /space=1.

```

```

end if.
else if (ovals <> 2).
do if (mcfoc=0).
print/title = 'DATA LIST FREE/Eco_Innovation
Plan_Orga_Pract Operational_Performance.'
/space=1.
end if.
do if (mcmod=0).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Eco_Innovation
Operational_Performance.' /space=1.
end if.
end if.
print/title = 'BEGIN DATA.' /space=0.
print focvals/title = ' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (ovals <> 2 and mcmod=0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Operational_Performance BY
Plan_Orga_Pract.' /space=0.
end if.
do if (ovals <> 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH Operational_Performance BY
Eco_Innovation.' /space=0.
end if.
do if (ovals = 2 and mcmod=0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Plan_Orga_Pract.'
/space=0.
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Plan_Orga_Pract.' /space=0.
end if.
do if (ovals = 2 and mcfoc=0).

```



```

print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH logodds BY Eco_Innovation.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH prob BY Eco_Innovation.' /space=0.
end if.
do if (ncovs > 0).
print/title = 'NOTE: For data above, covariates
are set to their sample means.'.
end if.
end if.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
end if.
END IF.
end if.
do if (bad > 0).
compute note(notes,1) = 9.
compute notes = notes + 1.
end if.
print/title = '***** ANALYSIS
NOTES AND WARNINGS
*****'.
loop i = 1 to errs.
do if (runerrs(i,1) = 1).
print/title = 'ERROR: One of your declared
mediators is dichotomous. This procedure
can"t be used.'.
end if.
do if (runerrs(i,1) = 2).

```

```
print/title = 'ERROR: For model 6, this
procedure limits the number of mediators to
four.'.
end if.
do if (runerrs(i,1) = 3).
print/title = 'ERROR: For models 1, 2, and 3,
only a single variable can be listed in the M
list.'.
end if.
do if (runerrs(i,1) = 4).
print/title = 'ERROR: You requested a model
involving W but did not provide a valid W
variable name.'.
end if.
do if (runerrs(i,1) = 5).
print/title = 'ERROR: You requested a model
involving Z but did not provide a valid Z
variable name.'.
end if.
do if (runerrs(i,1) = 6).
print/title = 'ERROR: You requested a model
involving Q but did not provide a valid Q
variable name.'.
end if.
do if (runerrs(i,1) = 7).
print/title = 'ERROR: You requested a model
involving V but did not provide a valid V
variable name.'.
end if.
do if (runerrs(i,1) = 8).
print/title = 'ERROR: You specified a W
variable for a model that does not need it'.
end if.
do if (runerrs(i,1) = 9).
print/title = 'ERROR: You specified a Z variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 10).
print/title = 'ERROR: You specified a Q variable
for a model that does not need it'.
```

```

end if.
do if (runerrs(i,1) = 11).
print/title = 'ERROR: You specified a V variable
for a model that does not need it.'.
end if.
do if (runerrs(i,1) = 12).
print/title = 'ERROR: The variable specified for
W has already been assigned.'.
end if.
do if (runerrs(i,1) = 13).
print/title = 'ERROR: The variable specified for
Z has already been assigned.'.
end if.
do if (runerrs(i,1) = 14).
print/title = 'ERROR: The variable specified for
Q has already been assigned.'.
end if.
do if (runerrs(i,1) = 15).
print/title = 'ERROR: The variable specified for
V has already been assigned.'.
end if.
do if (runerrs(i,1) = 16).
print/title = 'ERROR: You did not provide a
valid Y variable name.'.
end if.
do if (runerrs(i,1) = 17).
print/title = 'ERROR: The variable specified for
Y has already been assigned.'.
end if.
do if (runerrs(i,1) = 18).
print/title = 'ERROR: Model 6 requires more
than one mediator.'.
end if.
do if (runerrs(i,1) = 19).
print/title = 'ERROR: You have not specified a
valid model number.'.
end if.
do if (runerrs(i,1) = 20).
print/title = 'ERROR: At least one and only one
variable must be listed for X.'.

```

```

end if.
do if (runerrs(i,1) = 21).
print/title = 'ERROR: At least one and only one
variable must be listed for Y.'.
end if.
do if (runerrs(i,1) = 22).
print/title = 'ERROR: Iteration didn"t converge
to a solution. Interpret results with caution.'.
end if.
do if (runerrs(i,1) = 23).
print/title = 'ERROR: You specified a clustering
variable that does not exist in your variable
list.'.
end if.
do if (runerrs(i,1) = 24).
print/title = 'ERROR: You specified a clustering
variable that has already been assigned.'.
end if.
do if (runerrs(i,1) = 25).
print/title = 'ERROR: One or more of your M
variables is not listed in the variables list.'.
end if.
do if (runerrs(i,1) = 26).
print/title = 'ERROR: A maximum of 20 cluster
units is allowed. Use multilevel modeling
instead.'.
end if.
do if (runerrs(i,1) = 27).
print/title = 'ERROR: One of the variables in
your model is a constant.'.
end if.
do if (runerrs(i,1) = 28).
print/title = 'ERROR: Dichotomous Y is not
permitted with WS option.'.
end if.
do if (runerrs(i,1) = 29).
print/title = 'ERROR: Insufficient number of
variables in vars= list when using WS option.'.
end if.
do if (runerrs(i,1) = 30).

```

```

print/title = 'ERROR: Too many variables in
vars= list when using WS option. Covariates
not allowed.'.
end if.
do if (runerrs(i,1) = 31).
print/title = 'ERROR: mmodval and wmodval
can"t both be set to zero with contrast option.'.
end if.
do if (runerrs(i,1) = 32).
print/title = 'ERROR: You did not provide a
valid X variable name.'.
end if.
do if (runerrs(i,1) = 33).
print/title = 'ERROR: PROCESS requires all
variable names to be eight characters or
fewer.'.
print/title = '      Please shorten variable names
and reexecute.'/space=0.
end if.
do if (runerrs(i,1)=34).
print/title = 'ERROR: X and M cannot both be
specified as multicategorical.'.
end if.
do if (runerrs(i,1)=35).
print/title = 'ERROR: Multicategorical variable
specification available only for models 1 and
4.'.
end if.
do if (runerrs(i,1)=36).
print/title = 'ERROR: Categorical variables
cannot have more than 10 categories.'.
end if.
do if (runerrs(i,1)=37).
print/title = 'ERROR: Each group must have at
least two cases.'.
end if.
do if (runerrs(i,1)=38).
print/title = 'ERROR: Cluster option not
available with multicategorical variables.'.
end if.

```

```

do if (runerrs(i,1)=39).
print/title = 'ERROR: A maximum of 10
variables is allowed in the M= list.'.
end if.
do if (runerrs(i,1) = 41).
print/title = 'ERROR: MCM option not permitted
in model 4.'.
end if.
end loop.
do if (errs = 0).
do if (boot > 1 or mc > 0).
do if (bconoff = 1 and boot > 1).
print boot/title = 'Number of bootstrap samples
for bias corrected bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (bconoff = 0 and boot > 1).
print boot/title = 'Number of bootstrap samples
for percentile bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (mc > 1).
print mc/title = 'Number of samples for Monte
Carlo confidence intervals:'/format = F8.0.
end if.
do if (booterr = 1).
compute badend = badend(1,2:ncol(badend)).
print badend/title = 'WARNING: Bootstrap CI
endpoints below not trustworthy. Decrease
confidence or increase bootstraps'/format =
F10.4.
end if.
end if.
print conf/title = 'Level of confidence for all
confidence intervals in output:'/format = F8.2.
do if ((center = 1 or ws=1) and (ncol(centvar) >
1)).
compute centvar = centvar(1,2:ncol(centvar)).

```

```

print centvar/title = 'NOTE: The following
variables were mean centered prior to
analysis:.'/format = a8.
end if.
loop i = 1 to notes.
do if (note(i,1) = 1).
print/title = 'NOTE: Confidence level restricted
to between 50 and 99.9999%. 95% confidence
is provided in output'.
end if.
do if (note(i,1) = 2).
print/title = 'NOTE: Effect size measures for
indirect effects are not available for models
with dichotomous outcomes'.
end if.
do if (note(i,1) = 3).
print/title = 'NOTE: All standard errors for
continuous outcome models are based on the
HC3 estimator'.
end if.
do if (note(i,1) = 6).
print/title = 'NOTE: The number of bootstrap
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 7).
print/title = 'NOTE: The Johnson-Neyman
method is available only for Models 1 and 3'.
end if.
do if (note(i,1) = 8).
print/title = 'NOTE: The Johnson-Neyman
method cannot be used with a dichotomous
moderator'.
end if.
do if (note(i,1)=9).
print bad/title = 'NOTE: Some bootstrap
samples had to be replaced. The number of
such replacements was:'.
end if.
do if (note(i,1) = 11).

```

```

print nmiss/title = 'NOTE: Some cases were
deleted due to missing data. The number of
such cases was:'.
end if.
do if (note(i,1) = 12).
print /title = 'NOTE: Monte Carlo method
available only for models 4 and 5.
Bootstrapping was used instead.'.
end if.
do if (note(i,1) = 13).
print/title = 'NOTE: The number of Monte Carlo
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 19).
print/title = 'NOTE: Effect sizes not available for
within-subject analyses.'.
end if.
do if (note(i,1) = 16).
print/title = 'NOTE: Normal theory tests not
available for within-subject analyses.'.
end if.
do if (note(i,1) = 17).
print/title = 'NOTE: Monte Carlo confidence
intervals not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 18 and warnrep=0).
print/title = 'WARNING: You have requested
OLS estimation with a dichotomous criterion.'.
print/title = 'Interpret model coefficients and
inferential statistics with caution.'/space=0.
compute warnrep=1.
end if.
do if (note(i,1) = 20).
print/title = 'NOTE: Saving of bootstrap
estimates not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 22).

```



```
print/title = 'NOTE: Effect size option with
covariates requires covariates in models of M
and Y.'.
end if.
do if (note(i,1) = 23).
print/title = 'NOTE: Johnson-Neyman method
not available with multicategorical X'.
end if.
do if (note(i,1) = 24).
print/title = 'NOTE: MMODVAL option not
available with a multicategorical moderator'.
end if.
do if (note(i,1) = 25).
print/title = 'NOTE: Omnibus test for relative
indirect effects is not available with the MC
option.'.
end if.
do if (note(i,1) = 26).
print/title = 'NOTE: CONTRAST option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 27).
print/title = 'NOTE: EFFSIZE option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 28).
print/title = 'NOTE: NORMAL option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 31).
print/title = 'NOTE: The TOTAL option is not
available when Y is dichotomous.'.
end if.
do if (note(i,1) = 32).
print/title = 'NOTE: Kappa-squared is disabled
from output as of version 2.16.'.
end if.
do if (note(i,1) = 33).
print/title = 'NOTE: COVMY option ignored in
models 1, 2, and 3.'.
```

| | | | |
|-----------|----------------|-------------|-------------|
| | | end if. | |
| | | end loop. | |
| | | end if. | |
| | | end matrix. | |
| Resources | Processor Time | | 00:00:04.76 |
| | Elapsed Time | | 00:00:04.81 |

Run MATRIX procedure:

***** PROCESS Procedure for SPSS Release 2.16.3 *****

Written by Andrew F. Hayes, Ph.D. www.afhayes.com
Documentation available in Hayes (2013). www.guilford.com/p/hayes3

Model = 4
Y = Operatio
X = Plan_Org
M = Eco_Inno

Sample size
856

Outcome: Eco_Inno

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|-----------|--------|----------|-------|
| .8299 | .6887 | .3590 | 1889.6042 | 1.0000 | 854.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|-------|-------|---------|-------|-------|-------|
| constant | .3531 | .0810 | 4.3618 | .0000 | .1942 | .5121 |
| Plan_Org | .8827 | .0203 | 43.4696 | .0000 | .8429 | .9226 |

Outcome: Operatio

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|----------|--------|----------|-------|
| .8332 | .6941 | .3670 | 967.9640 | 2.0000 | 853.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|-------|-------|---------|-------|-------|-------|
| constant | .2605 | .0828 | 3.1478 | .0017 | .0981 | .4230 |
| Eco_Inno | .4238 | .0346 | 12.2506 | .0000 | .3559 | .4917 |
| Plan_Org | .4935 | .0368 | 13.4103 | .0000 | .4213 | .5657 |

***** DIRECT AND INDIRECT EFFECTS *****

Direct effect of X on Y

| Effect | SE | t | p | LLCI | ULCI |
|--------|----|---|---|------|------|
|--------|----|---|---|------|------|

```

.4935      .0368      13.4103      .0000      .4213      .5657

Indirect effect of X on Y
      Effect      Boot SE      BootLLCI      BootULCI
Eco_Inno      .3741      .0250      .3262      .4241

Normal theory tests for indirect effect
      Effect      se      Z      p
      .3741      .0317      11.7884      .0000

***** ANALYSIS NOTES AND WARNINGS *****

Number of bootstrap samples for bias corrected bootstrap confidence
intervals:
      5000

Level of confidence for all confidence intervals in output:
      95.00

----- END MATRIX -----

restore.

```

Matrix

| Notes | | |
|----------------|--------------------------------|----------------------|
| Output Created | | 20-OCT-2020 15:58:10 |
| Comments | | |
| Input | Active Dataset | DataSet1 |
| | Filter | <none> |
| | Weight | <none> |
| | Split File | <none> |
| | N of Rows in Working Data File | 856 |

Syntax

```
matrix.  
get dat/file = */variables =  
Operational_Performance Opra_Pract  
Eco_Innovation /names = vnames/missing =  
9999.  
compute ninit = nrow(dat).  
get dat/file = */variables =  
Operational_Performance Opra_Pract  
Eco_Innovation /names = vnames/missing =  
omit.  
get tmp/file = */variables =  
Operational_Performance /names =  
yname/missing = omit.  
get tmp2/file = */variables = Opra_Pract  
/names = xname/missing = omit.  
get tmp/file = */variables = Eco_Innovation  
/names = mnames/missing = omit.  
get tmp/file = */variables = w999999t z999999t  
v999999t q999999t.  
compute wname=tmp(1,1).  
do if (wname = ' ').  
compute wname = 'xxx'.  
end if.  
compute zname=tmp(1,2).  
do if (zname = ' ').  
compute zname = 'xxx'.  
end if.  
compute vname=tmp(1,3).  
do if (vname = ' ').  
compute vname = 'xxx'.  
end if.  
compute qname=tmp(1,4).  
do if (qname = ' ').  
compute qname = 'xxx'.  
end if.  
compute n = nrow(dat).  
compute p0=-.322232431088.  
compute p1 = -1.  
compute p2 = -.342242088547.  
compute p3 = -.0204231210245.
```

```

compute p4 = -.0000453642210148.
compute q0 = .0993484626060.
compute q1 = .588581570495.
compute q2 = .531103462366.
compute q3 = .103537752850.
compute q4 = .0038560700634.
compute badend = 0.
compute priorlo = -9999999.
compute priorhi = 9999999.
compute criterr = 0.
compute cluster = 0.
compute clsdmy = 0.
compute jndich = 0.
compute wvdich=0.
compute zqdich=0.
compute mod74dic=0.
compute warnrep=0.
compute contrast = trunc( 0 ).
do if (contrast > 2 or contrast < 0).
compute contrast = 0.
end if.
compute booterr = 0.
compute effsize = ( 0 = 1).
compute note = make(10,1,0).
compute notes = 1.
compute iterr = 0.
compute clsmtn = 0.
compute quantile = ( 0 = 1).
compute mmodval= 999.
compute jn = ( 0 = 1).
compute covcoeff=( 0 =1).
compute center = ( 0 = 1).
compute detail = ( 1 = 1).
compute coeffci = ( 1 = 1).
compute conf = 95.
compute longname = ( 1 =1).
compute bconoff=( 0 <> 1).
compute covmy = trunc( 0 ).
do if (covmy < 0 or covmy > 2).
compute covmy = 0.

```

```

end if.
do if (trunc(conf) >= 100 or (trunc(conf) <=
50)).
compute conf = 95.
compute note(notes,1) = 1.
compute notes = notes + 1.
end if.
do if (n < ninit).
compute nmiss = ninit-n.
compute note(notes,1) = 11.
compute notes = notes + 1.
end if.
compute errs = 0.
compute quantd = {0,0,0,0,0,0}.
compute quantc = {0,0,0,0,0,0}.
compute mcheck = 0.
compute ttt = 0.
compute plot = trunc( 0 ).
compute runerrs = make(50,1,0).
compute model = trunc( 4 ).
compute normal = 1.
compute ws= ( 0 =1).
compute olsdichm= ( 0 =1).
compute olsdichy= ( 0 =1).
compute mcx=trunc( 0 ).
compute mcm=trunc( 0 ).
do if (ws = 1).
do if (effsize = 1).
compute note(notes,1) = 19.
compute notes = notes + 1.
compute effsize=0.
end if.
do if (normal=1).
compute note(notes,1) = 16.
compute notes = notes + 1.
compute normal=0.
end if.
end if.
do if (model <> 4).
compute ws = 0.

```

```

end if.
do if (model > 0 and model < 4 and covmy <>
0).
compute covmy=0.
compute note(notes,1) = 33.
compute notes = notes + 1.
end if.
print/space=0.
do if (jn = 1 and model <> 1 and model <> 3).
compute note(notes,1) = 7.
compute notes = notes + 1.
end if.
do if ((model > 76) or (model < 1)).
compute model = 77.
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 19.
end if.
compute itprob=0.
do if (mcx < 0).
compute mcx=0.
end if.
do if (mcx > 4).
compute mcx=0.
end if.
do if (mcm < 0).
compute mcm=0.
end if.
do if (mcm > 4).
compute mcm=0.
end if.
do if (mcx > 0 and mcm > 0).
compute errs=errs+1.
compute runerrs(errs,1)=34.
compute criterr=1.
end if.
do if ((model <> 1 and model <> 4) and (mcx >
0 or mcm > 0)).
compute errs=errs+1.
compute runerrs(errs,1)=35.

```

```
compute criterr=1.  
end if.  
do if ((model = 4) and (mcm > 0)).  
compute errs=errs+1.  
compute runerrs(errs,1)=41.  
compute criterr=1.  
end if.  
do if ((mcx > 0) and jn <> 0).  
compute jn=0.  
compute note(notes,1) = 23.  
compute notes = notes + 1.  
end if.  
do if (mmodval <> 999 and mcm > 0).  
compute note(notes,1) = 24.  
compute notes = notes + 1.  
end if.  
compute toteff = 0.  
compute toteff = (( 0 = 1)*( 4 = 4 or 4 = 6)).  
compute varorder = 2.  
compute hc3 = ( 0 <> 0).  
compute cname = 'xxx'.  
compute centvar = {'xxx'}.  
compute nmeds = ncol(mnames).  
do if (longname=0).
```


compute criterr = 1.

compute errs = errs+1.

compute runerrs(errs,1) = 33.

|

|

|

end if.

compute modelm =

```
{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,6;  
1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,7;  
1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8;  
1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,9;  
1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,10;  
1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,11;  
1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,0,12;  
1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,13;  
0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,14;  
0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,15;  
0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,16;  
0,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,17;  
0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,18;  
0,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,19;  
0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,20;  
1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,21;  
1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,22;  
1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,23;  
1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,0,0,24; 1,1,1,1
```

```
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,25;
```

```
1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,26;
```

1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,27;
1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,7,28;
1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,7,29;
1,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,30;
1,1,0,1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,31;
1,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,32;
1,1,1,1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,33;
1,1,1,1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,34;
1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,35;
1,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,36;
1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,37;
1,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,38;
1,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,39;
1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,40;
1,0,0,1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,41;
1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,42;
1,0,0,1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,43;
1,0,0,1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,44;
1,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,45;
1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,46;
1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,47;
1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,48;
1,1,0,1,1,0,1,1,0,0,0,

0,0,0,0,0,0,0,0,49;
1,1,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,50;
1,1,1,1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,51;
1,1,1,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,52;
1,1,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,53;
1,1,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,54;
1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,55;
1,1,1,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,56;
1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,57;
1,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,58;
1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,59;
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,60;
1,1,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,61;
1,1,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,62;
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,0,0,0,63;
1,0,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,64;

```

1,0,0,1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,65;
1,0,0,1,0,0,0,0,0,1,0,0,1,0,0,0,0,0,66;
1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,67;
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,68;
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,69;
1,0,0,1,0,0,0,0,0,0,0,0,1,1,0,0,0,0,70;
1,0,0,1,0,0,1,0,0,1,0,0,1,1,1,0,0,0,71;
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,1,1,0,72;
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,1,1,0,0

,73; 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,74;
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,75;
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,1,0,0,76;
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,77}.
compute wm = modelm(model, 1).
compute zm = modelm(model, 2).
compute wzm = modelm(model, 3).
compute vy = modelm(model, 4).
compute qy = modelm(model, 5).
compute vqy = modelm(model, 6).
compute wy = modelm(model, 7).
compute zy = modelm(model, 8).
compute wzy = modelm(model, 9).
compute vxy = modelm(model, 10).
compute qxy = modelm(model, 11).
compute vqxy = modelm(model, 12).
compute wmy = modelm(model, 13).
compute wvmy = modelm(model, 14).
compute wvxy = modelm(model, 15).
compute zmy = modelm(model,16).
compute wzmy = modelm(model,17).
compute xmy = modelm(model,18).
compute imm=modelm(model,19).
do if (model = 74).
compute vdich=0.
end if.
do if (ncol(xname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 20.
compute criterr = 1.

```

```

end if.
do if (ncol(mnames) > 10).
compute errs = errs+1.
compute runerrs(errs,1) = 39.
compute criterr = 1.
end if.
do if (ncol(yname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 21.
compute criterr = 1.
end if.
do if (( 999 =0) and ( 999 =0) and (model=3 or
model=2) and (contrast=1)).
compute errs = errs+1.
compute runerrs(errs,1) = 31.
compute criterr = 1.
end if.
compute xlist = (wm or zm or wzm or wy or zy
or wzy or vxy or qxy or vqxy or wvxy or xmy).
compute mlist = (vy or qy or vqy or zmy or wmy
or wzmy or xmy or (model < 4)).
compute bad=0.
do if (criterr = 0).
compute werr = 0.
compute verr = 0.
compute qerr = 0.
compute zerr = 0.
compute yerr = 1.
compute xerr = 1.
compute wlist = (wm or wzm or wy or wzy or
wm or wvmy or wvmy or wvxy or wzmy).
do if (wlist = 1 and wname = 'xxx').
compute werr = 1.
compute wlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (wlist = 1 and ((wname = qname) or
(wname = vname) or (wname = zname) or
(wname = xname) or (wname = yname))).

```

```

compute werr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 12).
end if.
compute zlist = (zm or wzm or zy or wzy or
zmy or wzmy).
do if (zlist = 1 and zname = 'xxx').
compute zerr = 1.
compute zlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (zlist = 1 and ((zname = qname) or
(zname = vname) or (zname = wname) or
(zname = xname) or (zname = yname))).
compute zerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 13.
end if.
compute qlist = (qy or vqy or qxy or vqxy).
do if (qlist = 1 and qname = 'xxx').
compute qerr = 1.
compute qlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (qlist = 1 and ((qname = zname) or
(qname = vname) or (qname = wname) or
(qname = xname) or (qname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 14.
end if.
compute vlist = (vy or vqy or vxy or vqxy or
wvmy or wvxy).
do if (vlist = 1 and vname = 'xxx').
compute verr = 1.
compute vlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 7.

```

```

end if.
do if (vlist = 1 and ((vname = zname) or
(vname = qname) or (vname = wname) or
(vname = xname) or (vname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 15.
end if.
do if (wlist = 0 and wname <> 'xxx').
compute werr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 8.
end if.
do if (zlist = 0 and zname <> 'xxx').
compute zerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 9.
end if.
do if (qlist = 0 and qname <> 'xxx').
compute qerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 10.
end if.
do if (vlist = 0 and vname <> 'xxx').
compute verr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 11.
end if.
do if (hc3 = 1).
compute note(notes,1) = 3.
compute notes = notes+1.
end if.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute cons = make(n,1,1).
compute temp = (n*sscp(dat))-
(t(csum(dat))*(csum(dat))).

```



```

compute temp = temp/(n*(n-1)).
compute temp = csum(((diag(temp))=0)).
compute temp2=1.
do if (ws = 1).
compute temp2 = (n*sscp(tmp2))-
(t(csum(tmp2))*(csum(tmp2))).
compute temp2 = temp2/(n*(n-1)).
end if.
do if ((temp > 0 and ws = 0) or (temp > 0 and
ws = 1 and temp2 > 0)).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 27.
end if.
compute nmeds = ncol(mnames).
compute mcmats=ident(nmeds*2).
compute mccoef=make(nmeds*2,1,0).
compute sobel = make(nmeds,4,-999).
do if (model = 6 and nmeds > 4).
compute errs = errs+1.
compute runerrs(errs,1)=2.
end if.
do if (model < 4 and nmeds > 1).
compute errs = errs+1.
compute runerrs(errs,1) = 3.
end if.
compute nmods = (model = 74).
compute bad = 0.
compute intcnt = 1.
compute modvals = 0.
compute modvalsd = 0.
compute yintemp = {'int_1', 'int_2', 'int_3',
'int_4', 'int_5', 'int_6', 'int_7', 'int_8', 'int_9',
'int_10', 'int_11', 'int_12', 'int_13', 'int_14',
'int_15'}.
compute yintemp = {yintemp, 'int_16', 'int_17',
'int_18', 'int_19', 'int_20', 'int_21', 'int_22',
'int_23', 'int_24', 'int_25', 'int_26', 'int_27',
'int_28'}.

```

```

compute yintemp = {yintemp, 'int_29', 'int_30',
'int_31', 'int_32', 'int_33', 'int_34', 'int_35',
'int_36', 'int_37', 'int_38', 'int_39', 'int_40'}.
compute yintemp = {yintemp, 'int_41', 'int_42',
'int_43', 'int_44', 'int_45', 'int_46', 'int_47',
'int_48', 'int_49', 'int_50', 'int_51', 'int_52'}.
compute yintemp = {yintemp, 'int_53', 'int_54',
'int_55', 'int_56', 'int_57', 'int_58', 'int_59',
'int_60', 'int_61', 'int_62', 'int_63', 'int_64'}.
compute
cntname={('C1'),('C2'),('C3'),('C4'),('C5'),('C6'),('
C7'),('C8'),('C9'),('C10'),('C11'),('C12'),('C13'),('
C14'),('C15'),('C16'),('C17')}.
compute
cntname={cntname;('C18'),('C19'),('C20'),('C21
'),('C22'),('C23'),('C24'),('C25'),('C26'),('C27'),('
C28'),('C29'),('C30'),('C31')}.
compute
cntname={cntname;('C32'),('C33'),('C34'),('C35
'),('C36'),('C37'),('C38'),('C39'),('C40'),('C41'),('
C42'),('C43'),('C44'),('C45')}.
compute
cntname={cntname;('C46'),('C47'),('C48'),('C49
'),('C50'),('C51'),('C52'),('C53'),('C54'),('C55'),('
C56'),('C57'),('C58'),('C59')}.
compute
cntname={cntname;('C60'),('C61'),('C62'),('C63
'),('C64'),('C65'),('C66'),('C67'),('C68'),('C69'),('
C70'),('C71'),('C72'),('C73')}.
compute
cntname={cntname;('C74'),('C75'),('C76'),('C77
'),('C78'),('C79'),('C80'),('C81'),('C82'),('C83'),('
C84'),('C85'),('C86'),('C87')}.
compute
cntname={cntname;('C88'),('C89'),('C90'),('C91
'),('C92'),('C93'),('C94'),('C95'),('C96'),('C97'),('
C98'),('C99'),('C100'),('C101')}.
compute
cntname={cntname;('C102'),('C103'),('C104'),('
C105')}.

```

```

compute apathnam={'a path'; 'a1 path'; 'a2
path'; 'a3 path'; 'a4 path'; 'a5 path'; 'a6 path';
'a7 path'; 'a8 path'; 'a9 path'; 'a10 path'}.
compute bpathnam={'b path'; 'b1 path'; 'b2
path'; 'b3 path'; 'b4path'; 'b5path'; 'b6 path'; 'b7
path'; 'b8 path'; 'b9 path'; 'b10 path'}.
compute modvnm = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute modvnm2 = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute mlab = { ' M1 ='; ' M2 ='; ' M3 ='; '
M4 ='; ' M5 ='; ' M6 ='; ' M7 ='; ' M8 ='; '
M9 ='; ' M10 ='}.
compute m = make(n,nmeds,1).
compute mmat = make(16,nmeds,0).
compute ymat = make(8,nmeds,0).
compute deco = make(10,1,0).
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
compute iterate = abs(trunc( 10000 )).
compute converge = abs( 0.00000001 ).
compute boot = abs(trunc( 5000 )).
do if (ws=1 and 0 > 0).
compute note(notes,1)=17.
compute notes=notes+1.
end if.
compute mc=(abs(trunc( 0 ))*(1-ws)).
do if (mc > 0 and model > 5).
do if (boot = 0).
compute boot=mc.
end if.
compute mc=0.
compute note(notes,1)=12.
compute notes = notes+1.
end if.
compute adjust = 0.
do if (mc > 0 and (model > 3 and model < 6)).
compute boot=0.

```

```

compute bconoff=0.
end if.
do if (boot > 0 and mc > 0).
compute mc = 0.
end if.
compute savboot = 0.
compute saveboot = ( 0 = 1).
do if (ws=1 and saveboot=1).
compute saveboot=0.
compute note(notes,1) = 20.
compute notes = notes + 1.
compute normal=0.
end if.
do if (saveboot =1 and boot > 0 and model >
3).
compute savboot = 1.
end if.
do if (boot <> 0 or mc <> 0).
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
loop.
compute cilow = rnd(bootsz*(1-(conf/100))/2).
compute cihigh =
trunc((bootsz*(conf/100)+(bootsz*(1-
(conf/100))/2)))+1.
do if (cilow < 1 or cihigh > bootsz).
compute
bootsz=trunc((bootsz+1000)/1000)*1000.
compute adjust = 1.
end if.
end loop if (cilow gt 0 and cihigh le bootsz).
compute boot=bootsz.
do if (mc > 0).
compute mc = bootsz.
end if.
do if (boot > 0 and mc > 0 and (model > 3 and
model < 6)).
compute boot=0.

```

```

end if.
do if (adjust = 1 and boot > 0).
compute note(notes,1)=6.
compute notes = notes+1.
end if.
do if (adjust = 1 and mc > 0).
compute note(notes,1)=13.
compute notes = notes+1.
end if.
end if.
do if (model = 6 and nmeds > 1).
compute mmpaths =
make((nmeds+2),(nmeds+2),0).
do if (nmeds = 2).
compute indboot = make(boot+1, 3, 999).
else if (nmeds = 3).
compute indboot = make(boot+1, 7, 999).
else if (nmeds = 4).
compute indboot = make(boot+1, 15, 999).
end if.
compute indlbl = {'Total:','Ind1 :','Ind2 :','Ind3
:','Ind4 :','Ind5 :','Ind6 :','Ind7 :','Ind8 :','Ind9
:','Ind10:','Ind11:','Ind12:','Ind13:','Ind14:','
Ind15:'}.
compute indlbl2 = {'Ind1','Ind2','Ind3','Ind4';
'Ind5','Ind6','Ind7','Ind8','Ind9','Ind10';
'Ind11','Ind12','Ind13','Ind14','Ind15'}.
compute indces = make(boot+1, 4, 999).
end if.
do if (model < 4).
compute boot = 0.
compute cmat = make(10,1,0).
compute zmat = make(10,1,0).
end if.
compute nvarch = make(1,ncol(dat),0).
compute wmatch = 0.
compute zmatch = 0.
compute vmatch = 0.
compute qmatch = 0.
compute mmatch=0.

```

```

compute minprobe=0.
compute maxprobe=0.
loop i = 1 to ncol(vnames).
do if (vnames(:,i)=yname).
compute y = dat(:,i).
compute nvarch(1,i)=1.
compute yerr = 0.
do if ((yname = xname) or (yname = wname) or
(yname = zname) or (yname = vname) or
(yname = qname)).
compute errs = errs+1.
compute runerrs(errs,1)=17.
end if.
end if.
do if (vnames(:,i)=xname).
compute x = dat(:,i).
compute nvarch(1,i)=1.
compute xdich = 1.
compute xerr=0.
loop jj = 1 to n.
do if ((x(jj,1) <> cmin(x)) and (x(jj,1) <>
cmax(x))).
compute xdich = 0.
break.
end if.
end loop.
compute xmean = csum(x)/n.
do if (center = 1 and (model < 4 or xlist > 0)
and mcx = 0).
compute meanvec = make(n,1, xmean).
compute x = x-meanvec.
compute centvar = {centvar, xname}.
end if.
compute xmean = csum(x)/n.
compute tmp = x-(cons*xmean).
compute xsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
do if (xdich = 0).
compute quantc(1,6) = 1.
compute matx = {xmean-xsd; xmean;
xmean+xsd}.

```

```

do if ( 999 = 999 and quantile = 0).
do if ((matx(1,1) < cmin(x)) and model = 74).
compute matx(1,1)=cmin(x).
compute minprobe=1.
end if.
do if ((matx(3,1) > cmax(x)) and model = 74).
compute matx(3,1)=cmax(x).
compute maxprobe=1.
end if.
end if.
do if (quantile = 1).
compute quantd(1,6) = 1.
compute quantc(1,6) = 0.
compute tmp = x.
compute tmp(GRADE(x)) = x.
compute matx =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (xdich = 1).
compute matx = {cmin(x); cmax(x)}.
do if (model=74).
compute matx=cmin(x).
compute mod74dic=1.
end if.
end if.
do if ( 999 <> 999).
compute matx = 999.
compute quantd(1,6) = 0.
compute quantc(1,6) = 0.
end if.
end if.
do if (werr = 0 and wlist = 1).
do if (vnames(:,i)=wname).
compute werr = 0.
compute wmatch = 1.
compute w = dat(:,i).
do if (center = 1).

```

```

compute meanvec = make(n,1,csum(w)/n).
compute w = w-meanvec.
compute centvar = {centvar, wname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute wmean = csum(w)/n.
compute tmp = w-(cons*wmean).
compute wsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute wdich = 1.
loop jj = 1 to n.
do if ((w(jj,1) <> cmin(w)) and (w(jj,1) <>
cmax(w))).
compute wdich = 0.
break.
end if.
end loop.
do if (model = 3).
compute jndich=wdich.
compute jnmin=cmin(w).
compute jnmax=cmax(w).
end if.
do if (wdich = 0).
compute matw = {wmean-wsd; wmean;
wmean+wsd}.
do if ( 999 = 999 and quantile = 0).
do if (matw(1,1) < cmin(w)).
compute matw(1,1)=cmin(w).
compute minprobe=1.
end if.
do if (matw(3,1) > cmax(w)).
compute matw(3,1)=cmax(w).
compute maxprobe=1.
end if.
end if.
compute quantc(1,1) = 1.
do if (quantile = 1).
compute quantd(1,1) = 1.
compute quantc(1,1) = 0.
compute tmp = w.

```



```

compute tmp(GRADE(w)) = w.
compute matw =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (wdich = 1).
compute matw = {cmin(w); cmax(w)}.
compute wvdich=1.
compute cmaxw=cmax(w).
compute cminw=cmin(w).
end if.
do if ( 999 <> 999).
compute matw = 999.
compute quantd(1,1) = 0.
compute quantc(1,1) = 0.
end if.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm(1,1)=wname.
compute modmatp(1,1) = 1.
end if.
end if.
do if (zerr = 0 and zlist = 1).
do if (vnames(:,i)=zname).
compute zerr = 0.
compute zmatch = 1.
compute z = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(z)/n).
compute z = z-meanvec.
compute centvar = {centvar, zname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute zmean = csum(z)/n.
compute tmp = z-(cons*zmean).
compute zsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute zdich = 1.

```

```

loop jj = 1 to n.
do if ((z(jj,1) <> cmin(z)) and (z(jj,1) <>
cmax(z))).
compute zdich = 0.
break.
end if.
end loop.
do if (zdich = 0).
compute matz = {zmean-zsd; zmean;
zmean+zsd}.
do if ( 999 = 999 and quantile = 0).
do if (matz(1,1) < cmin(z)).
compute matz(1,1)=cmin(z).
compute minprobe=1.
end if.
do if (matz(3,1) > cmax(z)).
compute matz(3,1)=cmax(z).
compute maxprobe=1.
end if.
end if.
compute quantc(1,2) = 1.
do if (quantile = 1).
compute quantd(1,2) = 1.
compute quantc(1,2) = 0.
compute tmp = z.
compute tmp(GRADE(z)) = z.
compute matz = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (zdich = 1).
compute matz = {cmin(z); cmax(z)}.
compute cmaxz=cmax(z).
compute cminz=cmin(z).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matz = 999.
compute quantd(1,2) = 0.

```

```

compute quantc(1,2) = 0.
end if.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm(1,2)=zname.
compute modmatp(1,2) = 1.
end if.
end if.
do if (verr = 0 and vlist = 1).
do if (vnames(:,i)=vname).
compute verr = 0.
compute vmatch = 1.
compute v = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(v)/n).
compute v = v-meanvec.
compute centvar = {centvar, vname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute vmean = csum(v)/n.
compute tmp = v-(cons*vmean).
compute vsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute vdich = 1.
loop jj = 1 to n.
do if ((v(jj,1) <> cmin(v)) and (v(jj,1) <>
cmax(v))).
compute vdich = 0.
break.
end if.
end loop.
do if (vdich = 0).
compute matv = {vmean-vsd; vmean;
vmean+vsd}.
do if ( 999 = 999 and quantile = 0).
do if (matv(1,1) < cmin(v)).
compute matv(1,1)=cmin(v).
compute minprobe=1.
end if.
do if (matv(3,1) > cmax(v)).

```

```

compute matv(3,1)=cmax(v).
compute maxprobe=1.
end if.
end if.
compute quantc(1,3) = 1.
do if (quantile = 1).
compute quantd(1,3) = 1.
compute quantc(1,3) = 0.
compute tmp = v.
compute tmp(GRADE(v)) = v.
compute matv = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (vdich = 1).
compute matv = {cmin(v); cmax(v)}.
compute wvdich=1.
compute cmaxv=cmax(v).
compute cminv=cmin(v).
end if.
do if ( 999 <> 999).
compute matv = 999.
compute quantd(1,3) = 0.
compute quantc(1,3) = 0.
end if.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm(1,3)=vname.
compute modmatp(1,3) = 1.
end if.
end if.
do if (qerr = 0 and qlist = 1).
do if (vnames(:,i)=qname).
compute qerr = 0.
compute qmatch = 1.
compute q = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(q)/n).
compute q = q-meanvec.

```

```

compute centvar = {centvar, qname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute qmean = csum(q)/n.
compute tmp = q-(cons*qmean).
compute qsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute qdich = 1.
loop jj = 1 to n.
do if ((q(jj,1) <> cmin(q)) and (q(jj,1) <>
cmax(q))).
compute qdich = 0.
break.
end if.
end loop.
do if (qdich = 0).
compute matq = {qmean-qsd; qmean;
qmean+qsd}.
do if ( 999 = 999 and quantile = 0).
do if (matq(1,1) < cmin(q)).
compute matq(1,1)=cmin(q).
compute minprobe=1.
end if.
do if (matq(3,1) > cmax(q)).
compute matq(3,1)=cmax(q).
compute maxprobe=1.
end if.
end if.
compute quantc(1,4) = 1.
do if (quantile = 1).
compute quantd(1,4) = 1.
compute quantc(1,4) = 0.
compute tmp = q.
compute tmp(GRADE(q)) = q.
compute matq = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (qdich = 1).

```

```

compute matq = {cmin(q); cmax(q)}.
compute cmaxq=cmax(q).
compute cminq=cmin(q).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matq = 999.
compute quantd(1,4) = 0.
compute quantc(1,4) = 0.
end if.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm(1,4)=qname.
compute modmatp(1,4) = 1.
end if.
end if.
do if (vnames(:,i)=cname).
compute cld = dat(:,i).
compute cvname = vnames(:,i).
compute nvarch(1,i)=1.
compute clsmatch = 1.
end if.
loop j = 1 to ncol(mnames).
do if (vnames(:,i)=mnames(1,j)).
compute mmatch = mmatch + 1.
compute m(:,j)=dat(:,i).
do if (center = 1 and nvarch(1,i) = 0 and mlist >
0 and mcm = 0).
compute meanvec = make(n,1,csum(m(:,j))/n).
compute m(:,j) = m(:,j)-meanvec.
compute centvar = {centvar, mnames(1,j)}.
end if.
compute nvarch(1,i)=1.
compute dichm = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute dichm = 0.
break.
end if.

```

```

end loop.
do if ((dichm = 1) and (olsdichm = 1)).
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichm = 1 and model > 3 and mcheck = 0
and oldsdichm = 0).
compute errs= errs+1.
compute runerrs(errs,1) = 1.
compute mcheck = 1.
end if.
do if ((model <= 3) and (ncol(mnames)=1)).
compute nmods = nmods + 1.
compute mmean = csum(m(:,j))/n.
compute tmp = m(:,j)-(cons*mmean).
compute msd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute mdich = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute mdich = 0.
break.
end if.
end loop.
do if (model = 1).
compute jndich=mdich.
compute jnmin=cmin(m(:,j)).
compute jnmax=cmax(m(:,j)).
end if.
do if (mdich = 0).
compute matm = {mmean-msd; mmean;
mmean+msd}.
do if ( 999 = 999 and quantile = 0).
do if (matm(1,1) < cmin(m(:,j))).
compute matm(1,1)=cmin(m(:,j)).
compute minprobe=1.
end if.
do if (matm(3,1) > cmax(m(:,j))).
compute matm(3,1)=cmax(m(:,j)).
compute maxprobe=1.

```

```

end if.
end if.
compute quantc(1,5) = 1.
do if (quantile = 1).
compute quantd(1,5) = 1.
compute quantc(1,5) = 0.
compute tmp = m(:,j).
compute tmp(GRADE(m(:,j))) = m(:,j).
compute matm = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (mdich = 1).
compute matm = {cmin(m); cmax(m)}.
end if.
do if ( 999 <> 999).
compute matm = 999.
compute quantd(1,5) = 0.
compute quantc(1,5) = 0.
end if.
compute modmatv(1,5)=nrow(matm).
compute modmat((1:nrow(matm)), 5) = matm.
compute modvnm(1,5)=mnames(1,j).
compute modmatp(1,5) = 1.
end if.
end if.
end loop.
end loop.
do if (minprobe=1).
compute note(notes,1) = 14.
compute notes = notes + 1.
end if.
do if (maxprobe=1).
compute note(notes,1) = 15.
compute notes = notes + 1.
end if.
do if (cname <> 'xxx' and clsmth = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 23.

```



```

end if.
do if (cname <> 'xxx').
do if ((cname = zname) or (cname = vname)
or (cname = wname) or (cname = xname) or
(cname = yname) or (cname = qname)).
compute errs = errs+1.
compute runerrs(errs,1) = 24.
end if.
end if.
do if (wlist = 1 and werr = 0 and wmatch = 0).
compute werr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (zlist = 1 and zerr = 0 and zmatch = 0).
compute zerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (qlist = 1 and qerr = 0 and qmatch = 0).
compute qerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (vlist = 1 and verr = 0 and vmatch = 0).
compute verr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (yerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 16.
end if.
do if (xerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 32.
end if.
do if (model = 6 and nmeds < 2).
compute errs = errs+1.
compute runerrs(errs,1) = 18.

```

```

end if.
do if (mmatch < ncol(mnames)).
compute errs=errs+1.
compute runerrs(errs,1) = 25.
end if.
do if (cname <> 'xxx').
compute cld = design(cld).
compute cluster = ncol(cld).
compute cld = cld(:,2:ncol(cld)).
compute clsdmy = ncol(cld).
do if (clsdmy > 19).
compute errs = errs+1.
compute runerrs(errs,1) = 26.
end if.
do if (mcx > 0 or mcm > 0).
compute errs = errs+1.
compute runerrs(errs,1) = 38.
end if.
end if.
compute dichy = 1.
loop jj = 1 to n.
do if ((y(jj,1) <> cmin(y)) and (y(jj,1) <>
cmax(y))).
compute dichy = 0.
break.
end if.
end loop.
do if ((dichy = 1) and (olsdichy = 1)).
compute dichy=0.
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichy = 1).
do if (toteff=1).
compute toteff=0.
compute note(notes,1) = 31.
compute notes = notes + 1.
end if.
compute jncrit=xp2*xp2.
do if (ws = 1).

```

```

compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 28.
end if.
end if.
compute ncovs = ncol(dat)-rsum(nvarch).
do if (effsize = 1 and covmy <> 0 and ncovs >
0 and model > 3 and model < 7).
compute note(notes,1) = 22.
compute notes = notes + 1.
compute effsize = 0.
end if.
do if (ws = 1 and ncovs < nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 29.
end if.
do if (ws = 1 and ncovs > nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 30.
end if.
end if.
do if (errs = 0).
do if (rsum(quantd) > 0).
compute note(notes,1) = 4.
compute notes = notes+1.
end if.
do if (rsum(quantc) > 0).
compute note(notes,1) = 5.
compute notes = notes+1.
end if.
do if (ncovs > 0).
compute c = make(n,ncovs,0).
compute cnames = {'x'}.
compute j = 1.
loop i = 1 to ncol(vnames).
do if (nvarch(1,i)) = 0.
compute c(:,j) = dat(:,i).
compute nvarch(1,i)=1.

```

```

compute j=j+1.
compute cnames = {cnames, vnames(:,i)}.
end if.
end loop.
compute cnames = cnames(1,2:ncol(cnames)).
do if (ws = 1).
compute covmean=csum(c)/n.
loop i = 1 to ncovs.
compute meanvec = make(n,1, covmean(1,i)).
compute c(:,i) = c(:,i)-meanvec.
end loop.
compute centvar = {centvar, cnames}.
compute x=c(:,1:ncovs).
do if (ncovs=nmeds).
compute ncovs=0.
end if.
end if.
end if.
compute names = {yname, xname, mnames,
wname, zname, vname, qname}.
do if (ncovs > 0).
compute names = {names, cnames}.
end if.
do if (dichy = 1 and effsize = 1).
compute note(notes,1) = 2.
compute notes = notes+1.
end if.
do if (model > 3 and model < 6).
compute indeff=make(nmeds,1,0).
compute indboot=make(boot+1,nmeds,999).
do if (mc > 0).
compute indboot=make(mc+1,nmeds,999).
end if.
do if (effsize = 1 and dichy = 0).
compute rmeff=make(boot+1,nmeds+1,999).
compute abpseff=make(boot+1,nmeds+1,999).
compute abcseff=make(boot+1,nmeds+1,999).
compute pmeff=make(boot+1,nmeds+1,999).
compute r245 = make(boot+1,1,999).
compute kappa2 = make(boot+1,1,999).

```

```

end if.
end if.
do if (model = 6 and effsize = 1 and dichy = 0).
compute
rmeff=make(boot+1,ncol(indboot),999).
compute
abpseff=make(boot+1,ncol(indboot),999).
compute
abcseff=make(boot+1,ncol(indboot),999).
compute
pmeff=make(boot+1,ncol(indboot),999).
end if.
do if (nmods > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm(1,tmp) = modvnm(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:nmods).
compute modvnm=modvnm(:,1:nmods).
compute modmatv=modmatv(:,1:nmods).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvals =
make((modmatv(1,1)*modprod(1,1)),
nmods,0).
loop i = 1 to nmods.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvals)).
loop j = 1 to modmatv(1,i).

```

```

compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvals(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
do if (model = 74).
compute modvals=matx.
compute modvnm=xname.
end if.
compute vmat = make(8,nrow(modvals),0).
compute vmat(1,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute vmat(5,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute indeff=make(nrow(modvals),1,0).
do if (model <> 5).
compute indboot = make(((boot+1)*nmeds),
nrow(modvals), -99999999).
compute indbootp =
make(boot+1,(nmeds*nmods), -99999999).
do if (imm > 4 and imm < 8).
do if (imm = 7).
compute immmv1=matw.
compute immmv2=matv.
end if.
do if (imm = 5).
compute immmv1=matw.
compute immmv2=matz.
end if.
do if (imm = 6).
compute immmv1=matv.
compute immmv2=matq.
end if.
compute
indboop1=make((boot+1),(nmeds*(nrow(immm
v2))),-99999999).

```

```

compute
indboop2=make((boot+1),(nmeds*(nrow(immmv1))),-9999999).
compute onesmv1=make(1,nrow(immmv1),1).
compute onesmv2=make(1,nrow(immmv2),1).
compute
nimvs=nrow(immmv1)+nrow(immmv2).
compute cmmval=0.
compute cmmlbs1={'xx'}.
compute cmmlbs2={'xx'}.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv2}.
compute
cmmlbt=make(nrow(immmv2),1,mnames(1,i)).
compute cmmlbs1={cmmlbs1;cmmlbt}.
end loop.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv1}.
compute
cmmlbt=make(nrow(immmv1),1,mnames(1,i)).
compute cmmlbs2={cmmlbs2;cmmlbt}.
end loop.
compute cmmval=cmmval(2:nrow(cmmval),:).
end if.
end if.
end if.
do if (nmods> 0).
loop i = 1 to ncol(modvals).
do if (modvnm(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm(1,i)=qname).
compute qcol = i.
end if.

```

```

end loop.
end if.
do if (dichy = 1).
compute omx = cmax(y).
compute omn = cmin(y).
compute y = (y = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute data = {cons,y,m,x}.
compute datamed = data.
compute datayed = data.
compute datanm = {'constant'; yname;
t(mnames); xname}.
compute datanmm = {'constant'; yname;
t(mnames); xname}.
compute datanmy = {'constant'; yname;
t(mnames); xname}.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
do if (model < 4 and errs = 0).
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' ', ' '}.
compute datayed = {datayed, x&*m}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(1,i) = 1.
compute vmat(2,i) = modvals(i,1).
end loop.
compute mmat = make(16,nmeds,1).
end if.
do if (model = 2 or model = 3).
compute int1 = x&*w.
compute datayed = {datayed, w, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).

```



```

compute vmat(2,i) = modvals(i,2).
compute vmat(3,i) = modvals(i,1).
compute vmat(4,i) = modvals(i,1)*modvals(i,2).
end loop.
end if.
do if (model = 3).
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*m.
compute int2 = x&*w&*m.
compute datayed = {datayed, int1, int2}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' X', wname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (model = 4 or model = 5).
compute vmat = make(8,1,1).
end if.
compute yintkey2 = yintkey.
do if (wm = 1).
compute int1 = x&*w.
compute datamed = {datamed, w,int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmm = {datanmm; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,wcol).
end loop.
do if (zm = 1).
compute int1 = x&*z.
compute datamed = {datamed, z, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.

```

```

compute datanmm = {datanmm; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(3,i) = modvals(i,zcol).
end loop.
end if.
do if (wzm = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*z.
compute int2 = x&*w&*z.
compute datamed = {datamed, int1, int2}.
loop i = 1 to nrow(modvals).
compute vmat(4,i) =
modvals(i,wcol)*modvals(i,zcol).
end loop.
end if.
end if.
compute mdatacol=ncol(datamed).
compute mintkey = yintkey.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
compute medints = intcnt-1.
do if (vy = 1 or xmy = 1).
compute mp = 1.
do if (xmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,1).
end loop.
end if.
do if (vy = 1).
compute datayed = {datayed, v}.

```

```

compute datanmy = {datanmy; vname}.
compute mmods = 1.
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,vcol).
end loop.
do if (qy = 1).
compute mp = 2.
compute datayed = {datayed,q}.
compute datanmy = {datanmy; qname}.
compute mmods = 2.
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,qcol).
end loop.
end if.
do if (vqy = 1).
compute mp = 3.
compute datayed = {datayed, v&*q}.
compute mmods = 3.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,vcol)*modvals(i,qcol).
end loop.
end if.
end if.
compute mints = make(n,(nmeds*mp),0).
loop i = 0 to (nmeds-1).
do if (i = 0 and vqy = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
vname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.

```

```

end if.
do if (xmy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))&*x.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', xname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (qy = 1).
compute mints(:,((i*mp)+2))=m(:,(i+1))&*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqy = 1).
compute mints(:,((i*mp)+3))=m(:,(i+1))&*v&*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end loop.
compute datayed = {datayed, mints}.
end if.
compute mp = 1.
do if (wvmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,wcol)*modvals(i,vcol).
end loop.
end if.
compute mints2 = make(n,(nmeds*mp),0).
do if (wmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,wcol).

```

```

end loop.
do if (wy = 0 and model > 3).
compute datayed = {datayed, w}.
compute datanmy = {datanmy; wname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wvmy = 1).
compute datayed = {datayed,w&*v}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints2(:,((i*mp)+1))= m(:,(i+1))&*w.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wvmy = 1).
compute mints2(:,((i*mp)+2))=
m(:,(i+1))&*w&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints2}.
end if.
compute mp = 1.
do if (zmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,zcol).
end loop.
do if (wzmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).

```

```

compute vmat(8,i) =
modvals(i,zcol)&*modvals(i,wcol).
end loop.
end if.
end if.
do if (zmy = 1).
compute mints3 = make(n,(nmeds*mp),0).
do if (zy = 0).
compute datayed = {datayed, z}.
compute datanmy = {datanmy; zname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wzmy = 1 and wzy = 0).
compute datayed = {datayed,w&*z}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints3(:,((i*mp)+1))= m(:,(i+1))&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wzmy = 1).
compute mints3(:,((i*mp)+2))=
m(:,(i+1))&*w&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', zname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints3}.
end if.
compute decoc = 1.
compute modmat = make(5,5,999).

```

```

compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
do if (wy = 1 and model > 3).
compute datayed = {datayed, w, x&*w}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm2(1,1)=wname.
compute modmatp(1,1) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ',', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (zy = 1).
compute datayed = {datayed,z,x&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm2(1,2)=zname.
compute modmatp(1,2) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ',', ' '}.
compute datanmy = {datanmy; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (wzy = 1).
compute datayed = {datayed,w&*z,x&*w&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ',', ' '}.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.

```

```

compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vxy = 1).
compute datayed = {datayed, x&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm2(1,3)=vname.
compute modmatp(1,3) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (qxy = 1).
compute datayed = {datayed, x&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm2(1,4)=qname.
compute modmatp(1,4) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqxy = 1).
compute datayed = {datayed, x&*v&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, 'X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.

```



```

compute intcnt = intcnt+1.
end if.
end if.
end if.
do if (wvxy = 1).
compute datayed = {datayed, x*w&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, 'X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute modvalsd = 0.
compute ttt = rsum(modmatp).
do if (rsum(modmatp) > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm2(1,tmp) = modvnm2(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:ttt).
compute modvnm2=modvnm2(:,1:ttt).
compute modmatv=modmatv(:,1:ttt).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvalsd =
make((modmatv(1,1)*modprod(1,1)), ttt,0).
loop i = 1 to ttt.
compute strt = 1.

```

```

compute fnsh=0.
loop if (fnsh < nrow(modvalsd)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvalsd(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
end if.
do if (ttt > 0).
loop i = 1 to ncol(modvalsd).
do if (modvnm2(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm2(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm2(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm2(1,i)=qname).
compute qcol = i.
end if.
end loop.
compute directv = make(nrow(modvalsd),1,1).
do if (wy = 1).
compute directv ={directv, modvalsd(:,wcol)}.
end if.
do if (zy = 1).
compute directv ={directv, modvalsd(:,zcol)}.
end if.
do if (wzy = 1).
compute directv ={directv,
(modvalsd(:,wcol)&*modvalsd(:,zcol))}.
end if.
do if (vxy = 1).
compute directv ={directv, modvalsd(:,vcol)}.

```

```

end if.
do if (qxy = 1).
compute directv={directv, modvalsd(:,qcol)}.
end if.
do if (vqxy = 1).
compute directv={directv,
(modvalsd(:,vcol)&*modvalsd(:,qcol))}.
end if.
do if (wvxy = 1).
compute directv={directv,
(modvalsd(:,vcol)&*modvalsd(:,wcol))}.
end if.
end if.
compute ydatacol=ncol(datayed).
do if (ncovs > 0).
do if (covmy <> 2).
compute datamed = {datamed,c}.
end if.
do if (covmy <> 1).
compute datayed = {datayed,c}.
end if.
do if (covmy = 1 and model = 4 and mcx > 0).
compute datayed = {datayed,c}.
end if.
compute covmeans = csum(c)/n.
end if.
do if (cluster > 0).
compute datamed = {datamed,cld}.
compute datayed = {datayed, cld}.
compute cldmeans = csum(cld)/n.
end if.
compute mst = 3.
compute mnd = mst+nmeds-1.
compute ydatacol=ncol(datayed).
compute mdatacol=ncol(datamed).
do if (ncovs > 0).
compute datanmy = {datanmy; t(cnames)}.
do if (model > 3).
compute datanmm = {datanmm; t(cnames)}.
end if.

```

```

end if.
compute datanmy = {'constant';
datanmy(3:nrow(datanmy),1)}.
do if (model > 3).
compute datanmm = {'constant';
datanmm(3:nrow(datanmm),1)}.
end if.
compute amm = make(2,1,0).
compute abmm = make(2,1,0).
compute mnv = csum(datayed(:,2)/n).
compute mnv = make(n,1,mnv).
compute ssty = csum((datayed(:,2)-mnv)**2).
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
do if (ws=1).
compute stddevm=diag(sigma).
compute
stddevm=sqrt(stddevm(3:(nmeds+2),1)).
end if.
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute r2xy =
(sigma(2,(3+nmeds))/(stddevy*stddevx))**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))**2.
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).
do if (model = 4 and nmeds = 1 and cluster = 0
and ncovs = 0).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).

```

```

compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
compute datatm = datamed.
compute dataty = datayed.
compute mdlrms2 = { '4' ; yname; xname}.
compute mdlrms = {'Model ='; ' Y ='; ' X ='
}.
do if (ws = 1).
compute mdlrms2 = { '4' ; yname}.
compute mdlrms = {'Model ='; ' Y ='}.
end if.
loop i = 1 to ncol(mnames).
compute mdlrms2 = {mdlrms2; mnames(1,i)}.
do if (i = 1 and ncol(mnames) = 1).
compute mdlrms = {mdlrms; ' M ='}.
else.
compute mdlrms = {mdlrms; mlab(i,1)}.
end if.
end loop.
do if (wname <> 'xxx').
compute mdlrms2 = {mdlrms2; wname}.
compute mdlrms = {mdlrms; ' W ='}.
end if.
do if (zname <> 'xxx').
compute mdlrms2 = {mdlrms2; zname}.
compute mdlrms = {mdlrms; ' Z ='}.

```

```

end if.
do if (vname <> 'xxx').
compute mdlms2 = {mdlms2; vname}.
compute mdlms = {mdlms; '  V = '}.
end if.
do if (qname <> 'xxx').
compute mdlms2 = {mdlms2; qname}.
compute mdlms = {mdlms; '  Q = '}.
end if.
do if (jn = 1 and model = 1 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
do if (jn = 1 and model = 3 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
compute mdpbe=0.
compute medte2=0.
do if ( 4 =4 and mcx > 0).
compute medte2=1.
end if.
do if (((mcx > 0) or (mcm > 0)) AND model =
1).
compute mdpbe=1.
compute dd={datayed(:,4),datayed(:,3)}.
do if (ncovs > 0).
compute dd={datayed(:,6:(5+ncovs)),dd}.
end if.
compute dd={datayed(:,2),dd}.
end if.
compute yes = ((nrow(modvals)=1) and
(contrast = 1) and (model = 3 or model = 2)).
compute yes2 = ((nrow(modvals)=1) and
(contrast = 1) and (model = 2)).
print/title = '***** PROCESS Procedure
for SPSS Release 2.16.3 *****'.
print/title = '      Written by Andrew F. Hayes,
Ph.D.      www.afhayes.com'.

```

```

print/title = ' Documentation available in
Hayes (2013).
www.guilford.com/p/hayes3'/space=0.
print mdlrms2/title =
*****
*****'/rnames = mdlrms/format =
a8.
do if (ncovs > 0).
print cnames/title = 'Statistical Controls: '/rlabels
= 'CONTROL='/format a8.
end if.
print n/title = 'Sample size'/format F10.0.
do if (cluster > 0).
print cluster/rnames = cvname/title =
'Clustering Variable and Number of Clusters'.
end if.
do if (model > 3 and ( 'random' <> 'random')).
compute seedt= 'random'.
print seedt/title='Custom'/format=A12/rlabel =
'Seed:'.
end if.
do if (mdpbe=0 and medte2 = 0).
loop bt = 1 to (boot+1).
do if (bt = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef = 0.
do if (bt > 1).
loop.
compute v=trunc(uniform(n,1)*n)+1.
compute datayed = dataty(v,:).
do if (ws=1).
compute
bmeans=csum(datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed)))/n.
compute cntmat=make(n,nmeds,1).
compute bmeans2=bmeans.
do if (nmeds > 1).
compute bmeans2=mdiag(bmeans).

```

```

end if.
compute cntmat=cntmat*bmeans2.
compute datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed))=datayed(:,(ncol(data
yed)-nmeds+1):ncol(datayed))-cntmat.
end if.
compute detcheck=det(t(datayed)*datayed).
compute rk=(detcheck <> 0).
do if (model > 3).
compute datamed = datatm(v,:).
compute detcheck=det(t(datamed)*datamed).
do if (rk=1).
compute rk=(detcheck <> 0).
end if.
end if.
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
compute temp=diag(sigma).
compute bad = bad+(1-rk).
compute false = 1.
end loop if (rk = 1).
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nme
ds)).
do if (model = 4 and nmeds = 1 and ncovs = 0
and cluster = 0).
compute r2xy =
(sigma(2,4)/(stddevy*stddevx))&**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))&**2.
compute sstot = sigma(2,2)*(n-1).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).

```



```

compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
end if.
do if (model > 3).
loop im = 1 to nmeds.
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant';
datanmm((2+nmeds):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
do if (model = 6).
do if (im = 1).
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
end if.
do if (im > 1).

```

```

compute xm={cons, datamed(:,3:(im+1)),
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant'; datanmm(2:im,1);
datanmm((mnd):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
compute
mmpaths((im+1),(2:im))=t(coeff(2:im,1)).
end if.
end if.
do if (ws = 1).
compute
coeff(1,1)=csum(datamed(:,(2+im)))/n.
end if.
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
do if (bt = 1).
compute resid=datamed(:,(2+im))-xm*coeff.
compute sse =cssq(resid).
compute mse = sse/(n-ncol(xm)).
compute mnv = csum(data(:,(2+im)))/n.
compute mnv = make(n,1,mnv).
compute sstm = csum((data(:,(2+im))-
mnv)&**2).
compute k3 = nrow(coeff).
do if (hc3 = 1).
compute h = xm(:,1).
loop i3=1 to n.
compute h(i3,1)= xm(i3,:)*invXtX*t(xm(i3,:)).
end loop.
loop i3=1 to k3.
compute xm(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xm(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xm(:,i3) = sqrt(mse)&*xm(:,i3).

```

```

end loop.
end if.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute hccov=invXtX*t(xm)*xm*invXtX.
compute mcmats(im,im)=hccov(2,2).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*hccov*lmat)*((t(lmat)
)*coeff))/dfnum.
compute coeff = coeff(1:(nrow(coeff)-
clsdmy),1).
compute mcoeff(im,1)=coeff(2,1).
compute standerr =
sqrt(diag(invXtX*t(xm)*xm*invXtX)).
compute standerr = standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1).
compute standerr(1,1)=stddevm(im,1)/sqrt(n).
end if.
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-
ncol(xm)))).
compute temp=(n-ncol(xm)).
do if (ws=1).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute temp=(n-1).
end if.
compute xd = abs(xp2).
compute temp = (temp* (exp((temp-
(5/6))*((xd/(temp-(2/3)+(.11/temp)))*(xd/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = coeff-
sqrt(abs(temp))*standerr.
compute temp2 =
coeff+sqrt(abs(temp))*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (ws=1).

```

```

compute op=op(1,:).
compute xnmn =
apathnam(((nmeds>1)+1+(im-1)),1).
end if.
compute sobel(im,1) = coeff(2,1).
compute sobel(im,2)=standerr(2,1).
compute temp = mnames(1,im).
compute r2full = 1-(sse/sstm).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute summ = {sqrt(r2full), r2full, mse,
Fratio, dfnum, dfden, pfr}.
do if (detail = 1).
print temp/title =
*****
*****/rlabels = 'Outcome:'/format
= A8.
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels 'R',
'R-sq', 'MSE','F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
print op/title = 'Model'/rnames = xnmn/clabels
'coeff', 'se', 't', 'p', 'LLCI', 'ULCI'/format = F10.4.
do if (covcoeff=1 and ws <> 1).
compute
hccovtmp=hccov(1:nrow(op),1:nrow(op)).
compute cnamestp=t(xnmn).
print hccovtmp/title='Covariance matrix of
regression parameter
estimates'/rnames=xnmn/cnames=cnamestp/f
ormat= F10.4.
end if.
do if (nmods > 0 and nrow(mintkey) > 1).
print mintkey/title = 'Product terms key:'/format
= A8.
end if.
do if (ws = 1).
compute wsdf=n-1.

```

```

print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
end if.
end if.
compute ymat(1,im) = coeff(2,1).
do if (ws = 1).
compute ymat(1,im)=coeff(1,1).
end if.
do if (wm = 1).
compute ymat(2,im) = coeff(4,1).
do if (zm = 1).
compute ymat(3,im) = coeff(6,1).
do if (wzm = 1).
compute ymat(4, im) = coeff(8,1).
end if.
end if.
end if.
do if (model = 6).
compute mmpaths((im+1),1)=coeff((im+1),1).
end if.
end loop.
end if.
loop totlp = 1 to (1+(toteff*(bt = 1))).
do if (toteff = 1 and totlp =2).
compute xy =
{cons,datayed(:,(3+nmeds):ydatacol)}.
else.
compute xy={cons, datayed(:,3:ydatacol)}.
end if.
do if (dichy = 1).
compute pt2 =
make(nrow(datayed(:,2)),1,(csum(datayed(:,2))
/n)).
compute LL3 = datayed(:,2)*ln(pt2)+(1-
datayed(:,2))*ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xy),1,0).
compute LL1 = 0.
compute xy22=xy.

```

```

loop jjj = 1 to iterate.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.
compute coeff =
bt1+inv(t(xy22)*xy)*t(xy)*(datayed(:,2)-pt1).
compute pt1 = 1/(1+exp(-(xy*coeff))).
compute itprob = csum((pt1 <
.00000000000001) or (pt1 >
.99999999999999)).
do if (itprob = 0).
compute LL = datayed(:,2)*ln(pt1)+(1-
datayed(:,2))*ln(1-pt1).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
break.
end if.
compute bt1 = coeff.
compute LL1 = LL2.
end loop.
do if (jjj >= iterate and iterr = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
compute iterr = 1.
end if.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.
compute covmat = inv(t(xy22)*xy).
release xy22.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
end if.
do if (dichy = 0).
compute invXtX = inv(t(xy)*xy).
compute coeff =invXtX*t(xy)*datayed(:,2).

```

```

do if (effsize=1 and model > 3 and model < 7
and (ncovs > 0 or clsdmy > 0)).
compute xysdy =
{cons,datayed(:,(4+nmeds):ydatacol)}.
compute coeffsd
=inv(t(xysdy)*xysdy)*t(xysdy)*datayed(:,2).
compute resid=datayed(:,2)-(xysdy*coeffsd).
compute sdy cov =sqrt(csum(resid*&resid)/(n-
ncol(xysdy))).
compute xvaron=datayed(:,(3+nmeds)).
compute
coeffx2=inv(t(xysdy)*xysdy)*t(xysdy)*xvaron.
compute residx2=xvaron-(xysdy*coeffx2).
compute sdx cov
=sqrt(csum(residx2*&residx2)/(n-ncol(xysdy))).
end if.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4 and bt > 1).
compute resid=datayed(:,2)-xy*coeff.
compute sse = cssq(resid).
compute r2full = 1-(sse/sstot).
end if.
do if (bt =1).
compute resid=data(:,2)-xy*coeff.
compute k3 = nrow(coeff).
compute sse = cssq(resid).
compute mse = sse/(n-ncol(xy)).
compute
hcnott=coeff&/(sqrt(diag(mse*invXtX))).
do if (hc3 = 1).
compute h = xy(:,1).
loop i3=1 to n.
compute h(i3,1)= xy(i3,:)*invXtX*t(xy(i3,:)).
end loop.
loop i3=1 to k3.

```

```

compute xy(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xy(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xy(:,i3) = sqrt(mse)&*xy(:,i3).
end loop.
end if.
compute covmat = (invXtX*t(xy)*xy*invXtX).
end if.
end if.
do if (bt = 1).
do if (model = 2).
compute xy2={cons, datayed(:,3:ydatacol)}.
compute temp = ncol(xy2).
do if (temp > 6).
compute xy3=xy2(:,7:temp)}.
end if.
compute xy2={xy2(:,1:3), xy2(:,5)}.
do if (temp > 6).
compute xy2={xy2, xy3}.
release xy3.
end if.
compute invXtX = inv(t(xy2)*xy2).
compute coeff2=invXtX*t(xy2)*datayed(:,2).
compute ssem2=cssq(datayed(:,2)-xy2*coeff2).
release xy2.
end if.
compute standerr = sqrt(diag(covmat)).
do if (totlp =1).
compute
mcmats((nmeds+1):ncol(mcmats),(nmeds+1):n
col(mcmats))=covmat(2:(1+nmeds),2:(1+nmed
s)).
end if.
compute standerr=standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1 and totp=2).
compute standerr(1,1)=stddevy/sqrt(n).

```



```

end if.
compute coeffplt = coeff.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*covmat*lmat)*((t(lmat)*coeff))/dfnum).
compute coeff=coeff(1:(nrow(coeff)-clsdmy),1).
do if (totlp=1).
compute
mccoeff((nmeds+1):nrow(mccoeff))=coeff(2:(1+nmeds),1).
end if.
compute bbbb=coeff(2,1).
do if totp = 1.
compute deco(1,1)=2+nmeds.
compute deco = deco(1:decoc,1).
compute covdirt = make((nrow(covmat)-clsdmy),(ncol(covmat)-clsdmy),0).
compute covdirt = covmat(deco,:).
compute covdir =
make(nrow(covdirt),ncol(covdirt),0).
compute covdir = covdirt(:,t(deco)).
compute deco=coeff(deco,1).
do if (ttt > 0).
compute sedir =
sqrt(diag(directv*covdir*t(directv))).
compute directv = directv*deco.
end if.
compute sobel(:,3)=coeff(2:(1+nmeds),1).
compute sobel(:,4) = standerr(2:(1+nmeds),1).
compute sobel2 = sobel*sobel.
do if (varorder <> 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)*sobel2(:,4)+sobel2(:,3)*sobel2(:,2)).
end if.
do if (varorder = 2).

```

```

compute sobel(:,2) =
sqrt(sobel2(:,1)&*sobel2(:,4)+sobel2(:,3)&*sob
el2(:,2)+sobel2(:,2)&*sobel2(:,4)).
end if.
compute sobel(:,1) = sobel(:,1)&*sobel(:,3).
compute sobel(:,3) = sobel(:,1)&/sobel(:,2).
compute sobel(:,4) = 2*(1-
cdfnorm(abs(sobel(:,3)))).
end if.
do if (dichy = 0).
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
do if (ws= 1 and totlp=2).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute dfden=n-1.
end if.
compute cnms = {'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute op = {coeff, standerr, tratio, p}.
end if.
do if (dichy = 1).
compute tratio = (coeff&/standerr).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute wald = tratio&*tratio.
compute cnms = {'coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute temp=coeff-abs(xp2)*standerr.
compute op = {coeff, standerr, tratio, p, temp}.
compute temp=coeff+abs(xp2)*standerr.
compute op = {op, temp}.
end if.
do if (detail = 1).
do if (totlp = 2).
print yname/title = '*****
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:'/'format = A8.
end if.
do if (totlp <> 2).

```

```

print yname/title =
*****
*****'/rlabels = 'Outcome:.'/format
= A8.
end if.
end if.
do if (dichy = 1 and bt = 1 and totlp = 1).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:.'/cnames = nmsd/format = F9.2.
end if.
do if (dichy = 0).
compute r2full = 1-(sse/ssty).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute jndf=dfden.
compute wsdf=dfden.
compute xd = abs(xp2).
compute jncrit = (dfden* (exp((dfden-
(5/6))*(xd/(dfden-
(2/3)+(.11/dfden)))*(xd/(dfden-
(2/3)+(.11/dfden))))-1)).
compute summ = {sqrt(r2full), r2full, mse,
fratio, dfnum, dfden, pfr}.
compute temp1=coeff-sqrt(jncrit)*standerr.
compute temp2=coeff+sqrt(jncrit)*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (detail = 1).
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels =
'R', 'R-sq', 'MSE' 'F', 'df1', 'df2', 'p'/format =
F10.4.
end if.
end if.
end if.
do if (dichy = 1).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(coeff)-
1)).
compute mcF = LLdiff/LL3.

```

```

compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
do if (detail = 1).
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format =
F10.4.
end if.
end if.
do if (totlp = 2).
compute datanmy={constant';
datanmy((nmeds+2):nrow(datanmy),1)}.
end if.
do if (detail = 1).
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
do if (ws=1).
compute op2=op(1:(1+nmeds),:).
compute datanmy2 = {'c" path'; bpathnam(
(1+(nmeds>1)): ((1+(nmeds>1))+(nmeds-
1)),1)}.
do if (totlp=2).
compute datanmy2={'c path'}.
compute op2=op(1,:).
compute wsdf=n-1.
end if.
print op2/title = 'Model'/rnames =
datanmy2/cnames = cnms/format = F10.4.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
do if (ws <> 1).
print op/title = 'Model'/rnames =
datanmy/cnames = cnms/format = F10.4.
do if (covcoeff=1).
compute
covmattp=covmat(1:nrow(op),1:nrow(op)).
compute cnamesp=t(datanmy).

```

```

print covmattp/title='Covariance matrix of
regression parameter
estimates'/rnames=datanmy/cnames=cnamest
p/format= F10.4.
end if.
end if.
end if.
do if (ttt = 0 and totlp = 1).
compute deco = op((nmeds+2),:).
do if (ws = 1).
compute deco=op(1,:).
end if.
end if.
do if (ttt = 0 and totlp = 2).
compute decotot = op(2,:).
do if (ws = 1).
compute decotot=op(1,:).
end if.
end if.
do if (nmods > 0 and model > 4 and detail = 1
and nrow(yintkey) > 1)).
print yintkey/title = 'Product terms key:'/format
= A8.
end if.
do if (nmods > 0 and model < 4 and detail = 1).
print yintkey2/title = 'Product terms key:'/format
= A8.
do if ((model = 1 or model = 2) and dichy = 0).
compute temp={((op(4,3)**2)*(1-r2full))/dfden,
op(4,3)**2, 1, dfden, op(4,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(4,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(2,1).
do if (model = 2).
compute temp={temp;((op(6,3)**2)*(1-
r2full))/dfden, op(6,3)**2, 1, dfden, op(6,4)}.
do if (hc3=1).

```

```

compute temp(2,1)=((hcnot(6,1)**2)*(1-
r2full))/dfden.
end if.
compute frat2=(dfden*(r2full-(1-
(ssem2/ssty))))/(2*(1-r2full)).
do if (hc3=1).
compute lmat=make(nrow(coeffplt),2,0).
compute lmat(4,1)=1.
compute lmat(6,2)=1.
compute frat2 =
(t(t(lmat)*coeffplt)*inv(t(lmat)*covmat*lmat)*((t(l
mat)*coeffplt))/2).
end if.
compute temp={temp;(r2full-(1-
(ssem2/ssty))),frat2,2,dfden,1-
fcd(frat2,2,dfden)}.
compute rnms={rnms; yintkey2(3,1);'Both'}.
end if.
print temp/title = 'R-square increase due to
interaction(s):'/rnames=rnms/clabels = 'R2-
chng', 'F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (model = 3 and dichy = 0).
compute temp={((op(8,3)**2)*(1-r2full))/dfden,
op(8,3)**2, dfden, op(8,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(8,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(5,1).
print temp/title = 'R-square increase due to
three-way interaction:.'/rnames=rnms/clabels =
'R2-chng', 'F(1,df2)', 'df2', 'p'/format = F10.4.
end if.
end if.
end if.
do if (model = 6 and totlp = 1).
compute
mmpaths(nrow(mmpaths),1)=coeff(nrow(mmpa
ths),1).

```

```

compute
mmpaths(nrow(mmpaths),(2:(nmeds+1)))=t(co
eff(2:(nmeds+1),1)).
end if.
do if (totlp = 1).
loop im = 1 to nmeds.
do if (model < 4).
compute ymat(1,im) = coeff(3,1)*(1-yes).
compute ymat(2,im) = coeff(4,1).
compute cmat(1,im) = covmat(3,3).
compute cmat(2,im) = covmat(4,4).
compute cmat(5,im) = covmat(3,4).
compute jnb1=coeff(3,1).
compute jnb3=coeff(4,1).
compute jnsb1=covmat(3,3).
compute jnsb3=covmat(4,4).
compute jnsb1b3=covmat(3,4).
do if (model = 2 or model = 3).
compute ymat(3,im) = coeff(6,1).
compute cmat(3,im) = covmat(6,6).
compute cmat(6,im) = covmat(3,6).
compute cmat(8,im) = covmat(4,6).
end if.
do if (model = 3).
compute ymat(4,im) = coeff(8,1).
compute cmat(4,im) = covmat(8,8).
compute cmat(7,im) = covmat(3,8).
compute cmat(9,im) = covmat(4,8).
compute cmat(10,im) = covmat(6,8).
compute jnb1=coeff(4,1).
compute jnb3=coeff(8,1).
compute jnsb1=covmat(4,4).
compute jnsb3=covmat(8,8).
compute jnsb1b3=covmat(4,8).
end if.
end if.
do if (model > 3).
compute ymat(5,im) = coeff((1+im),1).
end if.
do if (xmy = 1).

```

```

compute ymat(6,im) = coeff((2+nmeds+im),1).
end if.
do if (vy = 1).
compute ymat(6,im) = coeff((3+nmeds+im),1).
end if.
do if (qy = 1 and vy = 1).
compute ymat(6,im) = coeff((5+nmeds+((im-1)*2)),1).
compute ymat(7,im) = coeff((6+nmeds+((im-1)*2)),1).
end if.
do if (vqy = 1).
compute ymat(6,im) = coeff((6+nmeds+((im-1)*3)),1).
compute ymat(7,im) = coeff((7+nmeds+((im-1)*3)),1).
compute ymat(8,im) = coeff((8+nmeds+((im-1)*3)),1).
end if.
do if (wmy = 1).
compute ymat(7,im) = coeff((3+nmeds+im-wy),1).
end if.
do if (wmy = 1 and vy = 1).
compute ymat(7,im) =coeff((4+(nmeds*2)+im-wy),1).
end if.
do if (wmy = 1 and vy = 1 and wvmy = 1).
compute ymat(7,im) =
coeff((6+(nmeds*2)+((im-1)*2)-wy),1).
compute ymat(8,im) =
coeff((7+(nmeds*2)+((im-1)*2)-wy),1).
end if.
do if (wmy = 1 and zmy = 1).
compute ymat(6,im) = coeff((6-(wzy*3)+(wzm-1)+(nmeds*2)+((im-1)*2)*wzm)+((im-1)*(1-wzm))-((zy-wzy)*2),1).
compute ymat(7,im) = coeff((3-zy+im+nmeds),1).
do if (wzmy = 1).

```



```

compute ymat(8,im) = coeff((7-
(wzy*3)+(nmeds*2)+((im-1)*2)),1).
end if.
end if.
do if (nmods > 0 and model <> 5).
loop indlp = 1 to nrow(modvals).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp)).
do if (model > 6).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp))*csum(ym
at(5:8,im)*vmat(5:8,indlp)).
end if.
end loop.
compute indboot((bt+(im-
1)*(boot+1)),:)=t(indeff).
do if (imm =1 or imm = 3).
compute
indbootp(bt,im)=ymat(2,im)*ymat(5,im).
do if (imm=3).
compute
indbootp(bt,(im+nmeds))=ymat(3,im)*ymat(5,i
m).
do if (zdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxz-cminz).
end if.
end if.
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
end if.
end if.
do if ((imm= 2 or imm = 4) or model = 74).
compute
indbootp(bt,im)=ymat(1,im)*ymat(6,im).
do if (imm=4).

```

```

compute
indbootp(bt,(im+nmeds))=ymat(1,im)*ymat(7,i
m).
do if (qdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxq-cminq).
end if.
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
end if.
end if.
do if (imm=7).
compute
indbootp(bt,im)=ymat(2,im)*ymat(6,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(2,im)*y
mat(6,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(6,im)*onesmv1)+(ymat(2,im)*y
mat(6,im)*t(immmv1)).
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).

```

```

compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1))))*(c
maxv-cminv).
end if.
end if.
do if (imm=5).
compute
indbootp(bt,im)=ymat(4,im)*ymat(5,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(4,im)*y
mat(5,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(3,im)*ymat(5,im)*onesmv1)+(ymat(4,im)*y
mat(5,im)*t(immmv1)).
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2))))*(c
maxw-cminw).
end if.
do if (zdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxz-
cminz).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1))))*(c
maxz-cminz).
end if.
end if.
do if (imm=6).

```

```

compute
indbootp(bt,im)=ymat(1,im)*ymat(8,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(1,im)*ymat(6,im)*onesmv2)+(ymat(1,im)*y
mat(8,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(7,im)*onesmv1)+(ymat(1,im)*y
mat(8,im)*t(immmv1)).
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxv-cminv).
end if.
do if (qdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxq-
cminq).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxq-cminq).
end if.
end if.
do if ((model = 58 or model = 59) and
wvdich=1).
compute
indbootp(bt,im)=(ymat(1,im)*ymat(7,im)*(cmax
w-cminw)+(ymat(2,im)*ymat(5,im)*(cmaxw-
cminw)+(ymat(2,im)*ymat(7,im)*((cmaxw*cma
xw)-(cminw*cminw))).
end if.
end if.

```

```

do if (model = 4 or model = 5).
compute
indboot(bt,im)=csum(yamat(1:4,im)*vmat(1:4,1
))*csum(yamat(5:8,im)*vmat(5:8,1)).
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.00000000000001.
end if.
compute sumind=rsum(indboot(bt,:)).
do if (im=nmeds).
compute
pmeff(bt,2:(im+1))=indboot(bt,:)/(sumind+coeff(
(2+nmeds),1)).
end if.
compute
rmeff(bt,(im+1))=indboot(bt,im)/coeff((2+nmeds
),1).
compute
abpseff(bt,(im+1))=indboot(bt,im)/stddevy.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*stddevx.
do if (ncovs > 0 or clsdmy > 0).
compute
abpseff(bt,(im+1))=indboot(bt,im)/sdycov.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*sdxcov.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,1) = r2my-(r2full-r2xy).
compute abmmr = 1.
do if (indboot(bt,im) < 0).
compute abmmr = cmin(abmm).
end if.
do if (indboot(bt,im) > 0).
compute abmmr = cmax(abmm).
end if.
compute kappa2(bt,1)=indboot(bt,im)/abmmr.
compute tmp = indboot(bt,im)/abmmr.
end if.

```

```

end if.
end if.
do if (model = 6).
do if (nmeds = 2).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(4,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3).
compute
indboot(bt,3)=mmpaths(3,1)*mmpaths(4,3).
end if.
do if (nmeds = 3).
compute indboot(bt,1) =
mmpaths(2,1)*mmpaths(5,2).
compute indboot(bt,2) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3).
compute indboot(bt,3) =
mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4).
compute indboot(bt,4) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4).
compute indboot(bt,5) =
mmpaths(3,1)*mmpaths(5,3).
compute indboot(bt,6) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4).
compute indboot(bt,7) =
mmpaths(4,1)*mmpaths(5,4).
end if.
do if (nmeds = 4).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(6,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(6,3).
compute
indboot(bt,3)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(6,4).

```

```

compute
indboot(bt,4)=mmpaths(2,1)*mmpaths(5,2)*mmpaths(6,5).
compute
indboot(bt,5)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(6,4).
compute
indboot(bt,6)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3)*mmpaths(6,5).
compute
indboot(bt,7)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,8)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,9)=mmpaths(3,1)*mmpaths(6,3).
compute
indboot(bt,10)=mmpaths(3,1)*mmpaths(4,3)*mmpaths(6,4).
compute indboot(bt,11)
=mmpaths(3,1)*mmpaths(5,3)*mmpaths(6,5).
compute indboot(bt,12) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,13) =
mmpaths(4,1)*mmpaths(6,4).
compute indboot(bt,14) =
mmpaths(4,1)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,15) =
mmpaths(5,1)*mmpaths(6,5).
end if.
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.00000000000001.
end if.
compute
pmeff(bt,:)=indboot(bt,:)/(rsum(indboot(bt,:))+mmpaths(nrow(mmpaths),1)).

```

```

compute rmeff(bt,:) =
indboot(bt,:)/mmpaths(nrow(mmpaths),1).
compute abpseff(bt,:)=indboot(bt,:)/stddevy.
compute
abcseff(bt,:)=(stddevx*indboot(bt,:))/stddevy.
do if (ncovs > 0 or clsdmy > 0).
compute abpseff(bt,:)=indboot(bt,:)/sdy cov.
compute
abcseff(bt,:)=(sdxcov*indboot(bt,:))/sdy cov.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,:) = r2my-(r2full-r2xy).
end if.
end if.
end if.
end loop.
end if.
end loop.
do if (savboot = 1 and bt > 1).
compute bootstrp((bt-1),:)=bootcoef.
end if.
end loop.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
release datayed, dat, datamed, data, y, x,
datatm, dataty, mints2, cons, mnv, tmp.
do if (mc > 0).
compute x1 = sqrt(-
2*ln(uniform(mc,nrow(mcmats))))&*cos((2*3.14
159265358979)*uniform(mc,nrow(mcmats))).
compute x1=x1*chol(mcmats).
loop i=1 to nrow(x1).
compute x1(i,:)=x1(i,:)+t(mccoeff).
end loop.
loop i = 1 to nmeds.

```



```

compute
indboot(2:nrow(indboot),i)=x1(:,i)*x1(:,(i+nme
ds)).
end loop.
end if.
do if (ttt = 0 and model > 3).
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
end if.
do if (model <> 74).
do if (dichy = 0).
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 't', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 't', 'p', 'LLCI', 'ULCI'/format =
F10.4.
else.
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 'Z', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 'Z', 'p', 'LLCI', 'ULCI' /format =
F10.4.
end if.
end if.
end if.
do if (ttt > 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
compute clbs = {modvnm2, 'Effect', 'SE', 't', 'p',
'LLCI', 'ULCI'}.

```

```

compute tratio = (directv&/sedir).
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
compute outp = {modvalsd, directv, sedir,
tratio, p}.
do if (dichy = 0).
compute temp1=directv-sqrt(jncrit)*sedir.
compute temp2=directv+sqrt(jncrit)*sedir.
compute outp = {outp, temp1, temp2}.
end if.
do if (dichy = 1).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute temp = directv-abs(xp2)*sedir.
compute outp = {outp, temp}.
compute temp = directv+abs(xp2)*sedir.
compute outp = {outp, temp}.
compute clbs = {modvnm2, 'Effect', 'SE', 'Z', 'p',
'LLCI', 'ULCI'}.
end if.
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
print outp/title = 'Conditional direct effect(s) of
X on Y at values of the moderator(s):'/format =
F10.4 /cnames = clbs.
end if.
do if (nmods > 0 and model <> 5).
do if (model < 4).
print/title =
'*****'
*****'.
compute zmat(1,1) = 1*(1-yes).
compute cfse = make(nrow(modvals),1,0).
loop #m = 1 to nrow(modvals).
do if (model = 1).
compute zmat(2,1)=modvals(#m,1)**2.
compute zmat(5,1)=2*modvals(#m,1).
end if.
do if (model = 2 or model = 3).
compute zmat(2,1)=modvals(#m,2)**2.
compute zmat(3,1)=modvals(#m,1)**2.

```

```

compute
zmat(4,1)=(modvals(#m,1)**2)*(modvals(#m,2)
**2)*(1-yes2).
compute zmat(5,1)=2*modvals(#m,2)*(1-yes).
compute zmat(6,1)=2*modvals(#m,1)*(1-yes).
compute
zmat(7,1)=2*modvals(#m,1)*modvals(#m,2)*(1-
yes).
compute
zmat(8,1)=2*modvals(#m,1)*modvals(#m,2).
compute
zmat(9,1)=2*modvals(#m,1)*(modvals(#m,2)**
2)*(1-yes2).
compute
zmat(10,1)=2*(modvals(#m,1)**2)*modvals(#m
,2)*(1-yes2).
end if.
compute cfse(#m,1)=sqrt(csum(zmat&*cmat)).
end loop.
end if.
do if (nmods > 0).
compute clbs = {modvnm, 'Effect'}.
loop im = 1 to nmeds.
compute obs = t(indboot(1+(im-1)*(boot+1),:)).
compute outp = {modvals, obs}.
do if (model < 4).
compute tstat = obs&/cfse.
do if (dichy = 0).
compute pval = 2*(1-tcdf(abs(tstat), (n-
ncol(xy)))).
compute temp=obs-sqrt(jncrit)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp=obs+sqrt(jncrit)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 't', 'p', 'LLCI', 'ULCI'}.
compute jnclbs=clbs.
end if.
do if (dichy = 1).
compute pval = 2*(1-cdfnorm(abs(tstat))).
compute temp = obs-abs(xp2)*cfse.

```

```

compute outp = {outp, cfse, tstat, pval, temp}.
compute temp = obs+abs(xp2)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute jnclbs=clbs.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indboot((1+(im-
1)*(boot+1)),:).
compute indboot2 = indboot((2+(im-
1)*(boot+1)):(1+(im-1)*(boot+1)+boot),:).
compute mnind = t(csum(indboot2)/boot).
compute stdind=t(sqrt((cssq(indboot2)-
((csum(indboot2)**2)/boot))/(boot-1))).
compute llci=make(1,ncol(indboot2),-999).
compute ulci=make(1,ncol(indboot2),-999).
loop #e = 1 to ncol(indboot2).

.
compute temp = indboot2(:,#e).
compute temp(GRADE( indboot2(:,#e) )) =
indboot2(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).

```

```

do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {modvals, obs, stdind, t(llci),
t(ulci)}.
compute clbs = {modvnm, 'Effect', 'Boot SE',
'BootLLCI', 'BootULCI'}.

```

```

end if.
compute mtemp = mnames(1, im).
compute rlbs =
make(nrow(modvals),1,mnames(1,im)).
do if (model < 4).
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
do if (yes=0).
print outp/title = 'Conditional effect of X on Y at
values of the moderator(s):'/cnames =
clbs/format = F10.4.
else.
compute outp=outp(:,3:ncol(outp)).
compute clbs=clbs(:,3:ncol(clbs)).
compute clbs(1,1)='Contrast'.
print outp/title = 'Contrast of conditional effects
of X on Y'/cnames = clbs/format = F10.4.
end if.
end if.
do if (model > 5 and mod74dic <> 1).
do if (im = 1).
print/title = 'Conditional indirect effect(s) of X on
Y at values of the moderator(s):'.
end if.
print outp/title = 'Mediator'/rnames =
rlbs/cnames = clbs/format = F10.4.
else if (mod74dic = 1 and model = 74).
do if (im = 1).
print/title = 'Indirect effect(s) of X on Y:'.
compute clbs3=clbs(1,2:ncol(clbs)).
end if.
compute outp3=outp(1,2:ncol(outp)).
print outp3/title = 'Mediator'/rnames =
rlbs/cnames = clbs3/format = F10.4.
end if.
end loop.
loop i = notes to 1 by -1.
do if (note(i,1) = 4 and yes=0).

```

```

print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 5 and yes=0).
print/title = 'Values for quantitative moderators
are the mean and plus/minus one SD from
mean.'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 14 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    below the mean was replaced
with the minimum because one SD below the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.
do if (note(i,1) = 15 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    above the mean was replaced
with the maximum because one SD above the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.
end loop.
do if (model = 3 and yes=0).
compute jnvals=make(nrow(matw),7,0).
compute jnvals(:,1)=matw.
compute jnvals(:,2)=jnb1+jnb3*jnvals(:,1).
compute
jnvals(:,3)=sqrt(jnsb1+2*jnvals(:,1)*jnsb1b3+(jn
vals(:,1)&*jnvals(:,1))*jnsb3).
compute jnvals(:,4)=jnvals(:,2)&/jnvals(:,3).

```

```

do if (dichy = 0).
compute jnvals(:,5)=2*(1-tcdf(abs(jnvals(:,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(:,5)=2*(1-
cdfnorm(abs(jnvals(:,4)))).
end if.
compute jnvals(:,6)=jnvals(:,2)-
sqrt(jncrit)*jnvals(:,3).
compute
jnvals(:,7)=jnvals(:,2)+sqrt(jncrit)*jnvals(:,3).
compute clbs={clbs(:,1),clbs(:,3:ncol(clbs))}.
do if (coeffci = 0).
compute jnvals =jnvals(:,1:(ncol(jnvals)-2)).
end if.
print jnvals/title = 'Conditional effect of X*M
interaction at values of W:'/cnames
=clbs/format = F10.4.
end if.
do if (jn = 1 and (model = 1 or model = 3) and
jndich = 0 and yes=0).
print/title = '***** JOHNSON-
NEYMAN TECHNIQUE
*****'.
compute ajn =(jncrit*jnsb3)-(jnb3*jnb3).
compute bjn = 2*((jncrit*jnsb1b3)-(jnb1*jnb3)).
compute cjn = (jncrit*jnsb1)-(jnb1*jnb1).
compute radarg = (bjn*bjn)-(4*ajn*cjn).
compute den = 2*ajn.
compute nrts = 0.
do if (radarg >= 0 and den <> 0).
compute x21 = (-bjn+sqrt(radarg))/den.
compute x22 = (-bjn-sqrt(radarg))/den.
compute roots = 0.
do if (x21 >= jnmin and x21 <= jnmax).
compute nrts = 1.
compute roots = {roots; x21}.
end if.
do if (x22 >= jnmin and x22 <= jnmax).

```



```

compute nrts = nrts + 1.
compute roots = {roots; x22}.
end if.
compute roots={roots,make(nrow(roots),2,0)}.
compute modtemp=m.
do if (model=3).
compute modtemp=w.
end if.
do if (nrts > 0).
compute roots = roots(2:nrow(roots),1:3).
compute roots(1,2)=(csum(modtemp <
roots(1,1))/n)*100.
compute roots(1,3)=(csum(modtemp >
roots(1,1))/n)*100.
do if (nrow(roots)=2).
compute roots(2,2)=(csum(modtemp <
roots(2,1))/n)*100.
compute roots(2,3)=(csum(modtemp >
roots(2,1))/n)*100.
end if.
print roots/title = 'Moderator value(s) defining
Johnson-Neyman significance
region(s)'/clabels = 'Value', '% below', '%
above'/format = F10.4.
compute jnvals=make((21+nrts),7,0).
loop i= 0 to 20.
compute jnvals((i+1),1)=jnmin+(i*((jnmax-
jnmin)/20)).
end loop.
loop i = 1 to nrts.
loop j = 2 to nrow(jnvals).
do if ((roots(i,1) > jnvals((j-1),1)) and (roots(i,1)
< jnvals(j,1))).
compute
jnvals((j+1):(21+i),1)=jnvals(j:(20+i),1).
compute jnvals(j,1)=roots(i,1).
end if.
end loop.
end loop.
loop i = 1 to nrow(jnvals).

```

```

compute jnvals(i,2)=jnb1+jnb3*jnvals(i,1).
compute
jnvals(i,3)=sqrt(jnsb1+2*jnvals(i,1)*jnsb1b3+(jn
vals(i,1)*jnvals(i,1))*jnsb3).
compute jnvals(i,4)=jnvals(i,2)/jnvals(i,3).
do if (dichy = 0).
compute jnvals(i,5)=2*(1-tcdf(abs(jnvals(i,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(i,5)=2*(1-
cdfnorm(abs(jnvals(i,4)))).
end if.
compute jnvals(i,6)=jnvals(i,2)-
sqrt(jncrit)*jnvals(i,3).
compute
jnvals(i,7)=jnvals(i,2)+sqrt(jncrit)*jnvals(i,3).
end loop.
do if (model = 1).
print jnvals/title = 'Conditional effect of X on Y
at values of the moderator (M)'/cnames
=jnclbs/format = F10.4.
end if.
do if (model = 3).
compute
jnclbs={jnclbs(:,1),jnclbs(:,3:ncol(jnclbs))}.
print jnvals/title = 'Conditional effect of X*M on
Y at values of the moderator (W)'/cnames
=jnclbs/format = F10.4.
end if.
end if.
do if (nrts = 0).
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
else.
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.

```

```

end if.
end if.
end if.
do if (model < 4 and (plot = 1)).
compute dataplot =
make((nrow(modvals)*nrow(matx)),(ncol(modv
als)+1),0).
compute tmp = 1.
loop i = 1 to nrow(modvals).
loop j = 1 to nrow(matx).
compute dataplot(tmp,:)= {matx(j,1),
modvals(i,:)}.
compute tmp=tmp+1.
end loop.
end loop.
compute dataplot = {dataplot,
make(nrow(dataplot),(1+dichy),0)}.
compute dataplo2 = make(nrow(dataplot),1,1).
do if (model = 1).
compute dataplo2 = {dataplo2, dataplot(:,2),
dataplot(:,1), (dataplot(:,1)*dataplot(:,2))}.
end if.
do if (model = 2 or model = 3).
compute dataplo2 = {dataplo2, dataplot(:,3),
dataplot(:,1), (dataplot(:,1)*dataplot(:,3)),
dataplot(:,2), (dataplot(:,1)*dataplot(:,2))}.
do if (model = 3).
compute dataplo2 = {dataplo2,
(dataplot(:,2)*dataplot(:,3)),
(dataplot(:,1)*dataplot(:,2)*dataplot(:,3))}.
end if.
end if.
loop i = 1 to nrow(dataplot).
compute tmp=dataplo2(i,:).
do if (ncovs > 0).
compute tmp = {tmp, covmeans}.
end if.
do if (cluster > 0).
compute tmp = {tmp, cldmeans}.
end if.

```

```

compute dataplot(i,(ncol(dataplot)-
(dichy)))=tmp*coeffplt.
do if (dichy = 1).
compute
dataplot(i,(ncol(dataplot)))=exp(tmp*coeffplt)/(1
+exp(tmp*coeffplt)).
end if.
end loop.
compute clbs = {xname, modvnm, 'yhat'}.
do if (dichy = 1).
compute clbs = {xname, modvnm, 'ln(odds)',
'prob'}.
end if.
print/title =
*****
*****!

```

```

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (model = 1).

```

```

do if (dichy = 0).
print/title = 'DATA LIST FREE/Opra_Pract
Eco_Innovation Operational_Performance.'
/space=1.
else if (dichy = 1).
print/title = 'DATA LIST FREE/Opra_Pract
Eco_Innovation logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
Operational_Performance BY Eco_Innovation.'
/space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Operational_Performance BY
Opra_Pract.' /space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
logodds BY Eco_Innovation.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
prob BY Eco_Innovation.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Opra_Pract.' /space=0.

```

```

print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Opra_Pract.' /space=0.
end if.
end if.
end if.
do if (model = 2 or model = 3).
do if (dichy = 0).
print/title = 'DATA LIST FREE/Opra_Pract
Eco_Innovation Operational_Performance.'
/space=1.
end if.
do if (dichy = 1).
print/title = 'DATA LIST FREE/Opra_Pract
Eco_Innovation logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
Operational_Performance BY
Eco_Innovation/PANEL ROWVAR=.' /space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Operational_Performance BY
Opra_Pract/PANEL ROWVAR=.' /space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
logodds BY Eco_Innovation/PANEL
ROWVAR=.' /space=0.

```

```

print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
prob BY Eco_Innovation/PANEL ROWVAR='
/space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Opra_Pract/PANEL
ROWVAR=' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Opra_Pract/PANEL
ROWVAR=' /space=0.
end if.
end if.
end if.
do if (ncovs > 0).
print/title = '* Estimates are based on setting
covariates to their sample means.'.
end if.
end if.
end if.
do if ((imm > 0) or (model = 74 and
mod74dic=0) or ((model = 58 or model = 59)
and wvdich=1))).
do if (imm > 4 and imm < 8).
compute
indbootp={indbootp(:,1:nmeds),indboop1,indbo
op2}.
end if.
compute obsprod = t(indbootp(1,:)).
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indbootp(1,:).
compute indbootp = indbootp(2:(boot+1),:).
compute mnindp = t(csum(indbootp)/boot).
compute stdindp=t(sqrt((cssq(indbootp)-
((csum(indbootp)**2)/boot))/(boot-1))).
compute llcip=make(1,ncol(indbootp),-999).

```

```

compute ulcip=make(1,ncol(indbootp),-999).
loop #e = 1 to ncol(indbootp).

.

compute temp = indbootp(:,#e).
compute temp(GRADE( indbootp(:,#e) )) =
indbootp(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).

```



```

end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcip(1,#e)=llcit.
compute ulcip(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {obsprod, stdindp, t(llcip),
t(ulcip)}.
compute clbs = {'Effect', 'SE(Boot)', 'BootLLCI',
'BootULCI'}.
do if (model = 8 or model = 12).
print/title='-----'.
compute outpimm=outp(1:nmeds,:).
print/title = 'Indirect effect of highest order
product:'/space=0.
print outpimm/title = 'Mediator'/cnames =
clbs/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).

```

```

print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'

end if.
compute clbs3 = {'Index', 'SE(Boot)',
'BootLLCI', 'BootULCI'}.
do if (imm < 5).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=outp((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'

compute outpimm=outp(1:nmeds,:).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
compute
outpimm=outp((nmeds+1):nrow(outp),:).
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).

```

```

compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),clbs3}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3(1,1)=modvnm(1,1).
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator:'/'format=A8.
print outpimm1/title =
'Mediator'/'cnames=clbs3/'rnames=cmmlbs/'form
at= F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/'space=0.
end if.
end if.
end if.
do if (boot = 0).
do if (model = 8 or model = 12).
print/title='-----'.
compute obsimm=obsprod(1:nmeds,:).
print/title = 'Indirect effect of highest order
product:'/'space=0.
print obsimm/title = 'Mediator'/'clabels =
'Effect'/'rnames = mnames/'format = F10.4.
end if.

```

```

do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'
.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'
.
end if.
do if ((imm > 0 and imm < 5) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator: '/format=A8.
end if.
compute outpimm=obsprod((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
end loop.
end if.
do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'
.
compute outpimm=obsprod(1:nmeds,1).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
compute
outpimm=obsprod((nmeds+1):nrow(obsprod),:)
.
compute outpimm={cmmval,outpimm}.

```

```

print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),'Index'}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3={modvnm(1,1),'Index'}.
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
end if.
end if.
end if.
compute conmake=0.
compute concols = 0.
do if ((model > 3 and model < 7) and (contrast
> 0) and nmods = 0 and nmeds > 1 ).
compute concols =
((ncol(indboot)*(ncol(indboot)-1))/2).
compute indcon=make(nrow(indboot),concols,-
999).
compute conkey = { ' ', ' ', ' ' }.
compute temp=1.
compute conmake=1.
loop i = 1 to (ncol(indboot)-1).

```

```

loop j = (i+1) to (ncol(indboot)).
compute indcon(:,temp)=indboot(:,i)-
indboot(:,j).
do if (contrast = 2).
compute indcon(:,temp)=abs(indboot(:,i))-
abs(indboot(:,j)).
end if.
do if (model <> 6).
compute conkey={conkey; mnames(1,i),
'minus', mnames(1,j)}.
end if.
do if (model = 6).
compute conkey={conkey; indlbl2(i,1), 'minus',
indlbl2(j,1)}.
end if.
compute temp=temp+1.
end loop.
end loop.
end if.
do if (model = 4 or model = 5).
compute clbs = {'Effect'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obs = {csum(obs); obs}.
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute rlbs = {rlbs;
cntname(1:ncol(indcon),1)}.
end if.
compute outp = obs.
compute outp2=outp.
do if (effsize = 1 and dichy = 0).
compute
pmeff(:,1)=rsum(pmeff(:,2:ncol(pmeff))).
compute
rmeff(:,1)=rsum(rmeff(:,2:ncol(rmeff))).
compute
abpseff(:,1)=rsum(abpseff(:,2:ncol(abpseff))).
compute
abcseff(:,1)=rsum(abcseff(:,2:ncol(abcseff))).

```

```

compute eff = {pmeff, rmeff, abpseff, abcseff}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute eff = {eff, r245}.
compute r245obs = {r245(1,1);r245(1,1)}.
compute kappa2ob = kappa2(1,1).
end if.
compute pmobs = t(pmeff(1,1:(nmeds+1))).
compute rmobs = t(rmeff(1,1:(nmeds+1))).
compute psobs = t(abpseff(1,1:(nmeds+1))).
compute csobs = t(abcseff(1,1:(nmeds+1))).
do if (contrast = 0).
compute outp2 = {obs, psobs, csobs, pmobs,
rmobs}.
end if.
do if (contrast > 0).
compute obs2=obs(1:nrow(psobs),:).
compute outp2 = {obs2, psobs, csobs, pmobs,
rmobs}.
end if.
compute clbs = {'ab', 'ab_ps', 'ab_cs', 'ab/c',
'ab/c'}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute outp2 = {outp2, r245obs}.
compute clbs = {clbs, 'R-sq_med'}.
do if (boot = 0 and mc = 0).
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
do if (boot = 0 and mc = 0).
do if (nmeds = 1).
compute outp2 = outp2(2,:).
compute rlbs = rlbs(2,1).
end if.
do if (model=5).
compute outp2=outp2(:,1:3).
end if.

```

```

print outp2/title = 'Indirect effect(s) of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1 and nmeds >
1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0 or mc > 0).
compute indboot = {rsum(indboot), indboot}.
compute bootSZ=boot.
do if (mc > 0).
compute bootSZ=mc.
end if.
compute ones = make(bootSZ,1,1).
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(bootSZ+1),:).
compute mnind = t(csum(indboot)/bootSZ).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/bootSZ))/(bootSZ-1))).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).
.
compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).

```



```

compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.

```

```

compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
do if (effsize = 1 and dichy = 0 and boot > 0).
compute estmte=eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).

.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2)))+1.

```

```

do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
end if.
end if.
do if (boot > 0 or mc > 0).
compute outp = {obs, stdind, t(llci), t(ulci)}.
do if (nmeds = 1).
compute outp = outp(2,:).
compute rlbs = rlbs(2,1).
end if.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.

```

```

do if (mc > 0).
compute clbs = {'Effect', 'MC SE', 'MC LLCI',
'MC ULCI'}.
end if.
print outp/title = 'Indirect effect of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (dichy = 0 and effsize = 1 and boot > 0).
compute outp = {psobs,
stdindf((2*(nmeds+1)+1):(3*(nmeds+1)),1),
t(llcif(1,(2*(nmeds+1)+1):(3*(nmeds+1)))),
t(ulcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))}.
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Partially standardized indirect
effect of X on Y'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute outp = {csobs,
stdindf((3*(nmeds+1)+1):(4*(nmeds+1)),1),
t(llcif(1,(3*(nmeds+1)+1):(4*(nmeds+1)))),
t(ulcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))}.
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Completely standardized
indirect effect of X on Y'/rnames = rlbs/cnames
= clbs/format = F10.4.
compute outp = {pmobs,
stdindf(1:(nmeds+1),1),t(llcif(1,1:(nmeds+1))),
t(ulcif(1,1:(nmeds+1))) }.
do if (model = 4).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
compute outp = {rmobs, stdindf(((nmeds+1)+1)
:(2*(nmeds+1)) ,1),

```

```

t(llcif(1,((nmeds+1)+1):(2*(nmeds+1))))),
t(ulcif(1,((nmeds+1)+1):(2*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
do if (nmeds = 1 and cluster = 0 and ncovs = 0
and model = 4).
compute r245obs = r245obs(1,1).
compute outp = {r245obs,
stdindf((4*(nmeds+1)+1):(4*(nmeds+1)+1),1),
t(llcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1))),
t(ulcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1)))}.
print outp/title = 'R-squared mediation effect
size (R-sq_med)'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
end if.
do if (normal = 1).
do if (nmeds = 1).
print sobel/title = 'Normal theory tests for
indirect effect'/clabels = 'Effect', 'se', 'Z',
'p'/format = F10.4.
end if.
do if (nmeds > 1).
compute rlbs2 = rlbs(2:nrow(rlbs),1).
print sobel/title = 'Normal theory tests for
specific indirect effects'/rnames = rlbs2/clabels
= 'Effect', 'se', 'Z', 'p'/format = F10.4.
end if.
end if.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).

```

```

print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
do if (model = 6).
compute clbs = {'eff'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obsfsum=csum(obs).
compute obs = {obsfsum; obs}.
compute indlbl = indlbl(1:nrow(obs),1).
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute indlbl = {indlbl;
cntname(1:ncol(indcon),1)}.
end if.
compute obs2=obs.
do if (boot = 0).
do if (dichy = 0 and effsize = 1).
compute clbs = {'eff', 'eff_ps', 'eff_cs', 'eff/c',
'eff/c'}.
do if (ncovs = 0).
compute obs = {obs, obs/stddevy,
(obs*stddevx/stddevy), obs/ctot,
obs/mmpaths(nrow(mmpaths),1)}.
end if.
do if (ncovs > 0 or clsdmy > 0).
compute obs = {obs, obs/sdy cov,
(obs*sdx cov/sdy cov),
obs/(obsfsum+mmpaths(nrow(mmpaths),1)),
obs/mmpaths(nrow(mmpaths),1)}.
end if.
compute obs2=obs.
do if (contrast > 0).
compute obs2 = obs(1:(nrow(obs)-concols),:).
end if.
end if.

```

```

print obs2/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute indboot = {rsum(indboot), indboot}.
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(boot+1),:).
compute mnind = t(csum(indboot)/boot).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/boot))/(boot-1))).
compute temp = nrow(indboot).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).
.
compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).

```

```

compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llici(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.

```



```

end if.
end loop.
compute obs = {obs, stdind, t(llci), t(ulci)}.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
print obs/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (effsize = 1 and dichy = 0).
compute indboot=indboot(:,1:(ncol(indboot)-
concols)).
compute eff = {rsum(abpseff), abpseff,
rsum(abcseff), abcseff, rsum(pmeff), pmeff,
rsum(rmeff), rmeff}.
compute effobs = eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).

.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(effobs(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).

```

```

do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( ( effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute temp2 = stdindf(1:ncol(indboot),1).
compute temp3 = effobs(:,1:ncol(indboot)).
compute templow = llcif(1,1:ncol(indboot)).
compute temphi = ulcif(1,1:ncol(indboot)).

```

```

compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Partially standardized indirect
effect of X on Y'/cnames = clbs/rnames =
indlbl/format = F10.4.
compute temp2 =
stdindf((ncol(indboot)+1):(2*ncol(indboot)),1).
compute temp3 =
effobs(:,(ncol(indboot)+1):(2*ncol(indboot))).
compute templow =
llcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute temphi =
ulcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Completely standardized
indirect effect of X on Y'/cnames = clbs/rnames
= indlbl/format = F10.4.
compute temp2 =
stdindf((2*(ncol(indboot))+1):(3*ncol(indboot)),1
).
compute temp3 =
effobs(:,(2*(ncol(indboot))+1):(3*ncol(indboot)))
.
compute templow =
llcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute temphi =
ulcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
compute temp =
eff(:,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temp2 =
stdindf((3*(ncol(indboot))+1):(4*ncol(indboot)),1
).

```

```

compute temp3 =
effobs(:,(3*(ncol(indboot))+1):(4*ncol(indboot)))
.
compute templow =
llcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temphi =
ulcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
end if.
end if.
do if (nmeds = 2).
compute effkey = {xname, '->', mnames(1,1), '->', yname, ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, ' ', ' '}.
compute effkey = {indlbl(2:4,1), effkey}.
end if.
do if (nmeds = 3).
compute effkey = {xname, '->', mnames(1,1), '->', yname, ' ', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname, ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,3), '->', yname, ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, ' ', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', mnames(1,3), '->', yname, ' ', ' '}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, '', '', '', ''}.
compute effkey = {indl(2:8,1), effkey}.
end if.
do if (nmeds = 4).
compute effkey = {xname, '->' , mnames(1,1), '->', yname, '', '', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->', yname, '
', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->', yname, '
', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,4), '->', yname, '
', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname, '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,4), '->', yname, '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', mnames(1,4), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, '', '', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,4), '->', yname, '
', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, '', ''}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, '',' ',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', mnames(1,4), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,4), '->', yname, '',' ',' ',' ',' '}.
compute effkey = {indlbl(2:16,1), effkey}.
end if.
print effkey/title = 'Indirect effect key'/format =
A8.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
end if.
DO IF (medte2=1).
compute
name=t(mdlnms2(2:nrow(mdlnms2),1)).
compute tmpdat=datayed(:,2:ncol(datayed)).
compute dd=tmpdat(:,1).
compute
dd={dd,tmpdat(:,(2+nmeds)),tmpdat(:,2:(1+nmeds
ds)))}.
compute stratify=0.
do if (ncovs > 0).
compute cname = cnames.
compute
dd={dd,tmpdat(:,(3+nmeds):(2+nmeds+ncovs))
}.
end if.
compute method = 1.
compute samples = boot.
do if (mc > 0).
compute samples = mc.
compute method=2.

```

```

end if.
compute omnibus = 1.
compute booterr = 0.
do if (mcx < 0 or mcx > 4).
compute mcx = 1.
end if.
compute ovals = ncol(design(dd(:,1))).
compute nte = notes.
compute criterr = 0.
compute xskip = 0.
compute dumok = 0.
compute badend = 0.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute priorlo = -9999999.
compute priorhi = 9999999.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute reps = samples.
do if (contrast = 1).
compute note(nte,1) = 26.
compute nte=nte+1.
end if.
do if (effsize = 1).
compute note(nte,1) = 27.
compute nte=nte+1.
end if.
do if (normal = 1).
compute note(nte,1) = 28.
compute nte=nte+1.
end if.
do if (method = 2).
do if (omnibus = 1).
compute note(nte,1) = 25.
compute nte=nte+1.
end if.
compute samples = 0.

```

```

do if (reps < 1000).
compute reps = 1000.
end if.
end if.
do if (method = 1).
compute samples = reps.
end if.
compute temp = ncol(dd).
compute n = nrow(dd).
compute ones = make(n,1,1).
compute temp=dd(:,2).
compute nxd = ncol(temp).
compute nx = nxd.
do if (nx = 1 and mcx > 0).
compute temp = dd.
compute temp(GRADE(dd(:,2)),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,2)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute mnvls = cmin(t(nnvls)).
do if (mnvls < 2).
compute errs = errs + 1.
compute runerrs(errs,1) = 37.
compute criterr = 1.
end if.
do if (nvls > 9).
compute errs = errs+1.
compute runerrs(errs,1) = 36.
compute criterr = 1.
end if.
do if (criterr = 0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,2).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,2) <> nnvls((temp-1),1)).
compute nnvls(temp,1)=dd(i,2).
compute temp = temp+1.

```



```

end if.
end loop.
do if (mcx > 0).
compute x = dummy(:,2:ncol(dummy)).
compute nx = ncol(x).
compute minus1 = make(1,ncol(x),-1).
compute xdes=make((nx+1),3,0).
compute xdes(1,1)=dd(1,2).
compute xdes(1,2)=1.
compute temp = 2.
loop k = 2 to n.
do if (dd(k,2) <> dd((k-1),2)).
compute xdes(temp,2) = k.
compute xdes(temp,1) = dd(k,2).
compute xdes((temp-1),3) = k-1.
compute temp=temp+1.
end if.
end loop.
compute xdes((temp-1),3)=n.
compute xdes = {xdes, (xdes(:,3)-xdes(:,2)+1)}.
do if (stratify <> 0).
compute note(nte,1) = 7.
compute nte=nte+1.
end if.
do if (mcx = 4).
loop k = 1 to n.
do if (rsum(x(k,:)) = 0).
compute x(k,:) = minus1.
end if.
end loop.
end if.
do if (mcx = 2 or mcx = 3).
compute note(nte,1)=4.
compute nte = nte+1.
loop k = 1 to n.
do if (rsum(x(k,:)) > 0).
loop i = 1 to ncol(x).
do if (x(k,i) = 0).
compute x(k,i) = 1.
else.

```

```

break.
end if.
end loop.
end if.
end loop.
do if (mcx = 3).
compute conmat1={-8,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1; 0,0,0,0,0,-
3,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.
loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute x(k,:)=conmat1((rsum(x(k,:))+1),:).
end loop.
end if.
end if.
compute xdata = x.
compute xskip = 1.
compute xname = ddd(1,1:nx).
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcx = 4).
compute dummat(1,:) = minus1.
end if.
do if (mcx = 2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcx = 3).
compute dummat=conmat1.
end if.

```

```

compute dummat={nnvls, dummat}.
compute xname2={name(1,2), xname}.
end if.
end if.
end if.
compute nm=nmeds.
compute r2b = make(nm,1,-999).
compute indeff = make((nx*nm),1,-999).
compute cov =
make((nx*nm+nm),(nx*nm+nm),0).
compute mns=make(nrow(cov),1,0).
compute ny = ncol(dd)-nm-nxd-ncovs.
compute y = dd(:,1).
compute ydata = y.
compute yname = name(1,1).
do if (xskip = 0).
compute x = dd(:,2:(1+nxd)).
compute xdata = x.
compute xname = name(1,2:(1+nxd)).
compute indlbs = t(xname).
end if.
compute m = dd(:,(2+nxd):(1+nxd+nm)).
compute mdata = m.
compute mlab = { ' M1 = ' ; ' M2 = ' ; ' M3 = ' ; '
M4 = ' ; ' M5 = ' ; ' M6 = ' ; ' M7 = ' ; ' M8 = ' ; '
M9 = ' ; ' M10 = ' }.
compute mlab = {mlab ; ' M11 = ' ; ' M12 = ' ; '
M13 = ' ; ' M14 = ' ; ' M15 = ' }.
compute mname =
name(1,(2+nxd):(1+nxd+nm)).
do if (ncovs > 0).
compute c =
dd(:,(2+nxd+nm):(1+nxd+nm+ncovs)).
compute x = {x, c}.
compute xdata = x.
compute cdata = c.
compute xname = {xname, cname}.
end if.
compute nms={yname; t(mname);
t(xname(1,1:(ncol(xname)-ncovs)))}.

```

```

do if (dumok = 1).
compute nms={yname; t(mname); name(1,2)}.
end if.
do if (nm < 16).
do if (dumok = 1).
print dummat/title = 'Coding of categorical X
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
compute
boots=make((samples+1),((nx+1)*nm),-999).
do if (errs = 0 and criterr = 0).
loop k = 1 to (samples+1).
do if (k = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef=0.
do if (k > 1).
compute u=trunc(uniform(n,1)*n)+1.
compute xdata = x(u,:).
compute mdata = m(u,:).
compute ydata = y(u,:).
do if (ncovs > 0).
compute cdata = c(u,:).
end if.
end if.
loop i = 1 to nm.
compute start = 1+(i-1)*nx.
compute xmat = {ones, xdata}.
do if (covmy=2).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = mdata(:,i).
compute amat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*amat.

```

```

compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute r2cha = r2.
compute adjr2cha = adjr2.
do if (ncovs > 0 and covmy <> 2).
compute xmat = {ones, cdata}.
compute atmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*atmat)**2).
compute r2step1 = (sstotal-ssresid2)/sstotal.
compute r2cha = r2-r2step1.
compute adjr2cha = adjr2- (1-((1-r2step1)*(n-
1)/(n-ncol(xmat)))).
compute xmat = {ones, xdata}.
end if.
compute bootcoef={bootcoef,t(amat)}.
compute r2b(i,1)=adjr2cha.
do if (k = 1).
.
compute hcinvtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.

```

```

end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute tratio = amat&/sqrt(diag(covmat)).
compute rnms = mname(1,i).
print rnms/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.
compute lmat = ident(nrow(amat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*amat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*amat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/'format = F10.4
/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'.
compute op = {amat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/'clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/'rnames = rnms/'format =
F10.4.
compute cov(start:(start+(nx-
1)),start:(start+(nx-

```

```

1)))=covmat(2:(ncol(covmat)-(ncovs*(covmy <>
2))),2:(ncol(covmat)-(ncovs*(covmy <> 2)))).
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
=F10.4.
end if.
end if.
compute mns(start:(start+(nx-
1)),1)=amat(2:(1+nx)).
end loop.
compute xmat = {ones,mdata,xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (k = 1).
print yname/title =
*****
*****'/rlabels = 'Outcome:'/format
=A8.
do if (ovals = 2).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:'/cnames = nmsd/format = F9.2.
end if.
end if.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(ymat)/n)).
compute LL3 = ymat*&ln(pt2)+(1-ymat)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.
.
compute pt1lp= pt1.
compute xlp= xmat.

```

```

compute ylp= ymat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp&*ln(pt1lp)+(1-ylp)&*ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.

```



```

compute bootcoef={bootcoef,t(bmat)}.
compute LL1 = LL2.
compute covmat=varb.
end if.
do if (ovals <> 2).
compute bmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*bmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute bootcoef={bootcoef,t(bmat)}.
end if.
do if (k = 1).
do if (ovals = 2).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).
compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)*sqrt(diag(covmat)).
compute outp = {outp, temp}.

```

```

compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms2/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute
opdirsum=outp((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
end if.
do if (ovals <> 2).

.
compute hcinvtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(bmat)).
compute lmat = lmat(:,2:ncol(lmat)).

```

```

compute fratio =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lmat)*bmat)))/(ncol(xmat)-1).
compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat((2+nmeds):(2+nmeds+(nx-1)),:)= lmat2.
compute fratiodr =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lmat)*bmat)))/nx.
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-ncol(xmat))).
compute df2=(n-ncol(xmat)).
compute op = {sqrt(r2), r2, mse, fratio, (ncol(xmat)-1), df2, pfr}.
print op/title = 'Model Summary'/format = F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'.
compute tratio = bmat&/sqrt(diag(covmat)).
compute rnms2 = {'Constant'; t(mname); t(xname)}.
compute op = {bmat, sqrt(diag(covmat)), tratio, (2*(1-tcdf(abs(tratio), n-ncol(xmat)))) }.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-(5/6))*((xp2/(temp-(2/3)+(.11/temp)))*xp2/(temp-(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-sqrt(abs(temp))*op(:,2).
compute temp2 = op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't', 'p', 'LLCI', 'ULCI'/rnames = rnms2/format = F10.4.
compute
opdirsum=op((2+nmeds):(1+nmeds+nx),:).

```

```

compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
compute xmat = {ones,mdata}.
do if (ncovs > 0).
compute xmat = {xmat, cdata}.
end if.
compute btmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*btmat)**2).
compute r2direct = r2-((sstotal-
ssresid2)/sstotal).
compute pfr = 1-fcdf(fratiodr,(nx),(n-1-nx-nm-
ncovs)).
compute r2direct = {r2direct, fratiodr, nx, (n-1-
nx-nm-ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms2/cnames=rnms2/form
at= F10.4.
end if.
end if.
do if (k = 1 and toteff = 1).
print yname/title = '*****
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome: '/format = A8.
compute xmat = {ones, xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(ymat)/n)).
compute LL3 = ymat*ln(pt2)+(1-ymat)*ln(1-
pt2).
compute LL3 = -2*csum(LL3).

```

```

compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.

compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= ymat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)*&ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.

```

```

end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute LL1 = LL2.
compute covmat=varb.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(bmat)-
1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms = {'Constant'; t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).
compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute optotsum=outp(2:(1+nx),:).
end if.
do if (ovals <> 2).

```

```

compute cmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*cmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).

.

compute hcinvtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvtX*t( xmat )* xmat
*hcinvtX
.
compute lmat = ident(nrow(cmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).

```

```

compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = cmat&/sqrt(diag(covmat)).
compute op = {cmat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute optotsum=op(2:(1+nx),:).
compute ssresid2 = ssresid.
compute r2total = r2.
compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat(2:(1+nx),:) = lmat2.
compute fratio3 =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/nx.
do if (ncovs > 0).
compute xmat = {ones, cdata}.
compute ctmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*ctmat)**2).
compute r2total = r2-((sstotal-ssresid2)/sstotal).
end if.

```



```

compute pfr = 1-fcdf(fratio3,(nx),(n-1-nx-
ncovs)).
compute r2tot2 = {r2total, fratio3, nx, (n-1-nx-
ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns((nrow(mns)-
nm+1):nrow(mns),1)=bmat(2:(1+nm),1).
compute tmp = 1.
loop i = 1 to (nm*nx).
compute
boots(k,i)=mns(i,1)*mns(((nm*nx)+tmp),1).
do if ((i/nx)=trunc(i/nx)).
compute
boots(k,((nm*nx)+tmp))=r2b(tmp,1)*mns(((nm*
nx)+tmp),1).
compute tmp = tmp+1.
end if.
end loop.
do if (savboot = 1 and k > 1).
compute bootstrp((k-1),:)=bootcoef.
end if.
end loop.
do if (method = 1).
compute llci=make(1,ncol(boots),-999).
compute ulci=make(1,ncol(boots),-999).
compute indeff = t(boots(1,1:(nm*nx))).
compute indeff22=t(boots(1,:)).
compute omni = t(boots(1,((nm*nx)+1) :
ncol(boots))).
do if (nrow(boots) > 1).
compute boots = boots(2:nrow(boots),:).
compute bootse = t(sqrt((((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).

```

```

loop i = 1 to ncol(boots).

.

compute temp = boots(:,i).
compute temp(GRADE( boots(:,i) )) = boots(:,i).
compute badlo = 0.
compute badhi = 0.
do if ( (indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(indeff22(i,1)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.

```

```

do if ( (indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,i)=llcit.
compute ulci(1,i)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute llci = t(llci).
compute ulci= t(ulci).
compute llci={llci,ulci}.
end if.
end if.
do if (method = 2).
compute tmp=1.
loop i = 1 to (nm*nx).
compute
indeff(i,1)=mns(i,1)*mns(((nm*nx)+tmp),1).
do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = sqrt(-
2*ln(uniform(reps,nrow(cov))))&*cos((2*3.1415
9265358979)*uniform(reps,nrow(cov))).
compute boots=boots*chol(cov).
loop i = 1 to ncol(boots).
compute boots(:,i)=boots(:,i)+mns(i,1).
end loop.
compute tmp=1.
loop i = 1 to (nm*nx).

```

```

compute
boots(:,i)=boots(:,i)*boots(:,((nm*nx)+tmp)).
compute temp = boots(:,i).
compute temp(GRADE(boots(:,i))) = boots(:,i).
compute boots(:,i) = temp.
do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = boots(:,1:(nm*nx)).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
compute llci =
t(((boots(cilow,:));(boots(cihigh,:)))).
end if.
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
compute rnmssum=rnms(2:(1+nx),1).
do if (ovals <> 2).
print optotsum/title = 'Relative total effects of X
of Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.
print r2tot2/title = 'Omnibus test of total effect of
X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print optotsum/title = 'Relative total effects of X
on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
end if.
compute
rnmssum=rnms2((2+nmeds):(1+nmeds+nx),1).
do if (ovals <> 2).

```

```

print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.
print r2direct/title = 'Omnibus test of direct
effect of X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/'space=0.
do if (omnibus = 1 and (method = 1 or method
= 3)).
compute indlbs = {indlbs; 'Omnibus'}.
end if.
loop i = 1 to nm.
compute op = mname(1,i).
print op/title = 'Relative indirect effect(s) of X on
Y through:.'/format = A8/space=0.
do if (method = 2).
compute clbs = {'Effect', 'SE MC', 'MC LLCI',
'MC ULCI'}.
compute op = {indeff, bootse(1:(nm*nx)), llci}.
end if.
do if (method = 1).
compute clbs = {'Effect', 'SE(boot)', 'LLCI',
'ULCI'}.
compute op = indeff.
do if (samples > 0).
compute op = {indeff, bootse(1:(nm*nx).:),
llci(1:(nm*nx).:)};.
end if.
end if.
compute temp = op((((i-1)*nx)+1):(nx*i).:).
do if ((method = 1) and omnibus = 1).
do if (samples = 0).
compute temp={temp;omni(i,1)}.
end if.

```

```

do if (samples > 0).
compute temp = {temp; omni(i,1),
bootse(((nm*nx)+i,:), llci(((nm*nx)+i,:),:))}.
end if.
end if.
print temp/title = ' '/rnames = indlbs/cnames =
clbs/format = F10.4 /space=0.
print/title = '-----'.
end loop.
end if.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
END IF.
DO IF (mdpbe=1).
compute nm={yname,xname,mnames}.
do if (ncovs > 0).
compute
nm={yname,cnames,xname,mnames}.
end if.
compute tpx=xname.
compute x = dd(:,2:ncol(dd)).
compute tpy = yname.
compute tempy= dd(:,1).
compute n = nrow(dd).
compute nx=1.
compute criterr=0.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute ddd1 = {'int_1', 'int_2', 'int_3', 'int_4',
'int_5', 'int_6', 'int_7', 'int_8', 'int_9'}.

```

```

compute dumok = 0.
compute mcfoc=mcx.
compute mcmmod=mcm.
compute mcloc=(mcfoc>0).
compute con = make(n,1,1).
compute ncovs=ncol(x)-2.
do if (mcfoc > 0 or mcmmod > 0).
compute temp = dd.
compute temp(GRADE(dd(:,(ncol(dd)-
mcloc))),:)= dd.
compute dd = temp.
compute dummy = design(dd(:,ncol(dd)-
mcloc)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute toosmall=rsum(nnvls < 2).
compute mnvls = cmin(t(nnvls)).
do if ((rsum(nnvls < 2)) > 0).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=37.
end if.
do if (nvls > 10).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=36.
end if.
do if (criterr=0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,ncol(dd)-mcloc).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,ncol(dd)-mcloc) <> nnvls((temp-
1),1)).
compute nnvls(temp,1)=dd(i,ncol(dd)-mcloc).
compute temp = temp+1.
end if.
end loop.
compute dummy = dummy(:,2:ncol(dummy)).

```

```

do if (mcfoc=4 or mcmmod=4).
compute minus1=make(1,ncol(dummy),-1).
loop k = 1 to n.
do if (rsum(dummy(k,:)) = 0).
compute dummy(k,:)=minus1.
end if.
end loop.
end if.
do if (mcfoc = 2 or mcfoc = 3) or (mcmmod = 2 or
mcmmod = 3)).
loop k = 1 to n.
do if (rsum(dummy(k,:)) > 0).
loop i = 1 to ncol(dummy).
do if (dummy(k,i) = 0).
compute dummy(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcfoc = 3 or mcmmod=3).
compute conmat1={-8,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1; 0,0,0,0,0,-
3,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.
loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute
dummy(k,:)=conmat1((rsum(dummy(k,:))+1),:).
end loop.
end if.
end if.
compute nx = ncol(dummy).
compute xname = ddd(1,1:nx).
compute xdata = dummy.

```



```

compute xname = ddd(1,1:nx).
compute xname2={nm(1,(ncol(nm)-mcloc)),
xname}.
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcfoc = 2 or mcmmod=2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcfoc = 3 or mcmmod=3).
compute dummat=conmat1.
end if.
do if (mcfoc = 4 or mcmmod = 4).
compute dummat(1,:)=minus1.
end if.
compute dummat={nnvls, dummat}.
end if.
end if.
do if (criterr=0).
compute y = dd(:,1).
compute ovals = ncol(design(dd(:,1))).
compute itprob = 0.
do if (ovals = 2).
compute omx = cmax(y(:,1)).
compute omn = cmin(y(:,1)).
compute y(:,1) = (y(:,1) = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute sstotal = csum((y-(csum(y)/n))&**2).
compute outv = t(nm(1,1)).
compute mdtr = nm(1,ncol(dd)).
compute fciv = nm(1,(ncol(dd)-1)).
do if (mcfoc <> 0 or mcmmod <> 0).
compute inter=make(n,nx,0).
loop i = 1 to nx.

```

```

compute inter(:,i)=dummy(:,i)*dd(:,ncol(dd)-
(1-mcloc)).
end loop.
do if (ncovs = 0).
compute x={con,dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y)))}.
end if.
do if (ncovs > 0).
compute
x={con,dd(:,2:(1+ncovs)),dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={t(nm(1,2:(1+ncovs)));
nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y)))}.
end if.
end if.
compute xmns = csum(x)/n.
compute focvals = {1,0,0}.
compute highwarn = 0.
compute lowwarn = 0.
do if (ncol(x) > (2*nx+2)).
compute covmns = {1, xmns(1,2:(ncol(x)-1-
(2*nx)))}.
compute focvals = {covmns,0,0}.
end if.
compute dfres = n-ncol(x).
do if (ovals = 2).
compute pt2 = make(n,1,(csum(y)/n)).
compute LL3 = y*&ln(pt2)+(1-y)*&ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(x),1,0).
compute LL1 = 0.

```

```

.
compute pt1lp= pt1.
compute xlp= x.
compute ylp= y.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)*&ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.

```

```

end if.
compute tval=xp2*xp2.
else if (ovals <> 2).
compute invXtX=inv(t(x)*x).
compute b = invXtX*t(x)*y.
compute resid = y-(x*b).
compute ssresid=csum(resid&*resid).
compute msresid=ssresid/dfres.
compute varb = msresid*invXtX.
compute
seint=sqrt(varb(nrow(varb),nrow(varb))).
compute k3 = nrow(b).
do if (hc3 = 1).
compute xhc=x.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)= xhc(i3,:)*invXtX*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).
end loop.
compute varb=(invXtX*t(xhc)*xhc*invXtX).
end if.
compute seb = sqrt(diag(varb)).
compute r2 = 1-(ssresid/sstotal).
do if (mcfoc <> 0 or mcmmod <> 0).
compute ytp=y.
compute xtp=x(:,1:(ncol(x)-nx)).
compute btp = inv(t(xtp)*xtp)*t(xtp)*ytp.
compute residt = ytp-(xtp*btp).
compute ssresidt=csum(residt&*residt).
compute r2noint = 1-(ssresidt/sstotal).
end if.
compute pr = ncol(x)-1.
compute lmat = ident(nrow(b)).
compute lmat = lmat(:,2:ncol(lmat)).
compute f =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
pr).

```

```

compute pf = 1-fcdf(f,pr,dfres).
compute pf = {sqrt(r2),r2,msresid,f,pr,dfres,pf}.
compute xd=abs(xp2).
end if.
do if (itprob = 0).
compute tstat = b&/seb.
do if (ovals <> 2).
compute p = 2*(1-tcdf(abs(tstat), dfres)).
else if (ovals = 2).
compute p = 2*(1-cdfnorm(abs(tstat))).
end if.
compute outp = {b,seb,tstat,p}.
compute nms = {'constant'; nms; 'interact'}.
do if (ovals <> 2).
compute tval = (dfres* (exp((dfres-
(5/6))*((xd/(dfres-(2/3)+(.11/dfres)))*(xd/(dfres-
(2/3)+(.11/dfres))))-1)).
compute outp={outp,(b-
sqrt(tval)&*seb),(b+sqrt(tval)&*seb)}.
end if.
compute bb = tval.
do if (dumok = 1).
do if (mcfoc > 0).
print dummat/title = 'Coding of categorical X
variable for analysis: '/cnames = xname2/format
= F5.2.
else if (mcmod > 0).
print dummat/title = 'Coding of categorical M
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
print yname/title =
*****
*****'/rlabels = 'Outcome: '/format
= A8.
do if (ovals = 2).
compute nmsd = {outv, 'Analysis'}.
print rcd/title = 'Coding of binary Y for
analysis: '/cnames = nmsd/format = F10.4.

```

```

compute outp = {outp,(outp(:,1)-
sqrt(tval)&*outp(:,2)),(outp(:,1)+sqrt(tval)&*outp
(:,2))}.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
print outp/title = 'Logistic Regression
Model'/rnames = nms/clabels 'Coeff' 'se' 'Z' 'p'
'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
compute varbtmp=varb.
compute btmp=b.
compute LL2f=LL2.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=x(:,1:(ncol(x)-nx)).
compute bt1 = make(ncol(xlp),1,0).

.

compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).

```

```

compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&*ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL2f.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(LLdiff,nx).
compute rcha={LLdiff,nx,pchi}.
end if.
do if (ovals <> 2).
print pf/title = 'Model Summary'/clabels = 'R' 'R-
sq' 'MSE' 'F' 'df1' 'df2' 'p'/format F10.4.
compute lmat=make(nrow(b),1,0).

```

```

compute lmat(nrow(lmat),1)=1.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
1).
compute
rcha=((b(nrow(b),1)/seint)*(b(nrow(b),1)/seint))*
(1-r2)/dfres.
compute rcha = {rcha, fcha, 1, dfres,
outp((pr+1),4)}.
print outp/title = 'Model'/rnames = nms/clabels
'coeff' 'se' 't' 'p' 'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute rcha=r2-r2noint.
compute lmat=make((nrow(b)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
nx).
compute pvalr2c=1-fcdf(fcha,nx,dfres).
compute rcha = {rcha, fcha, nx, dfres,pvalr2c}.
end if.
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute intkey = {'a', 'b', 'c', 'd', 'e'}.
loop i = 1 to nx.
compute intkey={intkey; ddd1(1,i), ' : ', ddd1(1,i),
' X ', nm(1,ncol(dd)-(1-mcloc))}.
end loop.
compute intkey=intkey(2:nrow(intkey),:).
print intkey/title='Product terms key:.'/format =
a8.
end if.
do if (ovals <> 2).

```



```

print rcha/title = 'R-square increase due to
interaction:.'/clabels 'R2-chng' 'F' 'df1' 'df2'
'p'/format F10.4.
else if (ovals = 2).
print rcha/title = 'Likelihood ratio test for
interaction:.'/clabels 'Chi-sq' 'df' 'p'/format F10.4.
end if.
print/title =
*****
*****!
compute mdvar = x(:,(ncol(x)-1)).
do if (mcfoc = 0 and mcmod = 0).
compute g1 = b((ncol(x)-2),1).
compute g3 = b(ncol(x),1).
compute vg1 = varb((ncol(x)-2),(ncol(x)-2)).
compute vg3 = varb(ncol(x),ncol(x)).
compute covg1g3 = varb((ncol(x)-2), ncol(x)).
end if.
do if (mcfoc > 0 or mcmod > 0).
compute mdvar = x(:,(ncol(x)-(2*nx))).
end if.
compute mdmin = cmin(mdvar).
compute mdmax = cmax(mdvar).
compute fvar = x(:,(ncol(x)-2)).
compute nval = ncol(design(mdvar)).
compute fvmin = cmin(fvar).
compute fvmax = cmax(fvar).
do if (mmodval = 999).
compute mnmd = csum(mdvar)/n.
compute tmp = make(n,1,mnmd).
compute sdmd = sqrt(csum(((mdvar-
tmp)**2))/(n-1)).
compute probeval = {mnmd-sdmd; mnmd;
mnmd+sdmd}.
do if (probeval(1,1) < mdmin).
compute lowwarn = 1.
compute probeval(1,1)=mdmin.
end if.
do if (probeval(nrow(probeval),1) > mdmax).
compute highwarn = 1.

```

```

compute probeval(nrow(probeval),1)=mdmax.
end if.
do if (quantile = 1).
compute tmp = mdvar.
compute tmp(GRADE(mdvar(:,1)),:)= mdvar.
compute mdvar = tmp.
compute
probeval={mdvar(trunc(0.1*n),1);mdvar(trunc(0
.25*n),1);mdvar(trunc(0.5*n),1);mdvar(trunc(0.7
5*n),1);mdvar(trunc(0.9*n),1)}.
end if.
end if.
do if (nval = 2).
compute probeval = make(2,1,0).
compute probeval(1,1)=cmin(mdvar).
loop i = 1 to n.
do if (mdvar(i,1) <> probeval(1,1)).
compute probeval(2,1) = mdvar(i,1).
BREAK.
end if.
end loop.
end if.
do if (mmodval <> 999).
compute probeval = mmodval.
end if.
compute outp = make(nrow(probeval),7,0).
do if (mcfoc > 0 or mcmmod > 0).
compute focvals=make(1,ncol(x)+1,1).
do if (mcfoc > 0).
print/title = 'Conditional effect of X on Y at
values of the moderator:'.
print/title = ' '/space=0.
compute rnn2=mdtr.
compute matt=make(nx,6,0).
end if.
loop jj=1 to nrow(probeval).
do if (mcfoc > 0).
loop ii=1 to nx.
compute g1=b((ncol(x)-(2*nx)+ii),1).
compute g3=b((ncol(x)-nx+ii),1).

```

```

compute vg1=varb((ncol(x)-(2*nx)+ii),(ncol(x)-
(2*nx)+ii)).
compute vg3=varb((ncol(x)-nx+ii),(ncol(x)-
nx+ii)).
compute covg1g3=varb((ncol(x)-
(2*nx)+ii),(ncol(x)-nx+ii)).
compute x2 = probeval(jj,1).
compute w1 = g1+g3*x2.
compute varw1 =
vg1+(2*x2*covg1g3)+((x2*x2)*vg3).
compute sew1 = sqrt(varw1).
compute t1 = w1/sew1.
do if (ovals <> 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-tcdf(abs(t1), dfres)).
compute cnms = {'Coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)={w1,sew1,t1,p,llci,ulci}.
else if (ovals = 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-cdfnorm(abs(t1))).
compute cnms = {'Coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)={w1,sew1,t1,p,llci,ulci}.
end if.
end loop.
compute rnms=t(xname).
compute mdvalpr=probeval(jj,1).
print mdvalpr/title = 'Moderator
value:.'/rnames=rnn2/format= F10.4 /space=0.
print matt/title='
'/cnames=cnms/rnames=rnms/format= F10.4
/space=0.
compute xprob=x(:,(ncol(x)-(2*nx)))-mdvalpr.
loop kk = 1 to nx.
compute xprob={xprob,
(xprob(:,1)&*x(:,(ncol(x)-(2*nx)+kk)))}.
end loop.

```

```

compute xprob={x(:,1:((ncol(x)-(2*nx)-
1)),xprob}.
do if (ovals <> 2).
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute residc = y-(xprob*bmultc).
compute ssresidc=csum(residc*residc).
compute r2c = r2-(1-(ssresidc/sstotal)).
compute fcha2=(dfres*r2c)/(nx*(1-r2)).
do if (hc3 = 1).
loop kk = 1 to nx.
compute xprob={xprob, x(:,(ncol(x)-
(2*nx)+kk))}.
end loop.
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute k3 = nrow(bmultc).
compute xhc=xprob.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)=
xhc(i3,)*inv(t(xprob)*xprob)*t(xhc(i3,)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).
end loop.
compute
varbc=(inv(t(xprob)*xprob)*t(xhc)*xhc*inv(t(xpr
ob)*xprob)).
compute lmat=make((nrow(bmultc)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha2 =
(t(t(lmat)*bmultc)*inv(t(lmat)*varbc*lmat)*((t(lm
at)*bmultc)))/nx).
end if.
compute pvalr2cc=1-fcdf(fcha2,nx,dfres).
compute rcha2 = {r2c, fcha2, nx,
dfres,pvalr2cc}.

```

```

print rcha2/title = 'Test of equality of conditional
means at this value of the moderator'/clabels
'R2-chng' 'F' 'df1' 'df2' 'p'/format F10.4.
else if (ovals = 2).
compute btmp=b.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=xprob.
compute bt1 = make(ncol(xlp),1,0).

.

compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*ln(pt1lp)+(1-ylp)*ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).

```

```

compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL4.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(LLdiff,nx).
compute rcha2 = {LLdiff, nx,pchi}.
print rcha2/title = 'Test of equality of log odds
conditioned on this moderator value'/clabels
'Chi-sq' 'df' 'p'/format F10.4.
end if.
end if.
compute
tttt=make(nrow(dummat),1,probeval(jj,1)).
compute tttt={tttt,dummat(:,2:ncol(dummat))}.
loop kkk=1 to nx.
compute tttt={tttt,tttt(:,1)*tttt(:,1+kkk)}.
end loop.
compute ones=make(nrow(dummat),1,1).
do if (ncol(x) > (2*nx+2)).
compute
covmnmat=make(nrow(tttt),ncol(covmns),0).
loop kkk=1 to nrow(tttt).
compute covmnmat(kkk,:)=covmns.
end loop.
compute tttt={covmnmat,tttt}.
else.
compute tttt={ones,tttt}.
end if.
compute focvals={focvals;dummat(:,1),tttt}.
do if (mcfoc > 0).
compute yhat={dummat(:,1),(tttt*b)}.
compute cmnms={fciv, 'yhat'}.
do if (ovals <> 2).

```

```

print yhat/title = 'Estimated conditional means
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (ovals = 2).
print yhat/title = 'Estimated conditional log odds
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (jj <> nrow(probeval)).
print/title='-----'/space=0.
end if.
end if.
end loop.
compute focvals=focvals(2:nrow(focvals),:).
compute yhat=focvals(:,2:ncol(focvals))*b.
compute
focvals={focvals(:,1),focvals(:,(ncol(focvals)-
(2*nx))),yhat}.
compute cnms={fciv,mdtr,'yhat'}.
do if (mcmod > 0).
compute cnms={mdtr,fciv,'yhat'}.
end if.
end if.
do if (mcmod > 0).
compute outp=make((nx+1),7,0).
compute bcatm={b((ncol(x)-
(2*nx)),1);b((ncol(x)-nx+1):ncol(x),1)}.
compute outp(:,1)=dummat(:,1).
compute bcatcov=varb((ncol(x)-
nx):ncol(x),(ncol(x)-nx):ncol(x)).
compute bcatcov(1,1)=varb((ncol(x)-
(2*nx)),(ncol(x)-(2*nx))).
compute
bcatcov(2:nrow(bcatcov),1)=varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx)).
compute
bcatcov(1,2:nrow(bcatcov))=t(varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx))).
loop i = 1 to nrow(dummat).

```

```

compute
catmval={1,dummat(i,2:ncol(dummat))}.
compute condeff=catmval*bcatm.
compute
condse=sqrt(catmval*bcatcov*t(catmval)).
compute outp(i,2:3)={condeff,condse}.
end loop.
compute outp(:,4)=outp(:,2)&/outp(:,3).
compute outp(:,5) = 2*(1-tcdf(abs(outp(:,4)),
dfres)).
compute outp(:,6) = (outp(:,2)-
sqrt(tval)&*outp(:,3)).
compute outp(:,7) =
(outp(:,2)+sqrt(tval)&*outp(:,3)).
do if (ovals <> 2).
compute cnmms = {xname2(1,1), 'coeff', 'se',
't', 'p', 'LLCI', 'ULCI'}.
end if.
do if (ovals = 2).
compute outp(:,5) = 2*(1-
cdfnorm(abs(outp(:,4)))).
compute cnmms = {xname2(1,1), 'coeff', 'se',
'Z', 'p', 'LLCI', 'ULCI'}.
end if.
print outp/title = 'Conditional Effect of Focal
Predictor in Groups Defined by the Moderator
Variable: '/cnames=cnmms/format = F10.4.
end if.
do if (nval > 2 and (mmodval = 999) and
mcmmod = 0).
do if (quantile <> 1).
print/title = 'Moderator values are the sample
mean and plus/minus one SD from mean'.
else.
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
end if.
do if (highwarn = 1 and quantile = 0).

```



```

print/title = 'NOTE: For the moderator values
above, one SD above the mean was'.
print/title = 'replaced with the maximum
because one SD above the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
do if (lowwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD below the mean was'.
print/title = 'replaced with the minimum
because one SD below the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
end if.
do if (nval = 2 and (mmodval = 999)).
print/title = 'The moderator variable is
dichotomous'.
end if.
do if (mmodval <> 999 and mcmmod = 0 and
((mmodval < mdmin) or (mmodval > mdmax))).
print/title = 'Warning: Moderator value specified
is outside of the range of the data'.
end if.
do if (plot = 1).
do if (mcfoc = 0 and mcmmod=0).
compute cnms = {t(nms((ncol(x)-2):(ncol(x)-
1),1)), 'yhat'}.
end if.
do if (ovals = 2).
compute prob =
exp(focvals(:,3))/(1+exp(focvals(:,3))).
compute focvals = {focvals, prob}.
compute cnms = {cnms, 'prob'}.
end if.
print/title =
*****
*****!

```

```
print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (ovals = 2).
do if (mcfoc=0).
print/title = 'DATA LIST FREE/Eco_Innovation
Opra_Pract logodds prob.' /space=1.
end if.
do if (mcmod=0).
print/title = 'DATA LIST FREE/Opra_Pract
Eco_Innovation logodds prob.' /space=1.
end if.
else if (ovals <> 2).
do if (mcfoc=0).
print/title = 'DATA LIST FREE/Eco_Innovation
Opra_Pract Operational_Performance.'
/space=1.
end if.
do if (mcmod=0).
print/title = 'DATA LIST FREE/Opra_Pract
Eco_Innovation Operational_Performance.'
/space=1.
end if.
end if.
print/title = 'BEGIN DATA.' /space=0.
print focvals/title = ' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
```

```

do if (ovals <> 2 and mcmmod=0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Operational_Performance BY
Opra_Pract.' /space=0.
end if.
do if (ovals <> 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
Operational_Performance BY Eco_Innovation.'
/space=0.
end if.
do if (ovals = 2 and mcmmod=0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Opra_Pract.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Opra_Pract.' /space=0.
end if.
do if (ovals = 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
logodds BY Eco_Innovation.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
prob BY Eco_Innovation.' /space=0.
end if.
do if (ncovs > 0).
print/title = 'NOTE: For data above, covariates
are set to their sample means.'.
end if.
end if.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.

```

```

compute runerrs(errs,1) = 22.
end if.
end if.
END IF.
end if.
do if (bad > 0).
compute note(notes,1) = 9.
compute notes = notes + 1.
end if.
print/title = '***** ANALYSIS
NOTES AND WARNINGS
*****'.

loop i = 1 to errs.
do if (runerrs(i,1) = 1).
print/title = 'ERROR: One of your declared
mediators is dichotomous. This procedure
can"t be used.'.
end if.
do if (runerrs(i,1) = 2).
print/title = 'ERROR: For model 6, this
procedure limits the number of mediators to
four.'.
end if.
do if (runerrs(i,1) = 3).
print/title = 'ERROR: For models 1, 2, and 3,
only a single variable can be listed in the M
list.'.
end if.
do if (runerrs(i,1) = 4).
print/title = 'ERROR: You requested a model
involving W but did not provide a valid W
variable name.'.
end if.
do if (runerrs(i,1) = 5).
print/title = 'ERROR: You requested a model
involving Z but did not provide a valid Z
variable name.'.
end if.
do if (runerrs(i,1) = 6).

```

```

print/title = 'ERROR: You requested a model
involving Q but did not provide a valid Q
variable name.'.
end if.
do if (runerrs(i,1) = 7).
print/title = 'ERROR: You requested a model
involving V but did not provide a valid V
variable name.'.
end if.
do if (runerrs(i,1) = 8).
print/title = 'ERROR: You specified a W
variable for a model that does not need it'.
end if.
do if (runerrs(i,1) = 9).
print/title = 'ERROR: You specified a Z variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 10).
print/title = 'ERROR: You specified a Q variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 11).
print/title = 'ERROR: You specified a V variable
for a model that does not need it.'.
end if.
do if (runerrs(i,1) = 12).
print/title = 'ERROR: The variable specified for
W has already been assigned.'.
end if.
do if (runerrs(i,1) = 13).
print/title = 'ERROR: The variable specified for
Z has already been assigned.'.
end if.
do if (runerrs(i,1) = 14).
print/title = 'ERROR: The variable specified for
Q has already been assigned.'.
end if.
do if (runerrs(i,1) = 15).
print/title = 'ERROR: The variable specified for
V has already been assigned.'.

```

```
end if.  
do if (runerrs(i,1) = 16).  
print/title = 'ERROR: You did not provide a  
valid Y variable name.'  
end if.  
do if (runerrs(i,1) = 17).  
print/title = 'ERROR: The variable specified for  
Y has already been assigned.'  
end if.  
do if (runerrs(i,1) = 18).  
print/title = 'ERROR: Model 6 requires more  
than one mediator.'  
end if.  
do if (runerrs(i,1) = 19).  
print/title = 'ERROR: You have not specified a  
valid model number.'  
end if.  
do if (runerrs(i,1) = 20).  
print/title = 'ERROR: At least one and only one  
variable must be listed for X.'  
end if.  
do if (runerrs(i,1) = 21).  
print/title = 'ERROR: At least one and only one  
variable must be listed for Y.'  
end if.  
do if (runerrs(i,1) = 22).  
print/title = 'ERROR: Iteration didn't converge  
to a solution. Interpret results with caution.'  
end if.  
do if (runerrs(i,1) = 23).  
print/title = 'ERROR: You specified a clustering  
variable that does not exist in your variable  
list.'  
end if.  
do if (runerrs(i,1) = 24).  
print/title = 'ERROR: You specified a clustering  
variable that has already been assigned.'  
end if.  
do if (runerrs(i,1) = 25).
```

```

print/title = 'ERROR: One or more of your M
variables is not listed in the variables list.'.
end if.
do if (runerrs(i,1) = 26).
print/title = 'ERROR: A maximum of 20 cluster
units is allowed. Use multilevel modeling
instead.'.
end if.
do if (runerrs(i,1) = 27).
print/title = 'ERROR: One of the variables in
your model is a constant.'.
end if.
do if (runerrs(i,1) = 28).
print/title = 'ERROR: Dichotomous Y is not
permitted with WS option.'.
end if.
do if (runerrs(i,1) = 29).
print/title = 'ERROR: Insufficient number of
variables in vars= list when using WS option.'.
end if.
do if (runerrs(i,1) = 30).
print/title = 'ERROR: Too many variables in
vars= list when using WS option. Covariates
not allowed.'.
end if.
do if (runerrs(i,1) = 31).
print/title = 'ERROR: mmodval and wmodval
can"t both be set to zero with contrast option.'.
end if.
do if (runerrs(i,1) = 32).
print/title = 'ERROR: You did not provide a
valid X variable name.'.
end if.
do if (runerrs(i,1) = 33).
print/title = 'ERROR: PROCESS requires all
variable names to be eight characters or
fewer.'.
print/title = '      Please shorten variable names
and reexecute.'/space=0.
end if.

```

```

do if (runerrs(i,1)=34).
print/title = 'ERROR: X and M cannot both be
specified as multicategorical.'.
end if.
do if (runerrs(i,1)=35).
print/title = 'ERROR: Multicategorical variable
specification available only for models 1 and
4.'.
end if.
do if (runerrs(i,1)=36).
print/title = 'ERROR: Categorical variables
cannot have more than 10 categories.'.
end if.
do if (runerrs(i,1)=37).
print/title = 'ERROR: Each group must have at
least two cases.'.
end if.
do if (runerrs(i,1)=38).
print/title = 'ERROR: Cluster option not
available with multicategorical variables.'.
end if.
do if (runerrs(i,1)=39).
print/title = 'ERROR: A maximum of 10
variables is allowed in the M= list.'.
end if.
do if (runerrs(i,1) = 41).
print/title = 'ERROR: MCM option not permitted
in model 4.'.
end if.
end loop.
do if (errs = 0).
do if (boot > 1 or mc > 0).
do if (bconoff = 1 and boot > 1).
print boot/title = 'Number of bootstrap samples
for bias corrected bootstrap confidence
intervals: '/format = F8.0.
end if.
do if (bconoff = 0 and boot > 1).

```



```

print boot/title = 'Number of bootstrap samples
for percentile bootstrap confidence
intervals: '/format = F8.0.
end if.
do if (mc > 1).
print mc/title = 'Number of samples for Monte
Carlo confidence intervals: '/format = F8.0.
end if.
do if (booterr = 1).
compute badend = badend(1,2:ncol(badend)).
print badend/title = 'WARNING: Bootstrap CI
endpoints below not trustworthy. Decrease
confidence or increase bootstraps' /format =
F10.4.
end if.
end if.
print conf/title = 'Level of confidence for all
confidence intervals in output: '/format = F8.2.
do if ((center = 1 or ws=1) and (ncol(centvar) >
1)).
compute centvar = centvar(1,2:ncol(centvar)).
print centvar/title = 'NOTE: The following
variables were mean centered prior to
analysis: '/format = a8.
end if.
loop i = 1 to notes.
do if (note(i,1) = 1).
print/title = 'NOTE: Confidence level restricted
to between 50 and 99.9999%. 95% confidence
is provided in output'.
end if.
do if (note(i,1) = 2).
print/title = 'NOTE: Effect size measures for
indirect effects are not available for models
with dichotomous outcomes'.
end if.
do if (note(i,1) = 3).
print/title = 'NOTE: All standard errors for
continuous outcome models are based on the
HC3 estimator'.

```

```

end if.
do if (note(i,1) = 6).
print/title = 'NOTE: The number of bootstrap
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 7).
print/title = 'NOTE: The Johnson-Neyman
method is available only for Models 1 and 3'.
end if.
do if (note(i,1) = 8).
print/title = 'NOTE: The Johnson-Neyman
method cannot be used with a dichotomous
moderator'.
end if.
do if (note(i,1)=9).
print bad/title = 'NOTE: Some bootstrap
samples had to be replaced. The number of
such replacements was:'.
end if.
do if (note(i,1) = 11).
print nmiss/title = 'NOTE: Some cases were
deleted due to missing data. The number of
such cases was:'.
end if.
do if (note(i,1) = 12).
print /title = 'NOTE: Monte Carlo method
available only for models 4 and 5.
Bootstrapping was used instead.'.
end if.
do if (note(i,1) = 13).
print/title = 'NOTE: The number of Monte Carlo
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 19).
print/title = 'NOTE: Effect sizes not available for
within-subject analyses.'.
end if.
do if (note(i,1) = 16).

```

```
print/title = 'NOTE: Normal theory tests not
available for within-subject analyses.'.
end if.
do if (note(i,1) = 17).
print/title = 'NOTE: Monte Carlo confidence
intervals not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 18 and warnrep=0).
print/title = 'WARNING: You have requested
OLS estimation with a dichotomous criterion.'.
print/title = 'Interpret model coefficients and
inferential statistics with caution.'/space=0.
compute warnrep=1.
end if.
do if (note(i,1) = 20).
print/title = 'NOTE: Saving of bootstrap
estimates not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 22).
print/title = 'NOTE: Effect size option with
covariates requires covariates in models of M
and Y.'.
end if.
do if (note(i,1) = 23).
print/title = 'NOTE: Johnson-Neyman method
not available with multicategorical X'.
end if.
do if (note(i,1) = 24).
print/title = 'NOTE: MMODVAL option not
available with a multicategorical moderator'.
end if.
do if (note(i,1) = 25).
print/title = 'NOTE: Omnibus test for relative
indirect effects is not available with the MC
option.'.
end if.
do if (note(i,1) = 26).
```

| | | |
|-----------|----------------|---|
| | | <pre> print/title = 'NOTE: CONTRAST option not available with multicategorical X.'. end if. do if (note(i,1) = 27). print/title = 'NOTE: EFFSIZE option not available with multicategorical X.'. end if. do if (note(i,1) = 28). print/title = 'NOTE: NORMAL option not available with multicategorical X.'. end if. do if (note(i,1) = 31). print/title = 'NOTE: The TOTAL option is not available when Y is dichotomous.'. end if. do if (note(i,1) = 32). print/title = 'NOTE: Kappa-squared is disabled from output as of version 2.16.'. end if. do if (note(i,1) = 33). print/title = 'NOTE: COVMY option ignored in models 1, 2, and 3.'. end if. end loop. end if. end matrix. </pre> |
| Resources | Processor Time | 00:00:04.58 |
| | Elapsed Time | 00:00:04.67 |

Run MATRIX procedure:

***** PROCESS Procedure for SPSS Release 2.16.3 *****

Written by Andrew F. Hayes, Ph.D. www.afhayes.com
Documentation available in Hayes (2013). www.guilford.com/p/hayes3

Model = 4
Y = Operatio
X = Opra_Pra
M = Eco_Inno

Sample size

Outcome: Eco_Inno

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|-----------|--------|----------|-------|
| .8594 | .7385 | .3016 | 2411.8474 | 1.0000 | 854.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|-------|-------|---------|-------|-------|-------|
| constant | .5645 | .0677 | 8.3404 | .0000 | .4317 | .6974 |
| Opra_Pra | .8185 | .0167 | 49.1106 | .0000 | .7858 | .8512 |

Outcome: Operatio

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|-----------|--------|----------|-------|
| .8477 | .7186 | .3376 | 1089.0820 | 2.0000 | 853.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|-------|-------|---------|-------|-------|-------|
| constant | .4278 | .0745 | 5.7441 | .0000 | .2816 | .5740 |
| Eco_Inno | .2980 | .0362 | 8.2315 | .0000 | .2270 | .3691 |
| Opra_Pra | .5662 | .0345 | 16.4178 | .0000 | .4985 | .6339 |

***** DIRECT AND INDIRECT EFFECTS *****

Direct effect of X on Y

| Effect | SE | t | p | LLCI | ULCI |
|--------|-------|---------|-------|-------|-------|
| .5662 | .0345 | 16.4178 | .0000 | .4985 | .6339 |

Indirect effect of X on Y

| | Effect | Boot SE | BootLLCI | BootULCI |
|----------|--------|---------|----------|----------|
| Eco_Inno | .2439 | .0207 | .2050 | .2861 |

Normal theory tests for indirect effect

| Effect | se | Z | p |
|--------|-------|--------|-------|
| .2439 | .0301 | 8.1166 | .0000 |

***** ANALYSIS NOTES AND WARNINGS *****

Number of bootstrap samples for bias corrected bootstrap confidence intervals:

5000

Level of confidence for all confidence intervals in output:

95.00

----- END MATRIX -----

restore.

Matrix

| Notes | | |
|----------------|--------------------------------|----------------------|
| Output Created | | 20-OCT-2020 15:58:43 |
| Comments | | |
| Input | Active Dataset | DataSet1 |
| | Filter | <none> |
| | Weight | <none> |
| | Split File | <none> |
| | N of Rows in Working Data File | 856 |

Syntax

```
matrix.  
get dat/file = */variables =  
Operational_Performance Comm_Pract  
Eco_Innovation /names = vnames/missing =  
9999.  
compute ninit = nrow(dat).  
get dat/file = */variables =  
Operational_Performance Comm_Pract  
Eco_Innovation /names = vnames/missing =  
omit.  
get tmp/file = */variables =  
Operational_Performance /names =  
yname/missing = omit.  
get tmp2/file = */variables = Comm_Pract  
/names = xname/missing = omit.  
get tmp/file = */variables = Eco_Innovation  
/names = mnames/missing = omit.  
get tmp/file = */variables = w999999t z999999t  
v999999t q999999t.  
compute wname=tmp(1,1).  
do if (wname = ' ').  
compute wname = 'xxx'.  
end if.  
compute zname=tmp(1,2).  
do if (zname = ' ').  
compute zname = 'xxx'.  
end if.  
compute vname=tmp(1,3).  
do if (vname = ' ').  
compute vname = 'xxx'.  
end if.  
compute qname=tmp(1,4).  
do if (qname = ' ').  
compute qname = 'xxx'.  
end if.  
compute n = nrow(dat).  
compute p0=-.322232431088.  
compute p1 = -1.  
compute p2 = -.342242088547.  
compute p3 = -.0204231210245.
```

```

compute p4 = -.0000453642210148.
compute q0 = .0993484626060.
compute q1 = .588581570495.
compute q2 = .531103462366.
compute q3 = .103537752850.
compute q4 = .0038560700634.
compute badend = 0.
compute priorlo = -9999999.
compute priorhi = 9999999.
compute criterr = 0.
compute cluster = 0.
compute clsdmy = 0.
compute jndich = 0.
compute wvdich=0.
compute zqdich=0.
compute mod74dic=0.
compute warnrep=0.
compute contrast = trunc( 0 ).
do if (contrast > 2 or contrast < 0).
compute contrast = 0.
end if.
compute booterr = 0.
compute effsize = ( 0 = 1).
compute note = make(10,1,0).
compute notes = 1.
compute iterr = 0.
compute clsmtn = 0.
compute quantile = ( 0 = 1).
compute mmodval= 999.
compute jn = ( 0 = 1).
compute covcoeff=( 0 =1).
compute center = ( 0 = 1).
compute detail = ( 1 = 1).
compute coeffci = ( 1 = 1).
compute conf = 95.
compute longname = ( 1 =1).
compute bconoff=( 0 <> 1).
compute covmy = trunc( 0 ).
do if (covmy < 0 or covmy > 2).
compute covmy = 0.

```



```

end if.
do if (trunc(conf) >= 100 or (trunc(conf) <=
50)).
compute conf = 95.
compute note(notes,1) = 1.
compute notes = notes + 1.
end if.
do if (n < ninit).
compute nmiss = ninit-n.
compute note(notes,1) = 11.
compute notes = notes + 1.
end if.
compute errs = 0.
compute quantd = {0,0,0,0,0,0}.
compute quantc = {0,0,0,0,0,0}.
compute mcheck = 0.
compute ttt = 0.
compute plot = trunc( 0 ).
compute runerrs = make(50,1,0).
compute model = trunc( 4 ).
compute normal = 1.
compute ws= ( 0 =1).
compute olsdichm= ( 0 =1).
compute olsdichy= ( 0 =1).
compute mcx=trunc( 0 ).
compute mcm=trunc( 0 ).
do if (ws = 1).
do if (effsize = 1).
compute note(notes,1) = 19.
compute notes = notes + 1.
compute effsize=0.
end if.
do if (normal=1).
compute note(notes,1) = 16.
compute notes = notes + 1.
compute normal=0.
end if.
end if.
do if (model <> 4).
compute ws = 0.

```

```

end if.
do if (model > 0 and model < 4 and covmy <>
0).
compute covmy=0.
compute note(notes,1) = 33.
compute notes = notes + 1.
end if.
print/space=0.
do if (jn = 1 and model <> 1 and model <> 3).
compute note(notes,1) = 7.
compute notes = notes + 1.
end if.
do if ((model > 76) or (model < 1)).
compute model = 77.
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 19.
end if.
compute itprob=0.
do if (mcx < 0).
compute mcx=0.
end if.
do if (mcx > 4).
compute mcx=0.
end if.
do if (mcm < 0).
compute mcm=0.
end if.
do if (mcm > 4).
compute mcm=0.
end if.
do if (mcx > 0 and mcm > 0).
compute errs=errs+1.
compute runerrs(errs,1)=34.
compute criterr=1.
end if.
do if ((model <> 1 and model <> 4) and (mcx >
0 or mcm > 0)).
compute errs=errs+1.
compute runerrs(errs,1)=35.

```

```
compute criterr=1.  
end if.  
do if ((model = 4) and (mcm > 0)).  
compute errs=errs+1.  
compute runerrs(errs,1)=41.  
compute criterr=1.  
end if.  
do if ((mcx > 0) and jn <> 0).  
compute jn=0.  
compute note(notes,1) = 23.  
compute notes = notes + 1.  
end if.  
do if (mmodval <> 999 and mcm > 0).  
compute note(notes,1) = 24.  
compute notes = notes + 1.  
end if.  
compute toteff = 0.  
compute toteff = (( 0 = 1)*( 4 = 4 or 4 = 6)).  
compute varorder = 2.  
compute hc3 = ( 0 <> 0).  
compute cname = 'xxx'.  
compute centvar = {'xxx'}.  
compute nmeds = ncol(mnames).  
do if (longname=0).
```

```
compute criterr = 1.
```

```
compute errs = errs+1.
```

```
compute runerrs(errs,1) = 33.
```

1

2

3

end if.

compute modelm =

```
{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,6;  
1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,7;  
1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8;  
1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,9;  
1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,10;  
1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,11;  
1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,0,12;  
1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,13;  
0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,14;  
0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,15;  
0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,16;  
0,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,17;  
0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,18;  
0,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,19;  
0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,20;  
1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,21;  
1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,22;  
1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,23;  
1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,0,0,24; 1,1,1,1
```

```
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,25;
```

```
1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,0,0,26;
```

1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,27;
1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,7,28;
1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,7,29;
1,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,30;
1,1,0,1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,31;
1,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,32;
1,1,1,1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,33;
1,1,1,1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,34;
1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,35;
1,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,36;
1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,37;
1,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,38;
1,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,39;
1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,40;
1,0,0,1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,41;
1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,42;
1,0,0,1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,43;
1,0,0,1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,44;
1,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,45;
1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,46;
1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,47;
1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,48;
1,1,0,1,1,0,1,1,0,0,0,

0,0,0,0,0,0,0,0,49;
1,1,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,50;
1,1,1,1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,51;
1,1,1,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,52;
1,1,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,53;
1,1,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,54;
1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,55;
1,1,1,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,56;
1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,57;
1,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,58;
1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,59;
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,60;
1,1,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,61;
1,1,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,62;
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,0,0,0,63;
1,0,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,64;

```

1,0,0,1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,65;
1,0,0,1,0,0,0,0,0,1,0,0,1,0,0,0,0,0,0,66;
1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,67;
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,68;
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,69;
1,0,0,1,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,70;
1,0,0,1,0,0,1,0,0,1,0,0,1,1,1,0,0,0,0,71;
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,1,1,0,0,72;
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,1,1,0,0

,73; 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,74;
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,75;
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,1,0,0,0,76;
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,77).
compute wm = modelm(model, 1).
compute zm = modelm(model, 2).
compute wzm = modelm(model, 3).
compute vy = modelm(model, 4).
compute qy = modelm(model, 5).
compute vqy = modelm(model, 6).
compute wy = modelm(model, 7).
compute zy = modelm(model, 8).
compute wzy = modelm(model, 9).
compute vxy = modelm(model, 10).
compute qxy = modelm(model, 11).
compute vqxy = modelm(model, 12).
compute wmy = modelm(model, 13).
compute wvmy = modelm(model, 14).
compute wvxy = modelm(model, 15).
compute zmy = modelm(model,16).
compute wzmy = modelm(model,17).
compute xmy = modelm(model,18).
compute imm=modelm(model,19).
do if (model = 74).
compute vdich=0.
end if.
do if (ncol(xname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 20.
compute criterr = 1.

```



```

end if.
do if (ncol(mnames) > 10).
compute errs = errs+1.
compute runerrs(errs,1) = 39.
compute criterr = 1.
end if.
do if (ncol(yname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 21.
compute criterr = 1.
end if.
do if (( 999 =0) and ( 999 =0) and (model=3 or
model=2) and (contrast=1)).
compute errs = errs+1.
compute runerrs(errs,1) = 31.
compute criterr = 1.
end if.
compute xlist = (wm or zm or wzm or wy or zy
or wzy or vxy or qxy or vqxy or wvxy or xmy).
compute mlist = (vy or qy or vqy or zmy or wmy
or wzmy or xmy or (model < 4)).
compute bad=0.
do if (criterr = 0).
compute werr = 0.
compute verr = 0.
compute qerr = 0.
compute zerr = 0.
compute yerr = 1.
compute xerr = 1.
compute wlist = (wm or wzm or wy or wzy or
wm or wvmy or wvmy or wvxy or wzmy).
do if (wlist = 1 and wname = 'xxx').
compute werr = 1.
compute wlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (wlist = 1 and ((wname = qname) or
(wname = vname) or (wname = zname) or
(wname = xname) or (wname = yname))).

```

```

compute werr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 12).
end if.
compute zlist = (zm or wzm or zy or wzy or
zmy or wzmy).
do if (zlist = 1 and zname = 'xxx').
compute zerr = 1.
compute zlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (zlist = 1 and ((zname = qname) or
(zname = vname) or (zname = wname) or
(zname = xname) or (zname = yname))).
compute zerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 13.
end if.
compute qlist = (qy or vqy or qxy or vqxy).
do if (qlist = 1 and qname = 'xxx').
compute qerr = 1.
compute qlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (qlist = 1 and ((qname = zname) or
(qname = vname) or (qname = wname) or
(qname = xname) or (qname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 14.
end if.
compute vlist = (vy or vqy or vxy or vqxy or
wvmy or wvxy).
do if (vlist = 1 and vname = 'xxx').
compute verr = 1.
compute vlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 7.

```

```

end if.
do if (vlist = 1 and ((vname = zname) or
(vname = qname) or (vname = wname) or
(vname = xname) or (vname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 15.
end if.
do if (wlist = 0 and wname <> 'xxx').
compute werr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 8.
end if.
do if (zlist = 0 and zname <> 'xxx').
compute zerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 9.
end if.
do if (qlist = 0 and qname <> 'xxx').
compute qerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 10.
end if.
do if (vlist = 0 and vname <> 'xxx').
compute verr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 11.
end if.
do if (hc3 = 1).
compute note(notes,1) = 3.
compute notes = notes+1.
end if.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute cons = make(n,1,1).
compute temp = (n*sscp(dat))-
(t(csum(dat))*(csum(dat))).

```

```

compute temp = temp/(n*(n-1)).
compute temp = csum(((diag(temp))=0)).
compute temp2=1.
do if (ws = 1).
compute temp2 = (n*sscp(tmp2))-
(t(csum(tmp2))*(csum(tmp2))).
compute temp2 = temp2/(n*(n-1)).
end if.
do if ((temp > 0 and ws = 0) or (temp > 0 and
ws = 1 and temp2 > 0)).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 27.
end if.
compute nmeds = ncol(mnames).
compute mcmats=ident(nmeds*2).
compute mccoef=make(nmeds*2,1,0).
compute sobel = make(nmeds,4,-999).
do if (model = 6 and nmeds > 4).
compute errs = errs+1.
compute runerrs(errs,1)=2.
end if.
do if (model < 4 and nmeds > 1).
compute errs = errs+1.
compute runerrs(errs,1) = 3.
end if.
compute nmods = (model = 74).
compute bad = 0.
compute intcnt = 1.
compute modvals = 0.
compute modvalsd = 0.
compute yintemp = {'int_1', 'int_2', 'int_3',
'int_4', 'int_5', 'int_6', 'int_7', 'int_8', 'int_9',
'int_10', 'int_11', 'int_12', 'int_13', 'int_14',
'int_15'}.
compute yintemp = {yintemp, 'int_16', 'int_17',
'int_18', 'int_19', 'int_20', 'int_21', 'int_22',
'int_23', 'int_24', 'int_25', 'int_26', 'int_27',
'int_28'}.

```

```

compute yintemp = {yintemp, 'int_29', 'int_30',
'int_31', 'int_32', 'int_33', 'int_34', 'int_35',
'int_36', 'int_37', 'int_38', 'int_39', 'int_40'}.
compute yintemp = {yintemp, 'int_41', 'int_42',
'int_43', 'int_44', 'int_45', 'int_46', 'int_47',
'int_48', 'int_49', 'int_50', 'int_51', 'int_52'}.
compute yintemp = {yintemp, 'int_53', 'int_54',
'int_55', 'int_56', 'int_57', 'int_58', 'int_59',
'int_60', 'int_61', 'int_62', 'int_63', 'int_64'}.
compute
cntname={('C1'),('C2'),('C3'),('C4'),('C5'),('C6'),('
C7'),('C8'),('C9'),('C10'),('C11'),('C12'),('C13'),('
C14'),('C15'),('C16'),('C17')}.
compute
cntname={cntname;('C18'),('C19'),('C20'),('C21
'),('C22'),('C23'),('C24'),('C25'),('C26'),('C27'),('
C28'),('C29'),('C30'),('C31')}.
compute
cntname={cntname;('C32'),('C33'),('C34'),('C35
'),('C36'),('C37'),('C38'),('C39'),('C40'),('C41'),('
C42'),('C43'),('C44'),('C45')}.
compute
cntname={cntname;('C46'),('C47'),('C48'),('C49
'),('C50'),('C51'),('C52'),('C53'),('C54'),('C55'),('
C56'),('C57'),('C58'),('C59')}.
compute
cntname={cntname;('C60'),('C61'),('C62'),('C63
'),('C64'),('C65'),('C66'),('C67'),('C68'),('C69'),('
C70'),('C71'),('C72'),('C73')}.
compute
cntname={cntname;('C74'),('C75'),('C76'),('C77
'),('C78'),('C79'),('C80'),('C81'),('C82'),('C83'),('
C84'),('C85'),('C86'),('C87')}.
compute
cntname={cntname;('C88'),('C89'),('C90'),('C91
'),('C92'),('C93'),('C94'),('C95'),('C96'),('C97'),('
C98'),('C99'),('C100'),('C101')}.
compute
cntname={cntname;('C102'),('C103'),('C104'),('
C105')}.

```

```

compute apathnam={'a path'; 'a1 path'; 'a2
path'; 'a3 path'; 'a4 path'; 'a5 path'; 'a6 path';
'a7 path'; 'a8 path'; 'a9 path'; 'a10 path'}.
compute bpathnam={'b path'; 'b1 path'; 'b2
path'; 'b3 path'; 'b4path'; 'b5path'; 'b6 path'; 'b7
path'; 'b8 path'; 'b9 path'; 'b10 path'}.
compute modvnm = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute modvnm2 = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute mlab = { ' M1 ='; ' M2 ='; ' M3 ='; '
M4 ='; ' M5 ='; ' M6 ='; ' M7 ='; ' M8 ='; '
M9 ='; ' M10 ='}.
compute m = make(n,nmeds,1).
compute mmat = make(16,nmeds,0).
compute ymat = make(8,nmeds,0).
compute deco = make(10,1,0).
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
compute iterate = abs(trunc( 10000 )).
compute converge = abs( 0.00000001 ).
compute boot = abs(trunc( 5000 )).
do if (ws=1 and 0 > 0).
compute note(notes,1)=17.
compute notes=notes+1.
end if.
compute mc=(abs(trunc( 0 ))*(1-ws)).
do if (mc > 0 and model > 5).
do if (boot = 0).
compute boot=mc.
end if.
compute mc=0.
compute note(notes,1)=12.
compute notes = notes+1.
end if.
compute adjust = 0.
do if (mc > 0 and (model > 3 and model < 6)).
compute boot=0.

```

```

compute bconoff=0.
end if.
do if (boot > 0 and mc > 0).
compute mc = 0.
end if.
compute savboot = 0.
compute saveboot = ( 0 = 1).
do if (ws=1 and saveboot=1).
compute saveboot=0.
compute note(notes,1) = 20.
compute notes = notes + 1.
compute normal=0.
end if.
do if (saveboot =1 and boot > 0 and model >
3).
compute savboot = 1.
end if.
do if (boot <> 0 or mc <> 0).
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
loop.
compute cilow = rnd(bootsz*(1-(conf/100))/2).
compute cihigh =
trunc((bootsz*(conf/100)+(bootsz*(1-
(conf/100))/2)))+1.
do if (cilow < 1 or cihigh > bootsz).
compute
bootsz=trunc((bootsz+1000)/1000)*1000.
compute adjust = 1.
end if.
end loop if (cilow gt 0 and cihigh le bootsz).
compute boot=bootsz.
do if (mc > 0).
compute mc = bootsz.
end if.
do if (boot > 0 and mc > 0 and (model > 3 and
model < 6)).
compute boot=0.

```

```

end if.
do if (adjust = 1 and boot > 0).
compute note(notes,1)=6.
compute notes = notes+1.
end if.
do if (adjust = 1 and mc > 0).
compute note(notes,1)=13.
compute notes = notes+1.
end if.
end if.
do if (model = 6 and nmeds > 1).
compute mmpaths =
make((nmeds+2),(nmeds+2),0).
do if (nmeds = 2).
compute indboot = make(boot+1, 3, 999).
else if (nmeds = 3).
compute indboot = make(boot+1, 7, 999).
else if (nmeds = 4).
compute indboot = make(boot+1, 15, 999).
end if.
compute indlbl = {'Total:','Ind1 :','Ind2 :','Ind3
:','Ind4 :','Ind5 :','Ind6 :','Ind7 :','Ind8 :','Ind9
:','Ind10:','Ind11:','Ind12:','Ind13:','Ind14:','
Ind15:'}.
compute indlbl2 = {'Ind1','Ind2','Ind3','Ind4';
'Ind5','Ind6','Ind7','Ind8','Ind9','Ind10';
'Ind11','Ind12','Ind13','Ind14','Ind15'}.
compute indces = make(boot+1, 4, 999).
end if.
do if (model < 4).
compute boot = 0.
compute cmat = make(10,1,0).
compute zmat = make(10,1,0).
end if.
compute nvarch = make(1,ncol(dat),0).
compute wmatch = 0.
compute zmatch = 0.
compute vmatch = 0.
compute qmatch = 0.
compute mmatch=0.

```



```

compute minprobe=0.
compute maxprobe=0.
loop i = 1 to ncol(vnames).
do if (vnames(:,i)=yname).
compute y = dat(:,i).
compute nvarch(1,i)=1.
compute yerr = 0.
do if ((yname = xname) or (yname = wname) or
(yname = zname) or (yname = vname) or
(yname = qname)).
compute errs = errs+1.
compute runerrs(errs,1)=17.
end if.
end if.
do if (vnames(:,i)=xname).
compute x = dat(:,i).
compute nvarch(1,i)=1.
compute xdich = 1.
compute xerr=0.
loop jj = 1 to n.
do if ((x(jj,1) <> cmin(x)) and (x(jj,1) <>
cmax(x))).
compute xdich = 0.
break.
end if.
end loop.
compute xmean = csum(x)/n.
do if (center = 1 and (model < 4 or xlist > 0)
and mcx = 0).
compute meanvec = make(n,1, xmean).
compute x = x-meanvec.
compute centvar = {centvar, xname}.
end if.
compute xmean = csum(x)/n.
compute tmp = x-(cons*xmean).
compute xsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
do if (xdich = 0).
compute quantc(1,6) = 1.
compute matx = {xmean-xsd; xmean;
xmean+xsd}.

```

```

do if ( 999 = 999 and quantile = 0).
do if ((matx(1,1) < cmin(x)) and model = 74).
compute matx(1,1)=cmin(x).
compute minprobe=1.
end if.
do if ((matx(3,1) > cmax(x)) and model = 74).
compute matx(3,1)=cmax(x).
compute maxprobe=1.
end if.
end if.
do if (quantile = 1).
compute quantd(1,6) = 1.
compute quantc(1,6) = 0.
compute tmp = x.
compute tmp(GRADE(x)) = x.
compute matx =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (xdich = 1).
compute matx = {cmin(x); cmax(x)}.
do if (model=74).
compute matx=cmin(x).
compute mod74dic=1.
end if.
end if.
do if ( 999 <> 999).
compute matx = 999.
compute quantd(1,6) = 0.
compute quantc(1,6) = 0.
end if.
end if.
do if (werr = 0 and wlist = 1).
do if (vnames(:,i)=wname).
compute werr = 0.
compute wmatch = 1.
compute w = dat(:,i).
do if (center = 1).

```

```

compute meanvec = make(n,1,csum(w)/n).
compute w = w-meanvec.
compute centvar = {centvar, wname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute wmean = csum(w)/n.
compute tmp = w-(cons*wmean).
compute wsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute wdich = 1.
loop jj = 1 to n.
do if ((w(jj,1) <> cmin(w)) and (w(jj,1) <>
cmax(w))).
compute wdich = 0.
break.
end if.
end loop.
do if (model = 3).
compute jndich=wdich.
compute jnmin=cmin(w).
compute jnmax=cmax(w).
end if.
do if (wdich = 0).
compute matw = {wmean-wsd; wmean;
wmean+wsd}.
do if ( 999 = 999 and quantile = 0).
do if (matw(1,1) < cmin(w)).
compute matw(1,1)=cmin(w).
compute minprobe=1.
end if.
do if (matw(3,1) > cmax(w)).
compute matw(3,1)=cmax(w).
compute maxprobe=1.
end if.
end if.
compute quantc(1,1) = 1.
do if (quantile = 1).
compute quantd(1,1) = 1.
compute quantc(1,1) = 0.
compute tmp = w.

```

```

compute tmp(GRADE(w)) = w.
compute matw =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (wdich = 1).
compute matw = {cmin(w); cmax(w)}.
compute wvdich=1.
compute cmaxw=cmax(w).
compute cminw=cmin(w).
end if.
do if ( 999 <> 999).
compute matw = 999.
compute quantd(1,1) = 0.
compute quantc(1,1) = 0.
end if.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm(1,1)=wname.
compute modmatp(1,1) = 1.
end if.
end if.
do if (zerr = 0 and zlist = 1).
do if (vnames(:,i)=zname).
compute zerr = 0.
compute zmatch = 1.
compute z = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(z)/n).
compute z = z-meanvec.
compute centvar = {centvar, zname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute zmean = csum(z)/n.
compute tmp = z-(cons*zmean).
compute zsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute zdich = 1.

```

```

loop jj = 1 to n.
do if ((z(jj,1) <> cmin(z)) and (z(jj,1) <>
cmax(z))).
compute zdich = 0.
break.
end if.
end loop.
do if (zdich = 0).
compute matz = {zmean-zsd; zmean;
zmean+zsd}.
do if ( 999 = 999 and quantile = 0).
do if (matz(1,1) < cmin(z)).
compute matz(1,1)=cmin(z).
compute minprobe=1.
end if.
do if (matz(3,1) > cmax(z)).
compute matz(3,1)=cmax(z).
compute maxprobe=1.
end if.
end if.
compute quantc(1,2) = 1.
do if (quantile = 1).
compute quantd(1,2) = 1.
compute quantc(1,2) = 0.
compute tmp = z.
compute tmp(GRADE(z)) = z.
compute matz = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (zdich = 1).
compute matz = {cmin(z); cmax(z)}.
compute cmaxz=cmax(z).
compute cminz=cmin(z).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matz = 999.
compute quantd(1,2) = 0.

```

```

compute quantc(1,2) = 0.
end if.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm(1,2)=zname.
compute modmatp(1,2) = 1.
end if.
end if.
do if (verr = 0 and vlist = 1).
do if (vnames(:,i)=vname).
compute verr = 0.
compute vmatch = 1.
compute v = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(v)/n).
compute v = v-meanvec.
compute centvar = {centvar, vname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute vmean = csum(v)/n.
compute tmp = v-(cons*vmean).
compute vsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute vdich = 1.
loop jj = 1 to n.
do if ((v(jj,1) <> cmin(v)) and (v(jj,1) <>
cmax(v))).
compute vdich = 0.
break.
end if.
end loop.
do if (vdich = 0).
compute matv = {vmean-vsd; vmean;
vmean+vsd}.
do if ( 999 = 999 and quantile = 0).
do if (matv(1,1) < cmin(v)).
compute matv(1,1)=cmin(v).
compute minprobe=1.
end if.
do if (matv(3,1) > cmax(v)).

```

```

compute matv(3,1)=cmax(v).
compute maxprobe=1.
end if.
end if.
compute quantc(1,3) = 1.
do if (quantile = 1).
compute quantd(1,3) = 1.
compute quantc(1,3) = 0.
compute tmp = v.
compute tmp(GRADE(v)) = v.
compute matv = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (vdich = 1).
compute matv = {cmin(v); cmax(v)}.
compute wvdich=1.
compute cmaxv=cmax(v).
compute cminv=cmin(v).
end if.
do if ( 999 <> 999).
compute matv = 999.
compute quantd(1,3) = 0.
compute quantc(1,3) = 0.
end if.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm(1,3)=vname.
compute modmatp(1,3) = 1.
end if.
end if.
do if (qerr = 0 and qlist = 1).
do if (vnames(:,i)=qname).
compute qerr = 0.
compute qmatch = 1.
compute q = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(q)/n).
compute q = q-meanvec.

```

```

compute centvar = {centvar, qname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute qmean = csum(q)/n.
compute tmp = q-(cons*qmean).
compute qsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute qdich = 1.
loop jj = 1 to n.
do if ((q(jj,1) <> cmin(q)) and (q(jj,1) <>
cmax(q))).
compute qdich = 0.
break.
end if.
end loop.
do if (qdich = 0).
compute matq = {qmean-qsd; qmean;
qmean+qsd}.
do if ( 999 = 999 and quantile = 0).
do if (matq(1,1) < cmin(q)).
compute matq(1,1)=cmin(q).
compute minprobe=1.
end if.
do if (matq(3,1) > cmax(q)).
compute matq(3,1)=cmax(q).
compute maxprobe=1.
end if.
end if.
compute quantc(1,4) = 1.
do if (quantile = 1).
compute quantd(1,4) = 1.
compute quantc(1,4) = 0.
compute tmp = q.
compute tmp(GRADE(q)) = q.
compute matq = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (qdich = 1).

```



```

compute matq = {cmin(q); cmax(q)}.
compute cmaxq=cmax(q).
compute cminq=cmin(q).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matq = 999.
compute quantd(1,4) = 0.
compute quantc(1,4) = 0.
end if.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm(1,4)=qname.
compute modmatp(1,4) = 1.
end if.
end if.
do if (vnames(:,i)=cname).
compute cld = dat(:,i).
compute cvname = vnames(:,i).
compute nvarch(1,i)=1.
compute clsmatch = 1.
end if.
loop j = 1 to ncol(mnames).
do if (vnames(:,i)=mnames(1,j)).
compute mmatch = mmatch + 1.
compute m(:,j)=dat(:,i).
do if (center = 1 and nvarch(1,i) = 0 and mlist >
0 and mcm = 0).
compute meanvec = make(n,1,csum(m(:,j))/n).
compute m(:,j) = m(:,j)-meanvec.
compute centvar = {centvar, mnames(1,j)}.
end if.
compute nvarch(1,i)=1.
compute dichm = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute dichm = 0.
break.
end if.

```

```

end loop.
do if ((dichm = 1) and (olsdichm = 1)).
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichm = 1 and model > 3 and mcheck = 0
and oldsdichm = 0).
compute errs= errs+1.
compute runerrs(errs,1) = 1.
compute mcheck = 1.
end if.
do if ((model <= 3) and (ncol(mnames)=1)).
compute nmods = nmods + 1.
compute mmean = csum(m(:,j))/n.
compute tmp = m(:,j)-(cons*mmean).
compute msd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute mdich = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute mdich = 0.
break.
end if.
end loop.
do if (model = 1).
compute jndich=mdich.
compute jnmin=cmin(m(:,j)).
compute jnmax=cmax(m(:,j)).
end if.
do if (mdich = 0).
compute matm = {mmean-msd; mmean;
mmean+msd}.
do if ( 999 = 999 and quantile = 0).
do if (matm(1,1) < cmin(m(:,j))).
compute matm(1,1)=cmin(m(:,j)).
compute minprobe=1.
end if.
do if (matm(3,1) > cmax(m(:,j))).
compute matm(3,1)=cmax(m(:,j)).
compute maxprobe=1.

```

```

end if.
end if.
compute quantc(1,5) = 1.
do if (quantile = 1).
compute quantd(1,5) = 1.
compute quantc(1,5) = 0.
compute tmp = m(:,j).
compute tmp(GRADE(m(:,j))) = m(:,j).
compute matm = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (mdich = 1).
compute matm = {cmin(m); cmax(m)}.
end if.
do if ( 999 <> 999).
compute matm = 999.
compute quantd(1,5) = 0.
compute quantc(1,5) = 0.
end if.
compute modmatv(1,5)=nrow(matm).
compute modmat((1:nrow(matm)), 5) = matm.
compute modvnm(1,5)=mnames(1,j).
compute modmatp(1,5) = 1.
end if.
end if.
end loop.
end loop.
do if (minprobe=1).
compute note(notes,1) = 14.
compute notes = notes + 1.
end if.
do if (maxprobe=1).
compute note(notes,1) = 15.
compute notes = notes + 1.
end if.
do if (cname <> 'xxx' and clsmth = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 23.

```

```

end if.
do if (cname <> 'xxx').
do if ((cname = zname) or (cname = vname)
or (cname = wname) or (cname = xname) or
(cname = yname) or (cname = qname)).
compute errs = errs+1.
compute runerrs(errs,1) = 24.
end if.
end if.
do if (wlist = 1 and werr = 0 and wmatch = 0).
compute werr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (zlist = 1 and zerr = 0 and zmatch = 0).
compute zerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (qlist = 1 and qerr = 0 and qmatch = 0).
compute qerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (vlist = 1 and verr = 0 and vmatch = 0).
compute verr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (yerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 16.
end if.
do if (xerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 32.
end if.
do if (model = 6 and nmeds < 2).
compute errs = errs+1.
compute runerrs(errs,1) = 18.

```

```

end if.
do if (mmatch < ncol(mnames)).
compute errs=errs+1.
compute runerrs(errs,1) = 25.
end if.
do if (cname <> 'xxx').
compute cld = design(cld).
compute cluster = ncol(cld).
compute cld = cld(:,2:ncol(cld)).
compute clsdmy = ncol(cld).
do if (clsdmy > 19).
compute errs = errs+1.
compute runerrs(errs,1) = 26.
end if.
do if (mcx > 0 or mcm > 0).
compute errs = errs+1.
compute runerrs(errs,1) = 38.
end if.
end if.
compute dichy = 1.
loop jj = 1 to n.
do if ((y(jj,1) <> cmin(y)) and (y(jj,1) <>
cmax(y))).
compute dichy = 0.
break.
end if.
end loop.
do if ((dichy = 1) and (olsdichy = 1)).
compute dichy=0.
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichy = 1).
do if (toteff=1).
compute toteff=0.
compute note(notes,1) = 31.
compute notes = notes + 1.
end if.
compute jncrit=xp2*xp2.
do if (ws = 1).

```

```

compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 28.
end if.
end if.
compute ncovs = ncol(dat)-rsum(nvarch).
do if (effsize = 1 and covmy <> 0 and ncovs >
0 and model > 3 and model < 7).
compute note(notes,1) = 22.
compute notes = notes + 1.
compute effsize = 0.
end if.
do if (ws = 1 and ncovs < nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 29.
end if.
do if (ws = 1 and ncovs > nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 30.
end if.
end if.
do if (errs = 0).
do if (rsum(quantd) > 0).
compute note(notes,1) = 4.
compute notes = notes+1.
end if.
do if (rsum(quantc) > 0).
compute note(notes,1) = 5.
compute notes = notes+1.
end if.
do if (ncovs > 0).
compute c = make(n,ncovs,0).
compute cnames = {'x'}.
compute j = 1.
loop i = 1 to ncol(vnames).
do if (nvarch(1,i)) = 0.
compute c(:,j) = dat(:,i).
compute nvarch(1,i)=1.

```

```

compute j=j+1.
compute cnames = {cnames, vnames(:,i)}.
end if.
end loop.
compute cnames = cnames(1,2:ncol(cnames)).
do if (ws = 1).
compute covmean=csum(c)/n.
loop i = 1 to ncovs.
compute meanvec = make(n,1, covmean(1,i)).
compute c(:,i) = c(:,i)-meanvec.
end loop.
compute centvar = {centvar, cnames}.
compute x=c(:,1:ncovs).
do if (ncovs=nmeds).
compute ncovs=0.
end if.
end if.
end if.
compute names = {yname, xname, mnames,
wname, zname, vname, qname}.
do if (ncovs > 0).
compute names = {names, cnames}.
end if.
do if (dichy = 1 and effsize = 1).
compute note(notes,1) = 2.
compute notes = notes+1.
end if.
do if (model > 3 and model < 6).
compute indeff=make(nmeds,1,0).
compute indboot=make(boot+1,nmeds,999).
do if (mc > 0).
compute indboot=make(mc+1,nmeds,999).
end if.
do if (effsize = 1 and dichy = 0).
compute rmeff=make(boot+1,nmeds+1,999).
compute abpseff=make(boot+1,nmeds+1,999).
compute abcseff=make(boot+1,nmeds+1,999).
compute pmeff=make(boot+1,nmeds+1,999).
compute r245 = make(boot+1,1,999).
compute kappa2 = make(boot+1,1,999).

```

```

end if.
end if.
do if (model = 6 and effsize = 1 and dichy = 0).
compute
rmeff=make(boot+1,ncol(indboot),999).
compute
abpseff=make(boot+1,ncol(indboot),999).
compute
abcseff=make(boot+1,ncol(indboot),999).
compute
pmeff=make(boot+1,ncol(indboot),999).
end if.
do if (nmods > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm(1,tmp) = modvnm(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:nmods).
compute modvnm=modvnm(:,1:nmods).
compute modmatv=modmatv(:,1:nmods).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvals =
make((modmatv(1,1)*modprod(1,1)),
nmods,0).
loop i = 1 to nmods.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvals)).
loop j = 1 to modmatv(1,i).

```



```

compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvals(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
do if (model = 74).
compute modvals=matx.
compute modvnm=xname.
end if.
compute vmat = make(8,nrow(modvals),0).
compute vmat(1,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute vmat(5,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute indeff=make(nrow(modvals),1,0).
do if (model <> 5).
compute indboot = make(((boot+1)*nmeds),
nrow(modvals), -99999999).
compute indbootp =
make(boot+1,(nmeds*nmods), -99999999).
do if (imm > 4 and imm < 8).
do if (imm = 7).
compute immmv1=matw.
compute immmv2=matv.
end if.
do if (imm = 5).
compute immmv1=matw.
compute immmv2=matz.
end if.
do if (imm = 6).
compute immmv1=matv.
compute immmv2=matq.
end if.
compute
indboop1=make((boot+1),(nmeds*(nrow(immm
v2))),-99999999).

```

```

compute
indboop2=make((boot+1),(nmeds*(nrow(immmv1))),-9999999).
compute onesmv1=make(1,nrow(immmv1),1).
compute onesmv2=make(1,nrow(immmv2),1).
compute
nimvs=nrow(immmv1)+nrow(immmv2).
compute cmmval=0.
compute cmmlbs1={'xx'}.
compute cmmlbs2={'xx'}.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv2}.
compute
cmmlbt=make(nrow(immmv2),1,mnames(1,i)).
compute cmmlbs1={cmmlbs1;cmmlbt}.
end loop.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv1}.
compute
cmmlbt=make(nrow(immmv1),1,mnames(1,i)).
compute cmmlbs2={cmmlbs2;cmmlbt}.
end loop.
compute cmmval=cmmval(2:nrow(cmmval),:).
end if.
end if.
end if.
do if (nmods> 0).
loop i = 1 to ncol(modvals).
do if (modvnm(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm(1,i)=qname).
compute qcol = i.
end if.

```

```

end loop.
end if.
do if (dichy = 1).
compute omx = cmax(y).
compute omn = cmin(y).
compute y = (y = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute data = {cons,y,m,x}.
compute datamed = data.
compute datayed = data.
compute datanm = {'constant'; yname;
t(mnames); xname}.
compute datanmm = {'constant'; yname;
t(mnames); xname}.
compute datanmy = {'constant'; yname;
t(mnames); xname}.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
do if (model < 4 and errs = 0).
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' ', ' '}.
compute datayed = {datayed, x&*m}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(1,i) = 1.
compute vmat(2,i) = modvals(i,1).
end loop.
compute mmat = make(16,nmeds,1).
end if.
do if (model = 2 or model = 3).
compute int1 = x&*w.
compute datayed = {datayed, w, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).

```

```

compute vmat(2,i) = modvals(i,2).
compute vmat(3,i) = modvals(i,1).
compute vmat(4,i) = modvals(i,1)*modvals(i,2).
end loop.
end if.
do if (model = 3).
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*m.
compute int2 = x&*w&*m.
compute datayed = {datayed, int1, int2}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' X', wname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (model = 4 or model = 5).
compute vmat = make(8,1,1).
end if.
compute yintkey2 = yintkey.
do if (wm = 1).
compute int1 = x&*w.
compute datamed = {datamed, w,int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmm = {datanmm; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,wcol).
end loop.
do if (zm = 1).
compute int1 = x&*z.
compute datamed = {datamed, z, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.

```

```

compute datanmm = {datanmm; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(3,i) = modvals(i,zcol).
end loop.
end if.
do if (wzm = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*z.
compute int2 = x&*w&*z.
compute datamed = {datamed, int1, int2}.
loop i = 1 to nrow(modvals).
compute vmat(4,i) =
modvals(i,wcol)*modvals(i,zcol).
end loop.
end if.
end if.
compute mdatacol=ncol(datamed).
compute mintkey = yintkey.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
compute medints = intcnt-1.
do if (vy = 1 or xmy = 1).
compute mp = 1.
do if (xmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,1).
end loop.
end if.
do if (vy = 1).
compute datayed = {datayed, v}.

```

```

compute datanmy = {datanmy; vname}.
compute mmods = 1.
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,vcol).
end loop.
do if (qy = 1).
compute mp = 2.
compute datayed = {datayed,q}.
compute datanmy = {datanmy; qname}.
compute mmods = 2.
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,qcol).
end loop.
end if.
do if (vqy = 1).
compute mp = 3.
compute datayed = {datayed, v&*q}.
compute mmods = 3.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,vcol)*modvals(i,qcol).
end loop.
end if.
end if.
compute mints = make(n,(nmeds*mp),0).
loop i = 0 to (nmeds-1).
do if (i = 0 and vqy = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
vname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.

```

```

end if.
do if (xmy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))&*x.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', xname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (qy = 1).
compute mints(:,((i*mp)+2))=m(:,(i+1))&*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqy = 1).
compute mints(:,((i*mp)+3))=m(:,(i+1))&*v&*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end loop.
compute datayed = {datayed, mints}.
end if.
compute mp = 1.
do if (wvmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,wcol)*modvals(i,vcol).
end loop.
end if.
compute mints2 = make(n,(nmeds*mp),0).
do if (wmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,wcol).

```

```

end loop.
do if (wy = 0 and model > 3).
compute datayed = {datayed, w}.
compute datanmy = {datanmy; wname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wvmy = 1).
compute datayed = {datayed,w&*v}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints2(:,((i*mp)+1))= m(:,(i+1))&*w.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wvmy = 1).
compute mints2(:,((i*mp)+2))=
m(:,(i+1))&*w&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints2}.
end if.
compute mp = 1.
do if (zmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,zcol).
end loop.
do if (wzmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).

```



```

compute vmat(8,i) =
modvals(i,zcol)&*modvals(i,wcol).
end loop.
end if.
end if.
do if (zmy = 1).
compute mints3 = make(n,(nmeds*mp),0).
do if (zy = 0).
compute datayed = {datayed, z}.
compute datanmy = {datanmy; zname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wzmy = 1 and wzy = 0).
compute datayed = {datayed,w&*z}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints3(:,((i*mp)+1))= m(:,(i+1))&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wzmy = 1).
compute mints3(:,((i*mp)+2))=
m(:,(i+1))&*w&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', zname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints3}.
end if.
compute decoc = 1.
compute modmat = make(5,5,999).

```

```

compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
do if (wy = 1 and model > 3).
compute datayed = {datayed, w, x&*w}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm2(1,1)=wname.
compute modmatp(1,1) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ',', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (zy = 1).
compute datayed = {datayed,z,x&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm2(1,2)=zname.
compute modmatp(1,2) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ',', ' '}.
compute datanmy = {datanmy; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (wzy = 1).
compute datayed = {datayed,w&*z,x&*w&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ',', ' '}.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.

```

```

compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vxy = 1).
compute datayed = {datayed, x&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm2(1,3)=vname.
compute modmatp(1,3) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (qxy = 1).
compute datayed = {datayed, x&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm2(1,4)=qname.
compute modmatp(1,4) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqxy = 1).
compute datayed = {datayed, x&*v&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, 'X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.

```

```

compute intcnt = intcnt+1.
end if.
end if.
end if.
do if (wvxy = 1).
compute datayed = {datayed, x*w&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, 'X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute modvalsd = 0.
compute ttt = rsum(modmatp).
do if (rsum(modmatp) > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm2(1,tmp) = modvnm2(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:ttt).
compute modvnm2=modvnm2(:,1:ttt).
compute modmatv=modmatv(:,1:ttt).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvalsd =
make((modmatv(1,1)*modprod(1,1)), ttt,0).
loop i = 1 to ttt.
compute strt = 1.

```

```

compute fnsh=0.
loop if (fnsh < nrow(modvalsd)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvalsd(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
end if.
do if (ttt > 0).
loop i = 1 to ncol(modvalsd).
do if (modvnm2(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm2(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm2(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm2(1,i)=qname).
compute qcol = i.
end if.
end loop.
compute directv = make(nrow(modvalsd),1,1).
do if (wy = 1).
compute directv ={directv, modvalsd(:,wcol)}.
end if.
do if (zy = 1).
compute directv ={directv, modvalsd(:,zcol)}.
end if.
do if (wzy = 1).
compute directv ={directv,
(modvalsd(:,wcol)&*modvalsd(:,zcol))}.
end if.
do if (vxy = 1).
compute directv ={directv, modvalsd(:,vcol)}.

```

```

end if.
do if (qxy = 1).
compute directv={directv, modvalsd(:,qcol)}.
end if.
do if (vqxy = 1).
compute directv={directv,
(modvalsd(:,vcol)&*modvalsd(:,qcol))}.
end if.
do if (wvxy = 1).
compute directv={directv,
(modvalsd(:,vcol)&*modvalsd(:,wcol))}.
end if.
end if.
compute ydatacol=ncol(datayed).
do if (ncovs > 0).
do if (covmy <> 2).
compute datamed = {datamed,c}.
end if.
do if (covmy <> 1).
compute datayed = {datayed,c}.
end if.
do if (covmy = 1 and model = 4 and mcx > 0).
compute datayed = {datayed,c}.
end if.
compute covmeans = csum(c)/n.
end if.
do if (cluster > 0).
compute datamed = {datamed,cld}.
compute datayed = {datayed, cld}.
compute cldmeans = csum(cld)/n.
end if.
compute mst = 3.
compute mnd = mst+nmeds-1.
compute ydatacol=ncol(datayed).
compute mdatacol=ncol(datamed).
do if (ncovs > 0).
compute datanmy = {datanmy; t(cnames)}.
do if (model > 3).
compute datanmm = {datanmm; t(cnames)}.
end if.

```

```

end if.
compute datanmy = {'constant';
datanmy(3:nrow(datanmy),1)}.
do if (model > 3).
compute datanmm = {'constant';
datanmm(3:nrow(datanmm),1)}.
end if.
compute amm = make(2,1,0).
compute abmm = make(2,1,0).
compute mnv = csum(datayed(:,2)/n).
compute mnv = make(n,1,mnv).
compute ssty = csum((datayed(:,2)-mnv)**2).
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
do if (ws=1).
compute stddevm=diag(sigma).
compute
stddevm=sqrt(stddevm(3:(nmeds+2),1)).
end if.
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute r2xy =
(sigma(2,(3+nmeds))/(stddevy*stddevx))**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))**2.
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).
do if (model = 4 and nmeds = 1 and cluster = 0
and ncovs = 0).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).

```

```

compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
compute datatm = datamed.
compute dataty = datayed.
compute mdlrms2 = { '4' ; yname; xname}.
compute mdlrms = {'Model ='; ' Y ='; ' X ='
}.
do if (ws = 1).
compute mdlrms2 = { '4' ; yname}.
compute mdlrms = {'Model ='; ' Y ='}.
end if.
loop i = 1 to ncol(mnames).
compute mdlrms2 = {mdlrms2; mnames(1,i)}.
do if (i = 1 and ncol(mnames) = 1).
compute mdlrms = {mdlrms; ' M ='}.
else.
compute mdlrms = {mdlrms; mlab(i,1)}.
end if.
end loop.
do if (wname <> 'xxx').
compute mdlrms2 = {mdlrms2; wname}.
compute mdlrms = {mdlrms; ' W ='}.
end if.
do if (zname <> 'xxx').
compute mdlrms2 = {mdlrms2; zname}.
compute mdlrms = {mdlrms; ' Z ='}.

```



```

end if.
do if (vname <> 'xxx').
compute mdlms2 = {mdlms2; vname}.
compute mdlms = {mdlms; '  V = '}.
end if.
do if (qname <> 'xxx').
compute mdlms2 = {mdlms2; qname}.
compute mdlms = {mdlms; '  Q = '}.
end if.
do if (jn = 1 and model = 1 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
do if (jn = 1 and model = 3 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
compute mdpbe=0.
compute medte2=0.
do if ( 4 =4 and mcx > 0).
compute medte2=1.
end if.
do if (((mcx > 0) or (mcm > 0)) AND model =
1).
compute mdpbe=1.
compute dd={datayed(:,4),datayed(:,3)}.
do if (ncovs > 0).
compute dd={datayed(:,6:(5+ncovs)),dd}.
end if.
compute dd={datayed(:,2),dd}.
end if.
compute yes = ((nrow(modvals)=1) and
(contrast = 1) and (model = 3 or model = 2)).
compute yes2 = ((nrow(modvals)=1) and
(contrast = 1) and (model = 2)).
print/title = '***** PROCESS Procedure
for SPSS Release 2.16.3 *****'.
print/title = '      Written by Andrew F. Hayes,
Ph.D.      www.afhayes.com'.

```

```

print/title = ' Documentation available in
Hayes (2013).
www.guilford.com/p/hayes3'/space=0.
print mdlrms2/title =
*****
*****'/rnames = mdlrms/format =
a8.
do if (ncovs > 0).
print cnames/title = 'Statistical Controls: '/rlabels
= 'CONTROL='/format a8.
end if.
print n/title = 'Sample size'/format F10.0.
do if (cluster > 0).
print cluster/rnames = cvname/title =
'Clustering Variable and Number of Clusters'.
end if.
do if (model > 3 and ( 'random' <> 'random')).
compute seedt= 'random'.
print seedt/title='Custom'/format=A12/rlabel =
'Seed:'.
end if.
do if (mdpbe=0 and medte2 = 0).
loop bt = 1 to (boot+1).
do if (bt = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef = 0.
do if (bt > 1).
loop.
compute v=trunc(uniform(n,1)*n)+1.
compute datayed = dataty(v,:).
do if (ws=1).
compute
bmeans=csum(datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed)))/n.
compute cntmat=make(n,nmeds,1).
compute bmeans2=bmeans.
do if (nmeds > 1).
compute bmeans2=mdiag(bmeans).

```

```

end if.
compute cntmat=cntmat*bmeans2.
compute datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed))=datayed(:,(ncol(data
yed)-nmeds+1):ncol(datayed))-cntmat.
end if.
compute detcheck=det(t(datayed)*datayed).
compute rk=(detcheck <> 0).
do if (model > 3).
compute datamed = datatm(v,:).
compute detcheck=det(t(datamed)*datamed).
do if (rk=1).
compute rk=(detcheck <> 0).
end if.
end if.
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
compute temp=diag(sigma).
compute bad = bad+(1-rk).
compute false = 1.
end loop if (rk = 1).
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nme
ds)).
do if (model = 4 and nmeds = 1 and ncovs = 0
and cluster = 0).
compute r2xy =
(sigma(2,4)/(stddevy*stddevx))&**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))&**2.
compute sstot = sigma(2,2)*(n-1).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).

```

```

compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
end if.
do if (model > 3).
loop im = 1 to nmeds.
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant';
datanmm((2+nmeds):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
do if (model = 6).
do if (im = 1).
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
end if.
do if (im > 1).

```

```

compute xm={cons, datamed(:,3:(im+1)),
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant'; datanmm(2:im,1);
datanmm((mnd):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
compute
mmpaths((im+1),(2:im))=t(coeff(2:im,1)).
end if.
end if.
do if (ws = 1).
compute
coeff(1,1)=csum(datamed(:,(2+im)))/n.
end if.
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
do if (bt = 1).
compute resid=datamed(:,(2+im))-xm*coeff.
compute sse =cssq(resid).
compute mse = sse/(n-ncol(xm)).
compute mnv = csum(data(:,(2+im)))/n.
compute mnv = make(n,1,mnv).
compute sstm = csum((data(:,(2+im))-
mnv)&**2).
compute k3 = nrow(coeff).
do if (hc3 = 1).
compute h = xm(:,1).
loop i3=1 to n.
compute h(i3,1)= xm(i3,:)*invXtX*t(xm(i3,:)).
end loop.
loop i3=1 to k3.
compute xm(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xm(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xm(:,i3) = sqrt(mse)&*xm(:,i3).

```

```

end loop.
end if.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute hccov=invXtX*t(xm)*xm*invXtX.
compute mcmats(im,im)=hccov(2,2).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*hccov*lmat)*((t(lmat)
)*coeff))/dfnum.
compute coeff = coeff(1:(nrow(coeff)-
clsdmy),1).
compute mcoeff(im,1)=coeff(2,1).
compute standerr =
sqrt(diag(invXtX*t(xm)*xm*invXtX)).
compute standerr = standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1).
compute standerr(1,1)=stddevm(im,1)/sqrt(n).
end if.
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-
ncol(xm)))).
compute temp=(n-ncol(xm)).
do if (ws=1).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute temp=(n-1).
end if.
compute xd = abs(xp2).
compute temp = (temp* (exp((temp-
(5/6))*((xd/(temp-(2/3)+(.11/temp)))*(xd/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = coeff-
sqrt(abs(temp))*standerr.
compute temp2 =
coeff+sqrt(abs(temp))*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (ws=1).

```

```

compute op=op(1,:).
compute xnmn =
apathnam(((nmeds>1)+1+(im-1)),1).
end if.
compute sobel(im,1) = coeff(2,1).
compute sobel(im,2)=standerr(2,1).
compute temp = mnames(1,im).
compute r2full = 1-(sse/sstm).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute summ = {sqrt(r2full), r2full, mse,
Fratio, dfnum, dfden, pfr}.
do if (detail = 1).
print temp/title =
*****
*****'/rlabels = 'Outcome:'/format
= A8.
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels 'R',
'R-sq', 'MSE','F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
print op/title = 'Model'/rnames = xnmn/clabels
'coeff', 'se', 't', 'p', 'LLCI', 'ULCI'/format = F10.4.
do if (covcoeff=1 and ws <> 1).
compute
hccovtmp=hccov(1:nrow(op),1:nrow(op)).
compute cnamestp=t(xnmn).
print hccovtmp/title='Covariance matrix of
regression parameter
estimates'/rnames=xnmn/cnames=cnamestp/f
ormat= F10.4.
end if.
do if (nmods > 0 and nrow(mintkey) > 1).
print mintkey/title = 'Product terms key:'/format
= A8.
end if.
do if (ws = 1).
compute wsdf=n-1.

```

```

print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
end if.
end if.
compute ymat(1,im) = coeff(2,1).
do if (ws = 1).
compute ymat(1,im)=coeff(1,1).
end if.
do if (wm = 1).
compute ymat(2,im) = coeff(4,1).
do if (zm = 1).
compute ymat(3,im) = coeff(6,1).
do if (wzm = 1).
compute ymat(4, im) = coeff(8,1).
end if.
end if.
end if.
do if (model = 6).
compute mmpaths((im+1),1)=coeff((im+1),1).
end if.
end loop.
end if.
loop totlp = 1 to (1+(toteff*(bt = 1))).
do if (toteff = 1 and totlp =2).
compute xy =
{cons,datayed(:,(3+nmeds):ydatacol)}.
else.
compute xy={cons, datayed(:,3:ydatacol)}.
end if.
do if (dichy = 1).
compute pt2 =
make(nrow(datayed(:,2)),1,(csum(datayed(:,2))
/n)).
compute LL3 = datayed(:,2)*ln(pt2)+(1-
datayed(:,2))*ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xy),1,0).
compute LL1 = 0.
compute xy22=xy.

```



```

loop jjj = 1 to iterate.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.
compute coeff =
bt1+inv(t(xy22)*xy)*t(xy)*(datayed(:,2)-pt1).
compute pt1 = 1/(1+exp(-(xy*coeff))).
compute itprob = csum((pt1 <
.00000000000001) or (pt1 >
.99999999999999)).
do if (itprob = 0).
compute LL = datayed(:,2)*ln(pt1)+(1-
datayed(:,2))*ln(1-pt1).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
break.
end if.
compute bt1 = coeff.
compute LL1 = LL2.
end loop.
do if (jjj >= iterate and iterr = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
compute iterr = 1.
end if.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.
compute covmat = inv(t(xy22)*xy).
release xy22.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
end if.
do if (dichy = 0).
compute invXtX = inv(t(xy)*xy).
compute coeff =invXtX*t(xy)*datayed(:,2).

```

```

do if (effsize=1 and model > 3 and model < 7
and (ncovs > 0 or clsdmy > 0)).
compute xysdy =
{cons,datayed(:,(4+nmeds):ydatacol)}.
compute coeffsd
=inv(t(xysdy)*xysdy)*t(xysdy)*datayed(:,2).
compute resid=datayed(:,2)-(xysdy*coeffsd).
compute sdy cov =sqrt(csum(resid*&resid)/(n-
ncol(xysdy))).
compute xvaron=datayed(:,(3+nmeds)).
compute
coeffx2=inv(t(xysdy)*xysdy)*t(xysdy)*xvaron.
compute residx2=xvaron-(xysdy*coeffx2).
compute sdx cov
=sqrt(csum(residx2*&residx2)/(n-ncol(xysdy))).
end if.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4 and bt > 1).
compute resid=datayed(:,2)-xy*coeff.
compute sse = cssq(resid).
compute r2full = 1-(sse/sstot).
end if.
do if (bt =1).
compute resid=data(:,2)-xy*coeff.
compute k3 = nrow(coeff).
compute sse = cssq(resid).
compute mse = sse/(n-ncol(xy)).
compute
hcnot=coeff&/(sqrt(diag(mse*invXtX))).
do if (hc3 = 1).
compute h = xy(:,1).
loop i3=1 to n.
compute h(i3,1)= xy(i3,:)*invXtX*t(xy(i3,:)).
end loop.
loop i3=1 to k3.

```

```

compute xy(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xy(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xy(:,i3) = sqrt(mse)&*xy(:,i3).
end loop.
end if.
compute covmat = (invXtX*t(xy)*xy*invXtX).
end if.
end if.
do if (bt = 1).
do if (model = 2).
compute xy2={cons, datayed(:,3:ydatacol)}.
compute temp = ncol(xy2).
do if (temp > 6).
compute xy3=xy2(:,7:temp)}.
end if.
compute xy2={xy2(:,1:3), xy2(:,5)}.
do if (temp > 6).
compute xy2={xy2, xy3}.
release xy3.
end if.
compute invXtX = inv(t(xy2)*xy2).
compute coeff2=invXtX*t(xy2)*datayed(:,2).
compute ssem2=cssq(datayed(:,2)-xy2*coeff2).
release xy2.
end if.
compute standerr = sqrt(diag(covmat)).
do if (totlp =1).
compute
mcmats((nmeds+1):ncol(mcmats),(nmeds+1):n
col(mcmats))=covmat(2:(1+nmeds),2:(1+nmed
s)).
end if.
compute standerr=standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1 and totp=2).
compute standerr(1,1)=stddevy/sqrt(n).

```

```

end if.
compute coeffplt = coeff.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*covmat*lmat)*((t(lmat)*coeff))/dfnum).
compute coeff=coeff(1:(nrow(coeff)-clsdmy),1).
do if (totlp=1).
compute
mccoeff((nmeds+1):nrow(mccoeff))=coeff(2:(1+nmeds),1).
end if.
compute bbbb=coeff(2,1).
do if totp = 1.
compute deco(1,1)=2+nmeds.
compute deco = deco(1:decoc,1).
compute covdirt = make((nrow(covmat)-clsdmy),(ncol(covmat)-clsdmy),0).
compute covdirt = covmat(deco,:).
compute covdir =
make(nrow(covdirt),ncol(covdirt),0).
compute covdir = covdirt(:,t(deco)).
compute deco=coeff(deco,1).
do if (ttt > 0).
compute sedir =
sqrt(diag(directv*covdir*t(directv))).
compute directv = directv*deco.
end if.
compute sobel(:,3)=coeff(2:(1+nmeds),1).
compute sobel(:,4) = standerr(2:(1+nmeds),1).
compute sobel2 = sobel*sobel.
do if (varorder <> 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)*sobel2(:,4)+sobel2(:,3)*sobel2(:,2)).
end if.
do if (varorder = 2).

```

```

compute sobel(:,2) =
sqrt(sobel2(:,1)&*sobel2(:,4)+sobel2(:,3)&*sob
el2(:,2)+sobel2(:,2)&*sobel2(:,4)).
end if.
compute sobel(:,1) = sobel(:,1)&*sobel(:,3).
compute sobel(:,3) = sobel(:,1)&/sobel(:,2).
compute sobel(:,4) = 2*(1-
cdfnorm(abs(sobel(:,3)))).
end if.
do if (dichy = 0).
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
do if (ws= 1 and totlp=2).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute dfden=n-1.
end if.
compute cnms = {'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute op = {coeff, standerr, tratio, p}.
end if.
do if (dichy = 1).
compute tratio = (coeff&/standerr).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute wald = tratio&*tratio.
compute cnms = {'coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute temp=coeff-abs(xp2)*standerr.
compute op = {coeff, standerr, tratio, p, temp}.
compute temp=coeff+abs(xp2)*standerr.
compute op = {op, temp}.
end if.
do if (detail = 1).
do if (totlp = 2).
print yname/title = '*****
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:'/'format = A8.
end if.
do if (totlp <> 2).

```

```

print yname/title =
*****
*****'/rlabels = 'Outcome:.'/format
= A8.
end if.
end if.
do if (dichy = 1 and bt = 1 and totlp = 1).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:.'/cnames = nmsd/format = F9.2.
end if.
do if (dichy = 0).
compute r2full = 1-(sse/ssty).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute jndf=dfden.
compute wsdf=dfden.
compute xd = abs(xp2).
compute jncrit = (dfden* (exp((dfden-
(5/6))*(xd/(dfden-
(2/3)+(.11/dfden)))*(xd/(dfden-
(2/3)+(.11/dfden))))-1)).
compute summ = {sqrt(r2full), r2full, mse,
fratio, dfnum, dfden, pfr}.
compute temp1=coeff-sqrt(jncrit)*standerr.
compute temp2=coeff+sqrt(jncrit)*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (detail = 1).
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels =
'R', 'R-sq', 'MSE' 'F', 'df1', 'df2', 'p'/format =
F10.4.
end if.
end if.
end if.
do if (dichy = 1).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(coeff)-
1)).
compute mcF = LLdiff/LL3.

```

```

compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
do if (detail = 1).
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format =
F10.4.
end if.
end if.
do if (totlp = 2).
compute datanmy={constant';
datanmy((nmeds+2):nrow(datanmy),1)}.
end if.
do if (detail = 1).
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
do if (ws=1).
compute op2=op(1:(1+nmeds),:).
compute datanmy2 = {'c" path'; bpathnam(
(1+(nmeds>1)): ((1+(nmeds>1))+(nmeds-
1)),1)}.
do if (totlp=2).
compute datanmy2={'c path'}.
compute op2=op(1,:).
compute wsdf=n-1.
end if.
print op2/title = 'Model'/rnames =
datanmy2/cnames = cnms/format = F10.4.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
do if (ws <> 1).
print op/title = 'Model'/rnames =
datanmy/cnames = cnms/format = F10.4.
do if (covcoeff=1).
compute
covmattp=covmat(1:nrow(op),1:nrow(op)).
compute cnamestp=t(datanmy).

```

```

print covmattp/title='Covariance matrix of
regression parameter
estimates'/rnames=datanmy/cnames=cnamest
p/format= F10.4.
end if.
end if.
end if.
do if (ttt = 0 and totlp = 1).
compute deco = op((nmeds+2),:).
do if (ws = 1).
compute deco=op(1,:).
end if.
end if.
do if (ttt = 0 and totlp = 2).
compute decotot = op(2,:).
do if (ws = 1).
compute decotot=op(1,:).
end if.
end if.
do if (nmods > 0 and model > 4 and detail = 1
and nrow(yintkey) > 1)).
print yintkey/title = 'Product terms key:'/format
= A8.
end if.
do if (nmods > 0 and model < 4 and detail = 1).
print yintkey2/title = 'Product terms key:'/format
= A8.
do if ((model = 1 or model = 2) and dichy = 0).
compute temp={((op(4,3)**2)*(1-r2full))/dfden,
op(4,3)**2, 1, dfden, op(4,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(4,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(2,1).
do if (model = 2).
compute temp={temp;((op(6,3)**2)*(1-
r2full))/dfden, op(6,3)**2, 1, dfden, op(6,4)}.
do if (hc3=1).

```



```

compute temp(2,1)=((hcnot(6,1)**2)*(1-
r2full))/dfden.
end if.
compute frat2=(dfden*(r2full-(1-
(ssem2/ssty))))/(2*(1-r2full)).
do if (hc3=1).
compute lmat=make(nrow(coeffplt),2,0).
compute lmat(4,1)=1.
compute lmat(6,2)=1.
compute frat2 =
(t(t(lmat)*coeffplt)*inv(t(lmat)*covmat*lmat)*((t(l
mat)*coeffplt))/2).
end if.
compute temp={temp;(r2full-(1-
(ssem2/ssty))),frat2,2,dfden,1-
fcd(frat2,2,dfden)}.
compute rnms={rnms; yintkey2(3,1);'Both'}.
end if.
print temp/title = 'R-square increase due to
interaction(s):'/rnames=rnms/clabels = 'R2-
chng', 'F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (model = 3 and dichy = 0).
compute temp={((op(8,3)**2)*(1-r2full))/dfden,
op(8,3)**2, dfden, op(8,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(8,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(5,1).
print temp/title = 'R-square increase due to
three-way interaction:.'/rnames=rnms/clabels =
'R2-chng', 'F(1,df2)', 'df2', 'p'/format = F10.4.
end if.
end if.
end if.
do if (model = 6 and totlp = 1).
compute
mmpaths(nrow(mmpaths),1)=coeff(nrow(mmpa
ths),1).

```

```

compute
mmpaths(nrow(mmpaths),(2:(nmeds+1)))=t(co
eff(2:(nmeds+1),1)).
end if.
do if (totlp = 1).
loop im = 1 to nmeds.
do if (model < 4).
compute ymat(1,im) = coeff(3,1)*(1-yes).
compute ymat(2,im) = coeff(4,1).
compute cmat(1,im) = covmat(3,3).
compute cmat(2,im) = covmat(4,4).
compute cmat(5,im) = covmat(3,4).
compute jnb1=coeff(3,1).
compute jnb3=coeff(4,1).
compute jnsb1=covmat(3,3).
compute jnsb3=covmat(4,4).
compute jnsb1b3=covmat(3,4).
do if (model = 2 or model = 3).
compute ymat(3,im) = coeff(6,1).
compute cmat(3,im) = covmat(6,6).
compute cmat(6,im) = covmat(3,6).
compute cmat(8,im) = covmat(4,6).
end if.
do if (model = 3).
compute ymat(4,im) = coeff(8,1).
compute cmat(4,im) = covmat(8,8).
compute cmat(7,im) = covmat(3,8).
compute cmat(9,im) = covmat(4,8).
compute cmat(10,im) = covmat(6,8).
compute jnb1=coeff(4,1).
compute jnb3=coeff(8,1).
compute jnsb1=covmat(4,4).
compute jnsb3=covmat(8,8).
compute jnsb1b3=covmat(4,8).
end if.
end if.
do if (model > 3).
compute ymat(5,im) = coeff((1+im),1).
end if.
do if (xmy = 1).

```

```

compute ymat(6,im) = coeff((2+nmeds+im),1).
end if.
do if (vy = 1).
compute ymat(6,im) = coeff((3+nmeds+im),1).
end if.
do if (qy = 1 and vy = 1).
compute ymat(6,im) = coeff((5+nmeds+((im-1)*2)),1).
compute ymat(7,im) = coeff((6+nmeds+((im-1)*2)),1).
end if.
do if (vqy = 1).
compute ymat(6,im) = coeff((6+nmeds+((im-1)*3)),1).
compute ymat(7,im) = coeff((7+nmeds+((im-1)*3)),1).
compute ymat(8,im) = coeff((8+nmeds+((im-1)*3)),1).
end if.
do if (wmy = 1).
compute ymat(7,im) = coeff((3+nmeds+im-wy),1).
end if.
do if (wmy = 1 and vy = 1).
compute ymat(7,im) =coeff((4+(nmeds*2)+im-wy),1).
end if.
do if (wmy = 1 and vy = 1 and wvmy = 1).
compute ymat(7,im) =
coeff((6+(nmeds*2)+((im-1)*2)-wy),1).
compute ymat(8,im) =
coeff((7+(nmeds*2)+((im-1)*2)-wy),1).
end if.
do if (wmy = 1 and zmy = 1).
compute ymat(6,im) = coeff((6-(wzy*3)+(wzm-1)+(nmeds*2)+((im-1)*2)*wzm)+((im-1)*(1-wzm))-((zy-wzy)*2),1).
compute ymat(7,im) = coeff((3-zy+im+nmeds),1).
do if (wzmy = 1).

```

```

compute ymat(8,im) = coeff((7-
(wzy*3)+(nmeds*2)+((im-1)*2)),1).
end if.
end if.
do if (nmods > 0 and model <> 5).
loop indlp = 1 to nrow(modvals).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp)).
do if (model > 6).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp))*csum(ym
at(5:8,im)*vmat(5:8,indlp)).
end if.
end loop.
compute indboot((bt+(im-
1)*(boot+1)),:)=t(indeff).
do if (imm =1 or imm = 3).
compute
indbootp(bt,im)=ymat(2,im)*ymat(5,im).
do if (imm=3).
compute
indbootp(bt,(im+nmeds))=ymat(3,im)*ymat(5,i
m).
do if (zdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxz-cminz).
end if.
end if.
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
end if.
end if.
do if ((imm= 2 or imm = 4) or model = 74).
compute
indbootp(bt,im)=ymat(1,im)*ymat(6,im).
do if (imm=4).

```

```

compute
indbootp(bt,(im+nmeds))=ymat(1,im)*ymat(7,i
m).
do if (qdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxq-cminq).
end if.
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
end if.
end if.
do if (imm=7).
compute
indbootp(bt,im)=ymat(2,im)*ymat(6,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(2,im)*y
mat(6,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(6,im)*onesmv1)+(ymat(2,im)*y
mat(6,im)*t(immmv1)).
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).

```

```

compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1))))*(c
maxv-cminv).
end if.
end if.
do if (imm=5).
compute
indbootp(bt,im)=ymat(4,im)*ymat(5,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(4,im)*y
mat(5,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(3,im)*ymat(5,im)*onesmv1)+(ymat(4,im)*y
mat(5,im)*t(immmv1)).
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2))))*(c
maxw-cminw).
end if.
do if (zdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxz-
cminz).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1))))*(c
maxz-cminz).
end if.
end if.
do if (imm=6).

```

```

compute
indbootp(bt,im)=ymat(1,im)*ymat(8,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(1,im)*ymat(6,im)*onesmv2)+(ymat(1,im)*y
mat(8,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(7,im)*onesmv1)+(ymat(1,im)*y
mat(8,im)*t(immmv1)).
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxv-cminv).
end if.
do if (qdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxq-
cminq).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxq-cminq).
end if.
end if.
do if ((model = 58 or model = 59) and
wvdich=1).
compute
indbootp(bt,im)=(ymat(1,im)*ymat(7,im)*(cmax
w-cminw))+(ymat(2,im)*ymat(5,im)*(cmaxw-
cminw))+(ymat(2,im)*ymat(7,im)*((cmaxw*cma
xw)-(cminw*cminw))).
end if.
end if.

```

```

do if (model = 4 or model = 5).
compute
indboot(bt,im)=csum(yamat(1:4,im)*vmat(1:4,1
))*csum(yamat(5:8,im)*vmat(5:8,1)).
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.00000000000001.
end if.
compute sumind=rsum(indboot(bt,:)).
do if (im=nmeds).
compute
pmeff(bt,2:(im+1))=indboot(bt,:)/(sumind+coeff(
(2+nmeds),1)).
end if.
compute
rmeff(bt,(im+1))=indboot(bt,im)/coeff((2+nmeds
),1).
compute
abpseff(bt,(im+1))=indboot(bt,im)/stddevy.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*stddevx.
do if (ncovs > 0 or clsdmy > 0).
compute
abpseff(bt,(im+1))=indboot(bt,im)/sdy cov.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*sdx cov.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,1) = r2my-(r2full-r2xy).
compute abmmr = 1.
do if (indboot(bt,im) < 0).
compute abmmr = cmin(abmm).
end if.
do if (indboot(bt,im) > 0).
compute abmmr = cmax(abmm).
end if.
compute kappa2(bt,1)=indboot(bt,im)/abmmr.
compute tmp = indboot(bt,im)/abmmr.
end if.

```



```

end if.
end if.
do if (model = 6).
do if (nmeds = 2).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(4,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3).
compute
indboot(bt,3)=mmpaths(3,1)*mmpaths(4,3).
end if.
do if (nmeds = 3).
compute indboot(bt,1) =
mmpaths(2,1)*mmpaths(5,2).
compute indboot(bt,2) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3).
compute indboot(bt,3) =
mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4).
compute indboot(bt,4) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4).
compute indboot(bt,5) =
mmpaths(3,1)*mmpaths(5,3).
compute indboot(bt,6) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4).
compute indboot(bt,7) =
mmpaths(4,1)*mmpaths(5,4).
end if.
do if (nmeds = 4).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(6,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(6,3).
compute
indboot(bt,3)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(6,4).

```

```

compute
indboot(bt,4)=mmpaths(2,1)*mmpaths(5,2)*mmpaths(6,5).
compute
indboot(bt,5)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(6,4).
compute
indboot(bt,6)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3)*mmpaths(6,5).
compute
indboot(bt,7)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,8)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,9)=mmpaths(3,1)*mmpaths(6,3).
compute
indboot(bt,10)=mmpaths(3,1)*mmpaths(4,3)*mmpaths(6,4).
compute indboot(bt,11)
=mmpaths(3,1)*mmpaths(5,3)*mmpaths(6,5).
compute indboot(bt,12) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,13) =
mmpaths(4,1)*mmpaths(6,4).
compute indboot(bt,14) =
mmpaths(4,1)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,15) =
mmpaths(5,1)*mmpaths(6,5).
end if.
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.00000000000001.
end if.
compute
pmeff(bt,:)=indboot(bt,:)/(rsum(indboot(bt,:))+mmpaths(nrow(mmpaths),1)).

```

```

compute rmeff(bt,:) =
indboot(bt,:)/mmpaths(nrow(mmpaths),1).
compute abpseff(bt,:)=indboot(bt,:)/stddevy.
compute
abcseff(bt,:)=(stddevx*indboot(bt,:))/stddevy.
do if (ncovs > 0 or clsdmy > 0).
compute abpseff(bt,:)=indboot(bt,:)/sdy cov.
compute
abcseff(bt,:)=(sdxcov*indboot(bt,:))/sdy cov.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,:) = r2my-(r2full-r2xy).
end if.
end if.
end if.
end loop.
end if.
end loop.
do if (savboot = 1 and bt > 1).
compute bootstrp((bt-1),:)=bootcoef.
end if.
end loop.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
release datayed, dat, datamed, data, y, x,
datatm, dataty, mints2, cons, mnv, tmp.
do if (mc > 0).
compute x1 = sqrt(-
2*ln(uniform(mc,nrow(mcmats))))&*cos((2*3.14
159265358979)*uniform(mc,nrow(mcmats))).
compute x1=x1*chol(mcmats).
loop i=1 to nrow(x1).
compute x1(i,:)=x1(i,:)+t(mccoeff).
end loop.
loop i = 1 to nmeds.

```

```

compute
indboot(2:nrow(indboot),i)=x1(:,i)*x1(:,(i+nme
ds)).
end loop.
end if.
do if (ttt = 0 and model > 3).
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
end if.
do if (model <> 74).
do if (dichy = 0).
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 't', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 't', 'p', 'LLCI', 'ULCI'/format =
F10.4.
else.
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 'Z', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 'Z', 'p', 'LLCI', 'ULCI' /format =
F10.4.
end if.
end if.
end if.
do if (ttt > 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
compute clbs = {modvnm2, 'Effect', 'SE', 't', 'p',
'LLCI', 'ULCI'}.

```

```

compute tratio = (directv&/sedir).
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
compute outp = {modvalsd, directv, sedir,
tratio, p}.
do if (dichy = 0).
compute temp1=directv-sqrt(jncrit)*sedir.
compute temp2=directv+sqrt(jncrit)*sedir.
compute outp = {outp, temp1, temp2}.
end if.
do if (dichy = 1).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute temp = directv-abs(xp2)*sedir.
compute outp = {outp, temp}.
compute temp = directv+abs(xp2)*sedir.
compute outp = {outp, temp}.
compute clbs = {modvnm2, 'Effect', 'SE', 'Z', 'p',
'LLCI', 'ULCI'}.
end if.
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
print outp/title = 'Conditional direct effect(s) of
X on Y at values of the moderator(s):'/format =
F10.4 /cnames = clbs.
end if.
do if (nmods > 0 and model <> 5).
do if (model < 4).
print/title =
'*****'
'*****'.
compute zmat(1,1) = 1*(1-yes).
compute cfse = make(nrow(modvals),1,0).
loop #m = 1 to nrow(modvals).
do if (model = 1).
compute zmat(2,1)=modvals(#m,1)**2.
compute zmat(5,1)=2*modvals(#m,1).
end if.
do if (model = 2 or model = 3).
compute zmat(2,1)=modvals(#m,2)**2.
compute zmat(3,1)=modvals(#m,1)**2.

```

```

compute
zmat(4,1)=(modvals(#m,1)**2)*(modvals(#m,2)
**2)*(1-yes2).
compute zmat(5,1)=2*modvals(#m,2)*(1-yes).
compute zmat(6,1)=2*modvals(#m,1)*(1-yes).
compute
zmat(7,1)=2*modvals(#m,1)*modvals(#m,2)*(1-
yes).
compute
zmat(8,1)=2*modvals(#m,1)*modvals(#m,2).
compute
zmat(9,1)=2*modvals(#m,1)*(modvals(#m,2)**
2)*(1-yes2).
compute
zmat(10,1)=2*(modvals(#m,1)**2)*modvals(#m
,2)*(1-yes2).
end if.
compute cfse(#m,1)=sqrt(csum(zmat&*cmat)).
end loop.
end if.
do if (nmods > 0).
compute clbs = {modvnm, 'Effect'}.
loop im = 1 to nmeds.
compute obs = t(indboot(1+(im-1)*(boot+1),:)).
compute outp = {modvals, obs}.
do if (model < 4).
compute tstat = obs&/cfse.
do if (dichy = 0).
compute pval = 2*(1-tcdf(abs(tstat), (n-
ncol(xy)))).
compute temp=obs-sqrt(jncrit)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp=obs+sqrt(jncrit)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 't', 'p', 'LLCI', 'ULCI'}.
compute jnclbs=clbs.
end if.
do if (dichy = 1).
compute pval = 2*(1-cdfnorm(abs(tstat))).
compute temp = obs-abs(xp2)*cfse.

```

```

compute outp = {outp, cfse, tstat, pval, temp}.
compute temp = obs+abs(xp2)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute jnclbs=clbs.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indboot((1+(im-
1)*(boot+1)),:).
compute indboot2 = indboot((2+(im-
1)*(boot+1)):(1+(im-1)*(boot+1)+boot),:).
compute mnind = t(csum(indboot2)/boot).
compute stdind=t(sqrt((cssq(indboot2)-
((csum(indboot2)**2)/boot))/(boot-1))).
compute llci=make(1,ncol(indboot2),-999).
compute ulci=make(1,ncol(indboot2),-999).
loop #e = 1 to ncol(indboot2).

.
compute temp = indboot2(:,#e).
compute temp(GRADE( indboot2(:,#e) )) =
indboot2(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).

```

```

do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {modvals, obs, stdind, t(llci),
t(ulci)}.
compute clbs = {modvnm, 'Effect', 'Boot SE',
'BootLLCI', 'BootULCI'}.

```



```

end if.
compute mtemp = mnames(1, im).
compute rlbs =
make(nrow(modvals),1,mnames(1,im)).
do if (model < 4).
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
do if (yes=0).
print outp/title = 'Conditional effect of X on Y at
values of the moderator(s):'/cnames =
clbs/format = F10.4.
else.
compute outp=outp(:,3:ncol(outp)).
compute clbs=clbs(:,3:ncol(clbs)).
compute clbs(1,1)='Contrast'.
print outp/title = 'Contrast of conditional effects
of X on Y'/cnames = clbs/format = F10.4.
end if.
end if.
do if (model > 5 and mod74dic <> 1).
do if (im = 1).
print/title = 'Conditional indirect effect(s) of X on
Y at values of the moderator(s):'.
end if.
print outp/title = 'Mediator'/rnames =
rlbs/cnames = clbs/format = F10.4.
else if (mod74dic = 1 and model = 74).
do if (im = 1).
print/title = 'Indirect effect(s) of X on Y:'.
compute clbs3=clbs(1,2:ncol(clbs)).
end if.
compute outp3=outp(1,2:ncol(outp)).
print outp3/title = 'Mediator'/rnames =
rlbs/cnames = clbs3/format = F10.4.
end if.
end loop.
loop i = notes to 1 by -1.
do if (note(i,1) = 4 and yes=0).

```

```

print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 5 and yes=0).
print/title = 'Values for quantitative moderators
are the mean and plus/minus one SD from
mean.'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 14 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    below the mean was replaced
with the minimum because one SD below the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.
do if (note(i,1) = 15 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    above the mean was replaced
with the maximum because one SD above the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.
end loop.
do if (model = 3 and yes=0).
compute jnvals=make(nrow(matw),7,0).
compute jnvals(:,1)=matw.
compute jnvals(:,2)=jnb1+jnb3*jnvals(:,1).
compute
jnvals(:,3)=sqrt(jnsb1+2*jnvals(:,1)*jnsb1b3+(jn
vals(:,1)&*jnvals(:,1))*jnsb3).
compute jnvals(:,4)=jnvals(:,2)&/jnvals(:,3).

```

```

do if (dichy = 0).
compute jnvals(:,5)=2*(1-tcdf(abs(jnvals(:,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(:,5)=2*(1-
cdfnorm(abs(jnvals(:,4)))).
end if.
compute jnvals(:,6)=jnvals(:,2)-
sqrt(jncrit)*jnvals(:,3).
compute
jnvals(:,7)=jnvals(:,2)+sqrt(jncrit)*jnvals(:,3).
compute clbs={clbs(:,1),clbs(:,3:ncol(clbs))}.
do if (coeffci = 0).
compute jnvals =jnvals(:,1:(ncol(jnvals)-2)).
end if.
print jnvals/title = 'Conditional effect of X*M
interaction at values of W:'/cnames
=clbs/format = F10.4.
end if.
do if (jn = 1 and (model = 1 or model = 3) and
jndich = 0 and yes=0).
print/title = '***** JOHNSON-
NEYMAN TECHNIQUE
*****'.
compute ajn =(jncrit*jnsb3)-(jnb3*jnb3).
compute bjn = 2*((jncrit*jnsb1b3)-(jnb1*jnb3)).
compute cjn = (jncrit*jnsb1)-(jnb1*jnb1).
compute radarg = (bjn*bjn)-(4*ajn*cjn).
compute den = 2*ajn.
compute nrts = 0.
do if (radarg >= 0 and den <> 0).
compute x21 = (-bjn+sqrt(radarg))/den.
compute x22 = (-bjn-sqrt(radarg))/den.
compute roots = 0.
do if (x21 >= jnmin and x21 <= jnmax).
compute nrts = 1.
compute roots = {roots; x21}.
end if.
do if (x22 >= jnmin and x22 <= jnmax).

```

```

compute nrts = nrts + 1.
compute roots = {roots; x22}.
end if.
compute roots={roots,make(nrow(roots),2,0)}.
compute modtemp=m.
do if (model=3).
compute modtemp=w.
end if.
do if (nrts > 0).
compute roots = roots(2:nrow(roots),1:3).
compute roots(1,2)=(csum(modtemp <
roots(1,1))/n)*100.
compute roots(1,3)=(csum(modtemp >
roots(1,1))/n)*100.
do if (nrow(roots)=2).
compute roots(2,2)=(csum(modtemp <
roots(2,1))/n)*100.
compute roots(2,3)=(csum(modtemp >
roots(2,1))/n)*100.
end if.
print roots/title = 'Moderator value(s) defining
Johnson-Neyman significance
region(s)'/clabels = 'Value', '% below', '%
above'/format = F10.4.
compute jnvals=make((21+nrts),7,0).
loop i= 0 to 20.
compute jnvals((i+1),1)=jnmin+(i*((jnmax-
jnmin)/20)).
end loop.
loop i = 1 to nrts.
loop j = 2 to nrow(jnvals).
do if ((roots(i,1) > jnvals((j-1),1)) and (roots(i,1)
< jnvals(j,1))).
compute
jnvals((j+1):(21+i),1)=jnvals(j:(20+i),1).
compute jnvals(j,1)=roots(i,1).
end if.
end loop.
end loop.
loop i = 1 to nrow(jnvals).

```

```

compute jnvals(i,2)=jnb1+jnb3*jnvals(i,1).
compute
jnvals(i,3)=sqrt(jnsb1+2*jnvals(i,1)*jnsb1b3+(jn
vals(i,1)*jnvals(i,1))*jnsb3).
compute jnvals(i,4)=jnvals(i,2)/jnvals(i,3).
do if (dichy = 0).
compute jnvals(i,5)=2*(1-tcdf(abs(jnvals(i,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(i,5)=2*(1-
cdfnorm(abs(jnvals(i,4)))).
end if.
compute jnvals(i,6)=jnvals(i,2)-
sqrt(jncrit)*jnvals(i,3).
compute
jnvals(i,7)=jnvals(i,2)+sqrt(jncrit)*jnvals(i,3).
end loop.
do if (model = 1).
print jnvals/title = 'Conditional effect of X on Y
at values of the moderator (M)'/cnames
=jnclbs/format = F10.4.
end if.
do if (model = 3).
compute
jnclbs={jnclbs(:,1),jnclbs(:,3:ncol(jnclbs))}.
print jnvals/title = 'Conditional effect of X*M on
Y at values of the moderator (W)'/cnames
=jnclbs/format = F10.4.
end if.
end if.
do if (nrts = 0).
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
else.
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.

```

```

end if.
end if.
end if.
do if (model < 4 and (plot = 1)).
compute dataplot =
make((nrow(modvals)*nrow(matx)),(ncol(modv
als)+1),0).
compute tmp = 1.
loop i = 1 to nrow(modvals).
loop j = 1 to nrow(matx).
compute dataplot(tmp,:)= {matx(j,1),
modvals(i,:)}.
compute tmp=tmp+1.
end loop.
end loop.
compute dataplot = {dataplot,
make(nrow(dataplot),(1+dichy),0)}.
compute dataplo2 = make(nrow(dataplot),1,1).
do if (model = 1).
compute dataplo2 = {dataplo2, dataplot(:,2),
dataplot(:,1), (dataplot(:,1)*dataplot(:,2))}.
end if.
do if (model = 2 or model = 3).
compute dataplo2 = {dataplo2, dataplot(:,3),
dataplot(:,1), (dataplot(:,1)*dataplot(:,3)),
dataplot(:,2), (dataplot(:,1)*dataplot(:,2))}.
do if (model = 3).
compute dataplo2 = {dataplo2,
(dataplot(:,2)*dataplot(:,3)),
(dataplot(:,1)*dataplot(:,2)*dataplot(:,3))}.
end if.
end if.
loop i = 1 to nrow(dataplot).
compute tmp=dataplo2(i,:).
do if (ncovs > 0).
compute tmp = {tmp, covmeans}.
end if.
do if (cluster > 0).
compute tmp = {tmp, cldmeans}.
end if.

```

```

compute dataplot(i,(ncol(dataplot)-
(dichy)))=tmp*coeffplt.
do if (dichy = 1).
compute
dataplot(i,(ncol(dataplot)))=exp(tmp*coeffplt)/(1
+exp(tmp*coeffplt)).
end if.
end loop.
compute clbs = {xname, modvnm, 'yhat'}.
do if (dichy = 1).
compute clbs = {xname, modvnm, 'ln(odds)',
'prob'}.
end if.
print/title =
*****
*****!

```

```

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (model = 1).

```

```

do if (dichy = 0).
print/title = 'DATA LIST FREE/Comm_Pract
Eco_Innovation Operational_Performance.'
/space=1.
else if (dichy = 1).
print/title = 'DATA LIST FREE/Comm_Pract
Eco_Innovation logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
Operational_Performance BY Eco_Innovation.'
/space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Operational_Performance BY
Comm_Pract.' /space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
logodds BY Eco_Innovation.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
prob BY Eco_Innovation.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Comm_Pract.' /space=0.

```



```

print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Comm_Pract.' /space=0.
end if.
end if.
end if.
do if (model = 2 or model = 3).
do if (dichy = 0).
print/title = 'DATA LIST FREE/Comm_Pract
Eco_Innovation Operational_Performance.'
/space=1.
end if.
do if (dichy = 1).
print/title = 'DATA LIST FREE/Comm_Pract
Eco_Innovation logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
Operational_Performance BY
Eco_Innovation/PANEL ROWVAR=.' /space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Operational_Performance BY
Comm_Pract/PANEL ROWVAR=.' /space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
logodds BY Eco_Innovation/PANEL
ROWVAR=.' /space=0.

```

```

print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
prob BY Eco_Innovation/PANEL ROWVAR='
/space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Comm_Pract/PANEL
ROWVAR=' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Comm_Pract/PANEL
ROWVAR=' /space=0.
end if.
end if.
end if.
do if (ncovs > 0).
print/title = '* Estimates are based on setting
covariates to their sample means.'.
end if.
end if.
end if.
do if ((imm > 0) or (model = 74 and
mod74dic=0) or ((model = 58 or model = 59)
and wvdich=1))).
do if (imm > 4 and imm < 8).
compute
indbootp={indbootp(:,1:nmeds),indboop1,indbo
op2}.
end if.
compute obsprod = t(indbootp(1,:)).
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indbootp(1,:).
compute indbootp = indbootp(2:(boot+1),:).
compute mnindp = t(csum(indbootp)/boot).
compute stdindp=t(sqrt((cssq(indbootp)-
((csum(indbootp)**2)/boot))/(boot-1))).
compute llcip=make(1,ncol(indbootp),-999).

```

```

compute ulcip=make(1,ncol(indbootp),-999).
loop #e = 1 to ncol(indbootp).

.

compute temp = indbootp(:,#e).
compute temp(GRADE( indbootp(:,#e) )) =
indbootp(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).

```

```

end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcip(1,#e)=llcit.
compute ulcip(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {obsprod, stdindp, t(llcip),
t(ulcip)}.
compute clbs = {'Effect', 'SE(Boot)', 'BootLLCI',
'BootULCI'}.
do if (model = 8 or model = 12).
print/title='-----'.
compute outpimm=outp(1:nmeds,:).
print/title = 'Indirect effect of highest order
product:'/space=0.
print outpimm/title = 'Mediator'/cnames =
clbs/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).

```

```

print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'

end if.
compute clbs3 = {'Index', 'SE(Boot)',
'BootLLCI', 'BootULCI'}.
do if (imm < 5).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=outp((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'

compute outpimm=outp(1:nmeds,:).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
compute
outpimm=outp((nmeds+1):nrow(outp),:).
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).

```

```

compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),clbs3}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3(1,1)=modvnm(1,1).
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator:'/'format=A8.
print outpimm1/title =
'Mediator'/'cnames=clbs3/'rnames=cmmlbs/'form
at= F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/'space=0.
end if.
end if.
end if.
do if (boot = 0).
do if (model = 8 or model = 12).
print/title='-----'.
compute obsimm=obsprod(1:nmeds,:).
print/title = 'Indirect effect of highest order
product:'/'space=0.
print obsimm/title = 'Mediator'/'clabels =
'Effect'/'rnames = mnames/'format = F10.4.
end if.

```

```

do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'
.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'
.
end if.
do if ((imm > 0 and imm < 5) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator: '/format=A8.
end if.
compute outpimm=obsprod((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
end loop.
end if.
do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'
.
compute outpimm=obsprod(1:nmeds,1).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
compute
outpimm=obsprod((nmeds+1):nrow(obsprod),:)
.
compute outpimm={cmmval,outpimm}.

```

```

print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),'Index'}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3={modvnm(1,1),'Index'}.
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
end if.
end if.
end if.
compute conmake=0.
compute concols = 0.
do if ((model > 3 and model < 7) and (contrast
> 0) and nmods = 0 and nmeds > 1 ).
compute concols =
((ncol(indboot)*(ncol(indboot)-1))/2).
compute indcon=make(nrow(indboot),concols,-
999).
compute conkey = { ' ', ' ', ' ' }.
compute temp=1.
compute conmake=1.
loop i = 1 to (ncol(indboot)-1).

```



```

loop j = (i+1) to (ncol(indboot)).
compute indcon(:,temp)=indboot(:,i)-
indboot(:,j).
do if (contrast = 2).
compute indcon(:,temp)=abs(indboot(:,i))-
abs(indboot(:,j)).
end if.
do if (model <> 6).
compute conkey={conkey; mnames(1,i),
'minus', mnames(1,j)}.
end if.
do if (model = 6).
compute conkey={conkey; indlbl2(i,1), 'minus',
indlbl2(j,1)}.
end if.
compute temp=temp+1.
end loop.
end loop.
end if.
do if (model = 4 or model = 5).
compute clbs = {'Effect'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obs = {csum(obs); obs}.
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute rlbs = {rlbs;
cntname(1:ncol(indcon),1)}.
end if.
compute outp = obs.
compute outp2=outp.
do if (effsize = 1 and dichy = 0).
compute
pmeff(:,1)=rsum(pmeff(:,2:ncol(pmeff))).
compute
rmeff(:,1)=rsum(rmeff(:,2:ncol(rmeff))).
compute
abpseff(:,1)=rsum(abpseff(:,2:ncol(abpseff))).
compute
abcseff(:,1)=rsum(abcseff(:,2:ncol(abcseff))).

```

```

compute eff = {pmeff, rmeff, abpseff, abcseff}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute eff = {eff, r245}.
compute r245obs = {r245(1,1);r245(1,1)}.
compute kappa2ob = kappa2(1,1).
end if.
compute pmobs = t(pmeff(1,1:(nmeds+1))).
compute rmobs = t(rmeff(1,1:(nmeds+1))).
compute psobs = t(abpseff(1,1:(nmeds+1))).
compute csobs = t(abcseff(1,1:(nmeds+1))).
do if (contrast = 0).
compute outp2 = {obs, psobs, csobs, pmobs,
rmobs}.
end if.
do if (contrast > 0).
compute obs2=obs(1:nrow(psobs),:).
compute outp2 = {obs2, psobs, csobs, pmobs,
rmobs}.
end if.
compute clbs = {'ab', 'ab_ps', 'ab_cs', 'ab/c',
'ab/c'}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute outp2 = {outp2, r245obs}.
compute clbs = {clbs, 'R-sq_med'}.
do if (boot = 0 and mc = 0).
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
do if (boot = 0 and mc = 0).
do if (nmeds = 1).
compute outp2 = outp2(2,:).
compute rlbs = rlbs(2,1).
end if.
do if (model=5).
compute outp2=outp2(:,1:3).
end if.

```

```

print outp2/title = 'Indirect effect(s) of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1 and nmeds >
1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0 or mc > 0).
compute indboot = {rsum(indboot), indboot}.
compute bootSZ=boot.
do if (mc > 0).
compute bootSZ=mc.
end if.
compute ones = make(bootSZ,1,1).
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(bootSZ+1),:).
compute mnind = t(csum(indboot)/bootSZ).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/bootSZ))/(bootSZ-1))).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).
.
compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).

```

```

compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.

```

```

compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
do if (effsize = 1 and dichy = 0 and boot > 0).
compute estmte=eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).

.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.

```

```

do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
end if.
end if.
do if (boot > 0 or mc > 0).
compute outp = {obs, stdind, t(llci), t(ulci)}.
do if (nmeds = 1).
compute outp = outp(2,:).
compute rlbs = rlbs(2,1).
end if.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.

```

```

do if (mc > 0).
compute clbs = {'Effect', 'MC SE', 'MC LLCI',
'MC ULCI'}.
end if.
print outp/title = 'Indirect effect of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (dichy = 0 and effsize = 1 and boot > 0).
compute outp = {psobs,
stdindf((2*(nmeds+1)+1):(3*(nmeds+1)),1),
t(llcif(1,(2*(nmeds+1)+1):(3*(nmeds+1)))),
t(ulcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))}.
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Partially standardized indirect
effect of X on Y'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute outp = {csobs,
stdindf((3*(nmeds+1)+1):(4*(nmeds+1)),1),
t(llcif(1,(3*(nmeds+1)+1):(4*(nmeds+1)))),
t(ulcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))}.
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Completely standardized
indirect effect of X on Y'/rnames = rlbs/cnames
= clbs/format = F10.4.
compute outp = {pmobs,
stdindf(1:(nmeds+1),1),t(llcif(1,1:(nmeds+1))),
t(ulcif(1,1:(nmeds+1))) }.
do if (model = 4).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
compute outp = {rmobs, stdindf(((nmeds+1)+1)
:(2*(nmeds+1)) ,1),

```

```

t(llcif(1,((nmeds+1)+1):(2*(nmeds+1))))),
t(ulcif(1,((nmeds+1)+1):(2*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
do if (nmeds = 1 and cluster = 0 and ncovs = 0
and model = 4).
compute r245obs = r245obs(1,1).
compute outp = {r245obs,
stdindf((4*(nmeds+1)+1):(4*(nmeds+1)+1),1),
t(llcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1))),
t(ulcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1)))}.
print outp/title = 'R-squared mediation effect
size (R-sq_med)'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
end if.
do if (normal = 1).
do if (nmeds = 1).
print sobel/title = 'Normal theory tests for
indirect effect'/clabels = 'Effect', 'se', 'Z',
'p'/format = F10.4.
end if.
do if (nmeds > 1).
compute rlbs2 = rlbs(2:nrow(rlbs),1).
print sobel/title = 'Normal theory tests for
specific indirect effects'/rnames = rlbs2/clabels
= 'Effect', 'se', 'Z', 'p'/format = F10.4.
end if.
end if.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).

```



```

print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
do if (model = 6).
compute clbs = {'eff'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obsfsum=csum(obs).
compute obs = {obsfsum; obs}.
compute indlbl = indlbl(1:nrow(obs),1).
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute indlbl = {indlbl;
cntname(1:ncol(indcon),1)}.
end if.
compute obs2=obs.
do if (boot = 0).
do if (dichy = 0 and effsize = 1).
compute clbs = {'eff', 'eff_ps', 'eff_cs', 'eff/c',
'eff/c'}.
do if (ncovs = 0).
compute obs = {obs, obs/stddevy,
(obs*stddevx/stddevy), obs/ctot,
obs/mmpaths(nrow(mmpaths),1)}.
end if.
do if (ncovs > 0 or clsdmy > 0).
compute obs = {obs, obs/sdy cov,
(obs*sdx cov/sdy cov),
obs/(obsfsum+mmpaths(nrow(mmpaths),1)),
obs/mmpaths(nrow(mmpaths),1)}.
end if.
compute obs2=obs.
do if (contrast > 0).
compute obs2 = obs(1:(nrow(obs)-concols),:).
end if.
end if.

```

```

print obs2/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute indboot = {rsum(indboot), indboot}.
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(boot+1),:).
compute mnind = t(csum(indboot)/boot).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/boot))/(boot-1))).
compute temp = nrow(indboot).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).
.
compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).

```

```

compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llici(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.

```

```

end if.
end loop.
compute obs = {obs, stdind, t(llci), t(ulci)}.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
print obs/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (effsize = 1 and dichy = 0).
compute indboot=indboot(:,1:(ncol(indboot)-
concols)).
compute eff = {rsum(abpseff), abpseff,
rsum(abcseff), abcseff, rsum(pmeff), pmeff,
rsum(rmeff), rmeff}.
compute effobs = eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).

.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(effobs(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).

```

```

do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( ( effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute temp2 = stdindf(1:ncol(indboot),1).
compute temp3 = effobs(:,1:ncol(indboot)).
compute templow = llcif(1,1:ncol(indboot)).
compute temphi = ulcif(1,1:ncol(indboot)).

```

```

compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Partially standardized indirect
effect of X on Y'/cnames = clbs/rnames =
indlbl/format = F10.4.
compute temp2 =
stdindf((ncol(indboot)+1):(2*ncol(indboot)),1).
compute temp3 =
effobs(:,(ncol(indboot)+1):(2*ncol(indboot))).
compute templow =
llcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute temphi =
ulcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Completely standardized
indirect effect of X on Y'/cnames = clbs/rnames
= indlbl/format = F10.4.
compute temp2 =
stdindf((2*(ncol(indboot))+1):(3*ncol(indboot)),1
).
compute temp3 =
effobs(:,(2*(ncol(indboot))+1):(3*ncol(indboot)))
.
compute templow =
llcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute temphi =
ulcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
compute temp =
eff(:,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temp2 =
stdindf((3*(ncol(indboot))+1):(4*ncol(indboot)),1
).

```

```

compute temp3 =
effobs(:,(3*(ncol(indboot))+1):(4*ncol(indboot)))
.
compute templow =
llcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temphi =
ulcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
end if.
end if.
do if (nmeds = 2).
compute effkey = {xname, '->', mnames(1,1), '->', yname, ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, ' ', ' '}.
compute effkey = {indlbl(2:4,1), effkey}.
end if.
do if (nmeds = 3).
compute effkey = {xname, '->', mnames(1,1), '->', yname, ' ', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname, ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,3), '->', yname, ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, ' ', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', mnames(1,3), '->', yname, ' ', ' '}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, '', '', '', ''}.
compute effkey = {indl(2:8,1), effkey}.
end if.
do if (nmeds = 4).
compute effkey = {xname, '->' , mnames(1,1), '->', yname, '', '', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->', yname, '
', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->', yname, '
', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,4), '->', yname, '
', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname, '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,4), '->', yname, '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', mnames(1,4), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, '', '', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,4), '->', yname, '
', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, '', ''}.

```



```

compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', mnames(1,4), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,4), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {indlbl(2:16,1), effkey}.
end if.
print effkey/title = 'Indirect effect key'/format =
A8.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
end if.
DO IF (medte2=1).
compute
name=t(mdlnms2(2:nrow(mdlnms2),1)).
compute tmpdat=datayed(:,2:ncol(datayed)).
compute dd=tmpdat(:,1).
compute
dd={dd,tmpdat(:,(2+nmeds)),tmpdat(:,2:(1+nmeds
ds)))}.
compute stratify=0.
do if (ncovs > 0).
compute cname = cnames.
compute
dd={dd,tmpdat(:,(3+nmeds):(2+nmeds+ncovs))
}.
end if.
compute method = 1.
compute samples = boot.
do if (mc > 0).
compute samples = mc.
compute method=2.

```

```

end if.
compute omnibus = 1.
compute booterr = 0.
do if (mcx < 0 or mcx > 4).
compute mcx = 1.
end if.
compute ovals = ncol(design(dd(:,1))).
compute nte = notes.
compute criterr = 0.
compute xskip = 0.
compute dumok = 0.
compute badend = 0.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute priorlo = -9999999.
compute priorhi = 9999999.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute reps = samples.
do if (contrast = 1).
compute note(nte,1) = 26.
compute nte=nte+1.
end if.
do if (effsize = 1).
compute note(nte,1) = 27.
compute nte=nte+1.
end if.
do if (normal = 1).
compute note(nte,1) = 28.
compute nte=nte+1.
end if.
do if (method = 2).
do if (omnibus = 1).
compute note(nte,1) = 25.
compute nte=nte+1.
end if.
compute samples = 0.

```

```

do if (reps < 1000).
compute reps = 1000.
end if.
end if.
do if (method = 1).
compute samples = reps.
end if.
compute temp = ncol(dd).
compute n = nrow(dd).
compute ones = make(n,1,1).
compute temp=dd(:,2).
compute nxd = ncol(temp).
compute nx = nxd.
do if (nx = 1 and mcx > 0).
compute temp = dd.
compute temp(GRADE(dd(:,2)),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,2)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute mnvls = cmin(t(nnvls)).
do if (mnvls < 2).
compute errs = errs + 1.
compute runerrs(errs,1) = 37.
compute criterr = 1.
end if.
do if (nvls > 9).
compute errs = errs+1.
compute runerrs(errs,1) = 36.
compute criterr = 1.
end if.
do if (criterr = 0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,2).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,2) <> nnvls((temp-1),1)).
compute nnvls(temp,1)=dd(i,2).
compute temp = temp+1.

```

```

end if.
end loop.
do if (mcx > 0).
compute x = dummy(:,2:ncol(dummy)).
compute nx = ncol(x).
compute minus1 = make(1,ncol(x),-1).
compute xdes=make((nx+1),3,0).
compute xdes(1,1)=dd(1,2).
compute xdes(1,2)=1.
compute temp = 2.
loop k = 2 to n.
do if (dd(k,2) <> dd((k-1),2)).
compute xdes(temp,2) = k.
compute xdes(temp,1) = dd(k,2).
compute xdes((temp-1),3) = k-1.
compute temp=temp+1.
end if.
end loop.
compute xdes((temp-1),3)=n.
compute xdes = {xdes, (xdes(:,3)-xdes(:,2)+1)}.
do if (stratify <> 0).
compute note(nte,1) = 7.
compute nte=nte+1.
end if.
do if (mcx = 4).
loop k = 1 to n.
do if (rsum(x(k,:)) = 0).
compute x(k,:) = minus1.
end if.
end loop.
end if.
do if (mcx = 2 or mcx = 3).
compute note(nte,1)=4.
compute nte = nte+1.
loop k = 1 to n.
do if (rsum(x(k,:)) > 0).
loop i = 1 to ncol(x).
do if (x(k,i) = 0).
compute x(k,i) = 1.
else.

```

```

break.
end if.
end loop.
end if.
end loop.
do if (mcx = 3).
compute conmat1={-8,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1; 0,0,0,0,0,-
3,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.
loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute x(k,:)=conmat1((rsum(x(k,:))+1),:).
end loop.
end if.
end if.
compute xdata = x.
compute xskip = 1.
compute xname = ddd(1,1:nx).
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcx = 4).
compute dummat(1,:) = minus1.
end if.
do if (mcx = 2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcx = 3).
compute dummat=conmat1.
end if.

```

```

compute dummat={nnvls, dummat}.
compute xname2={name(1,2), xname}.
end if.
end if.
end if.
compute nm=nmeds.
compute r2b = make(nm,1,-999).
compute indeff = make((nx*nm),1,-999).
compute cov =
make((nx*nm+nm),(nx*nm+nm),0).
compute mns=make(nrow(cov),1,0).
compute ny = ncol(dd)-nm-nxd-ncovs.
compute y = dd(:,1).
compute ydata = y.
compute yname = name(1,1).
do if (xskip = 0).
compute x = dd(:,2:(1+nxd)).
compute xdata = x.
compute xname = name(1,2:(1+nxd)).
compute indlbs = t(xname).
end if.
compute m = dd(:,(2+nxd):(1+nxd+nm)).
compute mdata = m.
compute mlab = { ' M1 = ';' ' M2 = ';' ' M3 = ';' '
M4 = ';' ' M5 = ';' ' M6 = ';' ' M7 = ';' ' M8 = ';' '
M9 = ';' ' M10 = '}.
compute mlab = {mlab; ' M11 = ';' ' M12 = ';' '
M13 = ';' ' M14 = ';' ' M15 = '}.
compute mname =
name(1,(2+nxd):(1+nxd+nm)).
do if (ncovs > 0).
compute c =
dd(:,(2+nxd+nm):(1+nxd+nm+ncovs)).
compute x = {x, c}.
compute xdata = x.
compute cdata = c.
compute xname = {xname, cname}.
end if.
compute nms={yname; t(mname);
t(xname(1,1:(ncol(xname)-ncovs)))}.

```

```

do if (dumok = 1).
compute nms={yname; t(mname); name(1,2)}.
end if.
do if (nm < 16).
do if (dumok = 1).
print dummat/title = 'Coding of categorical X
variable for analysis:'/cnames = xname2/format
= F5.2.
end if.
end if.
compute
boots=make((samples+1),((nx+1)*nm),-999).
do if (errs = 0 and criterr = 0).
loop k = 1 to (samples+1).
do if (k = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef=0.
do if (k > 1).
compute u=trunc(uniform(n,1)*n)+1.
compute xdata = x(u,:).
compute mdata = m(u,:).
compute ydata = y(u,:).
do if (ncovs > 0).
compute cdata = c(u,:).
end if.
end if.
loop i = 1 to nm.
compute start = 1+(i-1)*nx.
compute xmat = {ones, xdata}.
do if (covmy=2).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = mdata(:,i).
compute amat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*amat.

```

```

compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute r2cha = r2.
compute adjr2cha = adjr2.
do if (ncovs > 0 and covmy <> 2).
compute xmat = {ones, cdata}.
compute atmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*atmat)**2).
compute r2step1 = (sstotal-ssresid2)/sstotal.
compute r2cha = r2-r2step1.
compute adjr2cha = adjr2- (1-((1-r2step1)*(n-
1)/(n-ncol(xmat)))).
compute xmat = {ones, xdata}.
end if.
compute bootcoef={bootcoef,t(amat)}.
compute r2b(i,1)=adjr2cha.
do if (k = 1).
.
compute hcinvXtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtXtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.

```



```

end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute tratio = amat&/sqrt(diag(covmat)).
compute rnms = mname(1,i).
print rnms/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.
compute lmat = ident(nrow(amat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*amat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*amat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/'format = F10.4
/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'.
compute op = {amat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/'clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/'rnames = rnms/'format =
F10.4.
compute cov(start:(start+(nx-
1)),start:(start+(nx-

```

```

1)))=covmat(2:(ncol(covmat)-(ncovs*(covmy <>
2))),2:(ncol(covmat)-(ncovs*(covmy <> 2)))).
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
=F10.4.
end if.
end if.
compute mns(start:(start+(nx-
1)),1)=amat(2:(1+nx)).
end loop.
compute xmat = {ones,mdata,xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (k = 1).
print yname/title =
*****
*****'/rlabels = 'Outcome:'/format
=A8.
do if (ovals = 2).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:'/cnames = nmsd/format = F9.2.
end if.
end if.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(ymat)/n)).
compute LL3 = ymat*&ln(pt2)+(1-ymat)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.
.
compute pt1lp= pt1.
compute xlp= xmat.

```

```

compute ylp= ymat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp&*ln(pt1lp)+(1-ylp)&*ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.

```

```

compute bootcoef={bootcoef,t(bmat)}.
compute LL1 = LL2.
compute covmat=varb.
end if.
do if (ovals <> 2).
compute bmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*bmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute bootcoef={bootcoef,t(bmat)}.
end if.
do if (k = 1).
do if (ovals = 2).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).
compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)*sqrt(diag(covmat)).
compute outp = {outp, temp}.

```

```

compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms2/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute
opdirsum=outp((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
end if.
do if (ovals <> 2).

.

compute hcinvtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(bmat)).
compute lmat = lmat(:,2:ncol(lmat)).

```

```

compute fratio =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/(ncol(xmat)-1).
compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat((2+nmeds):(2+nmeds+(nx-1)),:)
= lmat2.
compute fratiodr =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/nx.
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute df2=(n-ncol(xmat)).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), df2, pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = bmat&/sqrt(diag(covmat)).
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute op = {bmat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat)))) }.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*(xp2/(temp-
(2/3)+(.11/temp)))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms2/format =
F10.4.
compute
opdirsum=op((2+nmeds):(1+nmeds+nx),:).

```

```

compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
compute xmat = {ones,mdata}.
do if (ncovs > 0).
compute xmat = {xmat, cdata}.
end if.
compute btmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*btmat)**2).
compute r2direct = r2-((sstotal-
ssresid2)/sstotal).
compute pfr = 1-fcdf(fratiodr,(nx),(n-1-nx-nm-
ncovs)).
compute r2direct = {r2direct, fratiodr, nx, (n-1-
nx-nm-ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms2/cnames=rnms2/form
at= F10.4.
end if.
end if.
do if (k = 1 and toteff = 1).
print yname/title = '*****
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome: '/format = A8.
compute xmat = {ones, xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(ymat)/n)).
compute LL3 = ymat*ln(pt2)+(1-ymat)*ln(1-
pt2).
compute LL3 = -2*csum(LL3).

```

```

compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.

compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= ymat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)*&ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.

```



```

end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute LL1 = LL2.
compute covmat=varb.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(bmat)-
1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms = {'Constant'; t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).
compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute optotsum=outp(2:(1+nx),:).
end if.
do if (ovals <> 2).

```

```

compute cmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*cmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).

.

compute hcinvtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvtX*t( xmat )* xmat
*hcinvtX
.
compute lmat = ident(nrow(cmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).

```

```

compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = cmat&/sqrt(diag(covmat)).
compute op = {cmat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute optotsum=op(2:(1+nx),:).
compute ssresid2 = ssresid.
compute r2total = r2.
compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat(2:(1+nx),:) = lmat2.
compute fratio3 =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/nx.
do if (ncovs > 0).
compute xmat = {ones, cdata}.
compute ctmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*ctmat)**2).
compute r2total = r2-((sstotal-ssresid2)/sstotal).
end if.

```

```

compute pfr = 1-fcdf(fratio3,(nx),(n-1-nx-
ncovs)).
compute r2tot2 = {r2total, fratio3, nx, (n-1-nx-
ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns((nrow(mns)-
nm+1):nrow(mns),1)=bmat(2:(1+nm),1).
compute tmp = 1.
loop i = 1 to (nm*nx).
compute
boots(k,i)=mns(i,1)*mns(((nm*nx)+tmp),1).
do if ((i/nx)=trunc(i/nx)).
compute
boots(k,((nm*nx)+tmp))=r2b(tmp,1)*mns(((nm*
nx)+tmp),1).
compute tmp = tmp+1.
end if.
end loop.
do if (savboot = 1 and k > 1).
compute bootstrp((k-1),:)=bootcoef.
end if.
end loop.
do if (method = 1).
compute llci=make(1,ncol(boots),-999).
compute ulci=make(1,ncol(boots),-999).
compute indeff = t(boots(1,1:(nm*nx))).
compute indeff22=t(boots(1,:)).
compute omni = t(boots(1,((nm*nx)+1) :
ncol(boots))).
do if (nrow(boots) > 1).
compute boots = boots(2:nrow(boots),:).
compute bootse = t(sqrt((((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).

```

```

loop i = 1 to ncol(boots).

.

compute temp = boots(:,i).
compute temp(GRADE( boots(:,i) )) = boots(:,i).
compute badlo = 0.
compute badhi = 0.
do if ( (indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(indeff22(i,1)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.

```

```

do if ( (indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,i)=llcit.
compute ulci(1,i)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute llci = t(llci).
compute ulci= t(ulci).
compute llci={llci,ulci}.
end if.
end if.
do if (method = 2).
compute tmp=1.
loop i = 1 to (nm*nx).
compute
indeff(i,1)=mns(i,1)*mns(((nm*nx)+tmp),1).
do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = sqrt(-
2*ln(uniform(reps,nrow(cov))))&*cos((2*3.1415
9265358979)*uniform(reps,nrow(cov))).
compute boots=boots*chol(cov).
loop i = 1 to ncol(boots).
compute boots(:,i)=boots(:,i)+mns(i,1).
end loop.
compute tmp=1.
loop i = 1 to (nm*nx).

```

```

compute
boots(:,i)=boots(:,i)*boots(:,((nm*nx)+tmp)).
compute temp = boots(:,i).
compute temp(GRADE(boots(:,i))) = boots(:,i).
compute boots(:,i) = temp.
do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = boots(:,1:(nm*nx)).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
compute llci =
t(((boots(cilow,:));(boots(cihigh,:)))).
end if.
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
compute rnmssum=rnms(2:(1+nx),1).
do if (ovals <> 2).
print optotsum/title = 'Relative total effects of X
of Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.
print r2tot2/title = 'Omnibus test of total effect of
X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print optotsum/title = 'Relative total effects of X
on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
end if.
compute
rnmssum=rnms2((2+nmeds):(1+nmeds+nx),1).
do if (ovals <> 2).

```

```

print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.
print r2direct/title = 'Omnibus test of direct
effect of X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/'space=0.
do if (omnibus = 1 and (method = 1 or method
= 3)).
compute indlbs = {indlbs; 'Omnibus'}.
end if.
loop i = 1 to nm.
compute op = mname(1,i).
print op/title = 'Relative indirect effect(s) of X on
Y through:.'/format = A8/space=0.
do if (method = 2).
compute clbs = {'Effect', 'SE MC', 'MC LLCI',
'MC ULCI'}.
compute op = {indeff, bootse(1:(nm*nx)), llci}.
end if.
do if (method = 1).
compute clbs = {'Effect', 'SE(boot)', 'LLCI',
'ULCI'}.
compute op = indeff.
do if (samples > 0).
compute op = {indeff, bootse(1:(nm*nx).:),
llci(1:(nm*nx).:)};.
end if.
end if.
compute temp = op((((i-1)*nx)+1):(nx*i).:).
do if ((method = 1) and omnibus = 1).
do if (samples = 0).
compute temp={temp;omni(i,1)}.
end if.

```



```

do if (samples > 0).
compute temp = {temp; omni(i,1),
bootse(((nm*nx)+i,:), llci(((nm*nx)+i,:),:))}.
end if.
end if.
print temp/title = ' '/rnames = indlbs/cnames =
clbs/format = F10.4 /space=0.
print/title = '-----'.
end loop.
end if.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
END IF.
DO IF (mdpbe=1).
compute nm={yname,xname,mnames}.
do if (ncovs > 0).
compute
nm={yname,cnames,xname,mnames}.
end if.
compute tpx=xname.
compute x = dd(:,2:ncol(dd)).
compute tpy = yname.
compute tempy= dd(:,1).
compute n = nrow(dd).
compute nx=1.
compute criterr=0.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute ddd1 = {'int_1', 'int_2', 'int_3', 'int_4',
'int_5', 'int_6', 'int_7', 'int_8', 'int_9'}.

```

```

compute dumok = 0.
compute mcfoc=mcx.
compute mcmmod=mcm.
compute mcloc=(mcfoc>0).
compute con = make(n,1,1).
compute ncovs=ncol(x)-2.
do if (mcfoc > 0 or mcmmod > 0).
compute temp = dd.
compute temp(GRADE(dd(:,(ncol(dd)-
mcloc))),:)= dd.
compute dd = temp.
compute dummy = design(dd(:,ncol(dd)-
mcloc)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute toosmall=rsum(nnvls < 2).
compute mnvls = cmin(t(nnvls)).
do if ((rsum(nnvls < 2)) > 0).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=37.
end if.
do if (nvls > 10).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=36.
end if.
do if (criterr=0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,ncol(dd)-mcloc).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,ncol(dd)-mcloc) <> nnvls((temp-
1),1)).
compute nnvls(temp,1)=dd(i,ncol(dd)-mcloc).
compute temp = temp+1.
end if.
end loop.
compute dummy = dummy(:,2:ncol(dummy)).

```

```

do if (mcfoc=4 or mcmmod=4).
compute minus1=make(1,ncol(dummy),-1).
loop k = 1 to n.
do if (rsum(dummy(k,:)) = 0).
compute dummy(k,:)=minus1.
end if.
end loop.
end if.
do if (mcfoc = 2 or mcfoc = 3) or (mcmmod = 2 or
mcmmod = 3)).
loop k = 1 to n.
do if (rsum(dummy(k,:)) > 0).
loop i = 1 to ncol(dummy).
do if (dummy(k,i) = 0).
compute dummy(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcfoc = 3 or mcmmod=3).
compute conmat1={-8,1,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1,1; 0,0,0,0,0,-
3,1,1,1,1; 0,0,0,0,0,0,-2,1,1,1; 0,0,0,0,0,0,0,-1,1}.
loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute
dummy(k,:)=conmat1((rsum(dummy(k,:))+1),:).
end loop.
end if.
end if.
compute nx = ncol(dummy).
compute xname = ddd(1,1:nx).
compute xdata = dummy.

```

```

compute xname = ddd(1,1:nx).
compute xname2={nm(1,(ncol(nm)-mcloc)),
xname}.
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcfoc = 2 or mcmmod=2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcfoc = 3 or mcmmod=3).
compute dummat=conmat1.
end if.
do if (mcfoc = 4 or mcmmod = 4).
compute dummat(1,:)=minus1.
end if.
compute dummat={nnvls, dummat}.
end if.
end if.
do if (criterr=0).
compute y = dd(:,1).
compute ovals = ncol(design(dd(:,1))).
compute itprob = 0.
do if (ovals = 2).
compute omx = cmax(y(:,1)).
compute omn = cmin(y(:,1)).
compute y(:,1) = (y(:,1) = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute sstotal = csum((y-(csum(y)/n))&**2).
compute outv = t(nm(1,1)).
compute mdtr = nm(1,ncol(dd)).
compute fciv = nm(1,(ncol(dd)-1)).
do if (mcfoc <> 0 or mcmmod <> 0).
compute inter=make(n,nx,0).
loop i = 1 to nx.

```

```

compute inter(:,i)=dummy(:,i)*dd(:,ncol(dd)-
(1-mcloc)).
end loop.
do if (ncovs = 0).
compute x={con,dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
do if (ncovs > 0).
compute
x={con,dd(:,2:(1+ncovs)),dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={t(nm(1,2:(1+ncovs)));
nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
end if.
compute xmns = csum(x)/n.
compute focvals = {1,0,0}.
compute highwarn = 0.
compute lowwarn = 0.
do if (ncol(x) > (2*nx+2)).
compute covmns = {1, xmns(1,2:(ncol(x)-1-
(2*nx)))}.
compute focvals = {covmns,0,0}.
end if.
compute dfres = n-ncol(x).
do if (ovals = 2).
compute pt2 = make(n,1,(csum(y)/n)).
compute LL3 = y*&ln(pt2)+(1-y)*&ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(x),1,0).
compute LL1 = 0.

```

```

.
compute pt1lp= pt1.
compute xlp= x.
compute ylp= y.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)*&ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.

```

```

end if.
compute tval=xp2*xp2.
else if (ovals <> 2).
compute invXtX=inv(t(x)*x).
compute b = invXtX*t(x)*y.
compute resid = y-(x*b).
compute ssresid=csum(resid&*resid).
compute msresid=ssresid/dfres.
compute varb = msresid*invXtX.
compute
seint=sqrt(varb(nrow(varb),nrow(varb))).
compute k3 = nrow(b).
do if (hc3 = 1).
compute xhc=x.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)= xhc(i3,:)*invXtX*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).
end loop.
compute varb=(invXtX*t(xhc)*xhc*invXtX).
end if.
compute seb = sqrt(diag(varb)).
compute r2 = 1-(ssresid/sstotal).
do if (mcfoc <> 0 or mcmmod <> 0).
compute ytp=y.
compute xtp=x(:,1:(ncol(x)-nx)).
compute btp = inv(t(xtp)*xtp)*t(xtp)*ytp.
compute residt = ytp-(xtp*btp).
compute ssresidt=csum(residt&*residt).
compute r2noint = 1-(ssresidt/sstotal).
end if.
compute pr = ncol(x)-1.
compute lmat = ident(nrow(b)).
compute lmat = lmat(:,2:ncol(lmat)).
compute f =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
pr).

```

```

compute pf = 1-fcdf(f,pr,dfres).
compute pf = {sqrt(r2),r2,msresid,f,pr,dfres,pf}.
compute xd=abs(xp2).
end if.
do if (itprob = 0).
compute tstat = b&/seb.
do if (ovals <> 2).
compute p = 2*(1-tcdf(abs(tstat), dfres)).
else if (ovals = 2).
compute p = 2*(1-cdfnorm(abs(tstat))).
end if.
compute outp = {b,seb,tstat,p}.
compute nms = {'constant'; nms; 'interact'}.
do if (ovals <> 2).
compute tval = (dfres* (exp((dfres-
(5/6))*((xd/(dfres-(2/3)+(.11/dfres)))*(xd/(dfres-
(2/3)+(.11/dfres)))))-1)).
compute outp={outp,(b-
sqrt(tval)&*seb),(b+sqrt(tval)&*seb)}.
end if.
compute bb = tval.
do if (dumok = 1).
do if (mcfoc > 0).
print dummat/title = 'Coding of categorical X
variable for analysis:.'/cnames = xname2/format
= F5.2.
else if (mcmod > 0).
print dummat/title = 'Coding of categorical M
variable for analysis:.'/cnames = xname2/format
= F5.2.
end if.
end if.
print yname/title =
*****
*****'/rlabels = 'Outcome:.'/format
= A8.
do if (ovals = 2).
compute nmsd = {outv, 'Analysis'}.
print rcd/title = 'Coding of binary Y for
analysis:.'/cnames = nmsd/format = F10.4.

```



```

compute outp = {outp,(outp(:,1)-
sqrt(tval)&*outp(:,2)),(outp(:,1)+sqrt(tval)&*outp
(:,2))}.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
print outp/title = 'Logistic Regression
Model'/rnames = nms/clabels 'Coeff' 'se' 'Z' 'p'
'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
compute varbtmp=varb.
compute btmp=b.
compute LL2f=LL2.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=x(:,1:(ncol(x)-nx)).
compute bt1 = make(ncol(xlp),1,0).

.

compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).

```

```

compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&*ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL2f.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(LLdiff,nx).
compute rcha={LLdiff,nx,pchi}.
end if.
do if (ovals <> 2).
print pf/title = 'Model Summary'/clabels = 'R' 'R-
sq' 'MSE' 'F' 'df1' 'df2' 'p'/format F10.4.
compute lmat=make(nrow(b),1,0).

```

```

compute lmat(nrow(lmat),1)=1.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
1).
compute
rcha=((b(nrow(b),1)/seint)*(b(nrow(b),1)/seint))*
(1-r2)/dfres.
compute rcha = {rcha, fcha, 1, dfres,
outp((pr+1),4)}.
print outp/title = 'Model'/rnames = nms/clabels
'coeff' 'se' 't' 'p' 'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute rcha=r2-r2noint.
compute lmat=make((nrow(b)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
nx).
compute pvalr2c=1-fcdf(fcha,nx,dfres).
compute rcha = {rcha, fcha, nx, dfres,pvalr2c}.
end if.
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute intkey = {'a', 'b', 'c', 'd', 'e'}.
loop i = 1 to nx.
compute intkey={intkey; ddd1(1,i), ' : ', ddd1(1,i),
' X ', nm(1,ncol(dd)-(1-mcloc))}.
end loop.
compute intkey=intkey(2:nrow(intkey),:).
print intkey/title='Product terms key:.'/format =
a8.
end if.
do if (ovals <> 2).

```

```

print rcha/title = 'R-square increase due to
interaction:.'/clabels 'R2-chng' 'F' 'df1' 'df2'
'p'/format F10.4.
else if (ovals = 2).
print rcha/title = 'Likelihood ratio test for
interaction:.'/clabels 'Chi-sq' 'df' 'p'/format F10.4.
end if.
print/title =
*****
*****!
compute mdvar = x(:,(ncol(x)-1)).
do if (mcfoc = 0 and mcmod = 0).
compute g1 = b((ncol(x)-2),1).
compute g3 = b(ncol(x),1).
compute vg1 = varb((ncol(x)-2),(ncol(x)-2)).
compute vg3 = varb(ncol(x),ncol(x)).
compute covg1g3 = varb((ncol(x)-2), ncol(x)).
end if.
do if (mcfoc > 0 or mcmod > 0).
compute mdvar = x(:,(ncol(x)-(2*nx))).
end if.
compute mdmin = cmin(mdvar).
compute mdmax = cmax(mdvar).
compute fvar = x(:,(ncol(x)-2)).
compute nval = ncol(design(mdvar)).
compute fvmin = cmin(fvar).
compute fvmax = cmax(fvar).
do if (mmodval = 999).
compute mnmd = csum(mdvar)/n.
compute tmp = make(n,1,mnmd).
compute sdmd = sqrt(csum(((mdvar-
tmp)**2))/(n-1)).
compute probeval = {mnmd-sdmd; mnmd;
mnmd+sdmd}.
do if (probeval(1,1) < mdmin).
compute lowwarn = 1.
compute probeval(1,1)=mdmin.
end if.
do if (probeval(nrow(probeval),1) > mdmax).
compute highwarn = 1.

```

```

compute probeval(nrow(probeval),1)=mdmax.
end if.
do if (quantile = 1).
compute tmp = mdvar.
compute tmp(GRADE(mdvar(:,1)),:)= mdvar.
compute mdvar = tmp.
compute
probeval={mdvar(trunc(0.1*n),1);mdvar(trunc(0
.25*n),1);mdvar(trunc(0.5*n),1);mdvar(trunc(0.7
5*n),1);mdvar(trunc(0.9*n),1)}.
end if.
end if.
do if (nval = 2).
compute probeval = make(2,1,0).
compute probeval(1,1)=cmin(mdvar).
loop i = 1 to n.
do if (mdvar(i,1) <> probeval(1,1)).
compute probeval(2,1) = mdvar(i,1).
BREAK.
end if.
end loop.
end if.
do if (mmodval <> 999).
compute probeval = mmodval.
end if.
compute outp = make(nrow(probeval),7,0).
do if (mcfoc > 0 or mcmmod > 0).
compute focvals=make(1,ncol(x)+1,1).
do if (mcfoc > 0).
print/title = 'Conditional effect of X on Y at
values of the moderator:'.
print/title = ' '/space=0.
compute rnn2=mdtr.
compute matt=make(nx,6,0).
end if.
loop jj=1 to nrow(probeval).
do if (mcfoc > 0).
loop ii=1 to nx.
compute g1=b((ncol(x)-(2*nx)+ii),1).
compute g3=b((ncol(x)-nx+ii),1).

```

```

compute vg1=varb((ncol(x)-(2*nx)+ii),(ncol(x)-
(2*nx)+ii)).
compute vg3=varb((ncol(x)-nx+ii),(ncol(x)-
nx+ii)).
compute covg1g3=varb((ncol(x)-
(2*nx)+ii),(ncol(x)-nx+ii)).
compute x2 = probeval(jj,1).
compute w1 = g1+g3*x2.
compute varw1 =
vg1+(2*x2*covg1g3)+((x2*x2)*vg3).
compute sew1 = sqrt(varw1).
compute t1 = w1/sew1.
do if (ovals <> 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-tcdf(abs(t1), dfres)).
compute cnms = {'Coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)={w1,sew1,t1,p,llci,ulci}.
else if (ovals = 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-cdfnorm(abs(t1))).
compute cnms = {'Coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)={w1,sew1,t1,p,llci,ulci}.
end if.
end loop.
compute rnms=t(xname).
compute mdvalpr=probeval(jj,1).
print mdvalpr/title = 'Moderator
value:.'/rnames=rnn2/format= F10.4 /space=0.
print matt/title='
'/cnames=cnms/rnames=rnms/format= F10.4
/space=0.
compute xprob=x(:,(ncol(x)-(2*nx)))-mdvalpr.
loop kk = 1 to nx.
compute xprob={xprob,
(xprob(:,1)&*x(:,(ncol(x)-(2*nx)+kk)))}.
end loop.

```

```

compute xprob={x(:,1:((ncol(x)-(2*nx)-
1)),xprob}.
do if (ovals <> 2).
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute residc = y-(xprob*bmultc).
compute ssresidc=csum(residc*residc).
compute r2c = r2-(1-(ssresidc/sstotal)).
compute fcha2=(dfres*r2c)/(nx*(1-r2)).
do if (hc3 = 1).
loop kk = 1 to nx.
compute xprob={xprob, x(:,(ncol(x)-
(2*nx)+kk))}.
end loop.
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute k3 = nrow(bmultc).
compute xhc=xprob.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)=
xhc(i3,)*inv(t(xprob)*xprob)*t(xhc(i3,)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).
end loop.
compute
varbc=(inv(t(xprob)*xprob)*t(xhc)*xhc*inv(t(xpr
ob)*xprob)).
compute lmat=make((nrow(bmultc)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha2 =
(t(t(lmat)*bmultc)*inv(t(lmat)*varbc*lmat)*((t(lm
at)*bmultc)))/nx).
end if.
compute pvalr2cc=1-fcdf(fcha2,nx,dfres).
compute rcha2 = {r2c, fcha2, nx,
dfres,pvalr2cc}.

```

```

print rcha2/title = 'Test of equality of conditional
means at this value of the moderator'/clabels
'R2-chng' 'F' 'df1' 'df2' 'p'/format F10.4.
else if (ovals = 2).
compute btmp=b.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=xprob.
compute bt1 = make(ncol(xlp),1,0).

.

compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*ln(pt1lp)+(1-ylp)*ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).

```



```

compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL4.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(LLdiff,nx).
compute rcha2 = {LLdiff, nx,pchi}.
print rcha2/title = 'Test of equality of log odds
conditioned on this moderator value'/clabels
'Chi-sq' 'df' 'p'/format F10.4.
end if.
end if.
compute
tttt=make(nrow(dummat),1,probeval(jj,1)).
compute tttt={tttt,dummat(:,2:ncol(dummat))}.
loop kkk=1 to nx.
compute tttt={tttt,tttt(:,1)*tttt(:,1+kkk)}.
end loop.
compute ones=make(nrow(dummat),1,1).
do if (ncol(x) > (2*nx+2)).
compute
covmnmat=make(nrow(tttt),ncol(covmns),0).
loop kkk=1 to nrow(tttt).
compute covmnmat(kkk,:)=covmns.
end loop.
compute tttt={covmnmat,tttt}.
else.
compute tttt={ones,tttt}.
end if.
compute focvals={focvals;dummat(:,1),tttt}.
do if (mcfoc > 0).
compute yhat={dummat(:,1),(tttt*b)}.
compute cmnms={fciv, 'yhat'}.
do if (ovals <> 2).

```

```

print yhat/title = 'Estimated conditional means
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (ovals = 2).
print yhat/title = 'Estimated conditional log odds
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (jj <> nrow(probeval)).
print/title='-----'/space=0.
end if.
end if.
end loop.
compute focvals=focvals(2:nrow(focvals),:).
compute yhat=focvals(:,2:ncol(focvals))*b.
compute
focvals={focvals(:,1),focvals(:,(ncol(focvals)-
(2*nx))),yhat}.
compute cnms={fciv,mdtr,'yhat'}.
do if (mcmod > 0).
compute cnms={mdtr,fciv,'yhat'}.
end if.
end if.
do if (mcmod > 0).
compute outp=make((nx+1),7,0).
compute bcatm={b((ncol(x)-
(2*nx)),1);b((ncol(x)-nx+1):ncol(x),1)}.
compute outp(:,1)=dummat(:,1).
compute bcatcov=varb((ncol(x)-
nx):ncol(x),(ncol(x)-nx):ncol(x)).
compute bcatcov(1,1)=varb((ncol(x)-
(2*nx)),(ncol(x)-(2*nx))).
compute
bcatcov(2:nrow(bcatcov),1)=varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx)).
compute
bcatcov(1,2:nrow(bcatcov))=t(varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx))).
loop i = 1 to nrow(dummat).

```

```

compute
catmval={1,dummat(i,2:ncol(dummat))}.
compute condeff=catmval*bcatm.
compute
condse=sqrt(catmval*bcatcov*t(catmval)).
compute outp(i,2:3)={condeff,condse}.
end loop.
compute outp(:,4)=outp(:,2)&/outp(:,3).
compute outp(:,5) = 2*(1-tcdf(abs(outp(:,4)),
dfres)).
compute outp(:,6) = (outp(:,2)-
sqrt(tval)&*outp(:,3)).
compute outp(:,7) =
(outp(:,2)+sqrt(tval)&*outp(:,3)).
do if (ovals <> 2).
compute cnmms = {xname2(1,1), 'coeff', 'se',
't', 'p', 'LLCI', 'ULCI'}.
end if.
do if (ovals = 2).
compute outp(:,5) = 2*(1-
cdfnorm(abs(outp(:,4)))).
compute cnmms = {xname2(1,1), 'coeff', 'se',
'Z', 'p', 'LLCI', 'ULCI'}.
end if.
print outp/title = 'Conditional Effect of Focal
Predictor in Groups Defined by the Moderator
Variable: '/cnames=cnmms/format = F10.4.
end if.
do if (nval > 2 and (mmodval = 999) and
mcmmod = 0).
do if (quantile <> 1).
print/title = 'Moderator values are the sample
mean and plus/minus one SD from mean'.
else.
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
end if.
do if (highwarn = 1 and quantile = 0).

```

```

print/title = 'NOTE: For the moderator values
above, one SD above the mean was'.
print/title = 'replaced with the maximum
because one SD above the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
do if (lowwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD below the mean was'.
print/title = 'replaced with the minimum
because one SD below the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
end if.
do if (nval = 2 and (mmodval = 999)).
print/title = 'The moderator variable is
dichotomous'.
end if.
do if (mmodval <> 999 and mcmmod = 0 and
((mmodval < mdmin) or (mmodval > mdmax))).
print/title = 'Warning: Moderator value specified
is outside of the range of the data'.
end if.
do if (plot = 1).
do if (mcfoc = 0 and mcmmod=0).
compute cnms = {t(nms((ncol(x)-2):(ncol(x)-
1),1)), 'yhat'}.
end if.
do if (ovals = 2).
compute prob =
exp(focvals(:,3))/(1+exp(focvals(:,3))).
compute focvals = {focvals, prob}.
compute cnms = {cnms, 'prob'}.
end if.
print/title =
*****
*****!

```

```
print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (ovals = 2).
do if (mcfoc=0).
print/title = 'DATA LIST FREE/Eco_Innovation
Comm_Pract logodds prob.' /space=1.
end if.
do if (mcmod=0).
print/title = 'DATA LIST FREE/Comm_Pract
Eco_Innovation logodds prob.' /space=1.
end if.
else if (ovals <> 2).
do if (mcfoc=0).
print/title = 'DATA LIST FREE/Eco_Innovation
Comm_Pract Operational_Performance.'
/space=1.
end if.
do if (mcmod=0).
print/title = 'DATA LIST FREE/Comm_Pract
Eco_Innovation Operational_Performance.'
/space=1.
end if.
end if.
print/title = 'BEGIN DATA.' /space=0.
print focvals/title = ' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
```

```

do if (ovals <> 2 and mcmmod=0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Operational_Performance BY
Comm_Pract.' /space=0.
end if.
do if (ovals <> 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
Operational_Performance BY Eco_Innovation.'
/space=0.
end if.
do if (ovals = 2 and mcmmod=0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Comm_Pract.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Comm_Pract.' /space=0.
end if.
do if (ovals = 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
logodds BY Eco_Innovation.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
prob BY Eco_Innovation.' /space=0.
end if.
do if (ncovs > 0).
print/title = 'NOTE: For data above, covariates
are set to their sample means.'.
end if.
end if.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.

```

```

compute runerrs(errs,1) = 22.
end if.
end if.
END IF.
end if.
do if (bad > 0).
compute note(notes,1) = 9.
compute notes = notes + 1.
end if.
print/title = '***** ANALYSIS
NOTES AND WARNINGS
*****'.

loop i = 1 to errs.
do if (runerrs(i,1) = 1).
print/title = 'ERROR: One of your declared
mediators is dichotomous. This procedure
can"t be used.'.
end if.
do if (runerrs(i,1) = 2).
print/title = 'ERROR: For model 6, this
procedure limits the number of mediators to
four.'.
end if.
do if (runerrs(i,1) = 3).
print/title = 'ERROR: For models 1, 2, and 3,
only a single variable can be listed in the M
list.'.
end if.
do if (runerrs(i,1) = 4).
print/title = 'ERROR: You requested a model
involving W but did not provide a valid W
variable name.'.
end if.
do if (runerrs(i,1) = 5).
print/title = 'ERROR: You requested a model
involving Z but did not provide a valid Z
variable name.'.
end if.
do if (runerrs(i,1) = 6).

```

```
print/title = 'ERROR: You requested a model
involving Q but did not provide a valid Q
variable name.'.
end if.
do if (runerrs(i,1) = 7).
print/title = 'ERROR: You requested a model
involving V but did not provide a valid V
variable name.'.
end if.
do if (runerrs(i,1) = 8).
print/title = 'ERROR: You specified a W
variable for a model that does not need it'.
end if.
do if (runerrs(i,1) = 9).
print/title = 'ERROR: You specified a Z variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 10).
print/title = 'ERROR: You specified a Q variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 11).
print/title = 'ERROR: You specified a V variable
for a model that does not need it.'.
end if.
do if (runerrs(i,1) = 12).
print/title = 'ERROR: The variable specified for
W has already been assigned.'.
end if.
do if (runerrs(i,1) = 13).
print/title = 'ERROR: The variable specified for
Z has already been assigned.'.
end if.
do if (runerrs(i,1) = 14).
print/title = 'ERROR: The variable specified for
Q has already been assigned.'.
end if.
do if (runerrs(i,1) = 15).
print/title = 'ERROR: The variable specified for
V has already been assigned.'.
```



```
end if.  
do if (runerrs(i,1) = 16).  
print/title = 'ERROR: You did not provide a  
valid Y variable name.'  
end if.  
do if (runerrs(i,1) = 17).  
print/title = 'ERROR: The variable specified for  
Y has already been assigned.'  
end if.  
do if (runerrs(i,1) = 18).  
print/title = 'ERROR: Model 6 requires more  
than one mediator.'  
end if.  
do if (runerrs(i,1) = 19).  
print/title = 'ERROR: You have not specified a  
valid model number.'  
end if.  
do if (runerrs(i,1) = 20).  
print/title = 'ERROR: At least one and only one  
variable must be listed for X.'  
end if.  
do if (runerrs(i,1) = 21).  
print/title = 'ERROR: At least one and only one  
variable must be listed for Y.'  
end if.  
do if (runerrs(i,1) = 22).  
print/title = 'ERROR: Iteration didn't converge  
to a solution. Interpret results with caution.'  
end if.  
do if (runerrs(i,1) = 23).  
print/title = 'ERROR: You specified a clustering  
variable that does not exist in your variable  
list.'  
end if.  
do if (runerrs(i,1) = 24).  
print/title = 'ERROR: You specified a clustering  
variable that has already been assigned.'  
end if.  
do if (runerrs(i,1) = 25).
```

```

print/title = 'ERROR: One or more of your M
variables is not listed in the variables list.'.
end if.
do if (runerrs(i,1) = 26).
print/title = 'ERROR: A maximum of 20 cluster
units is allowed. Use multilevel modeling
instead.'.
end if.
do if (runerrs(i,1) = 27).
print/title = 'ERROR: One of the variables in
your model is a constant.'.
end if.
do if (runerrs(i,1) = 28).
print/title = 'ERROR: Dichotomous Y is not
permitted with WS option.'.
end if.
do if (runerrs(i,1) = 29).
print/title = 'ERROR: Insufficient number of
variables in vars= list when using WS option.'.
end if.
do if (runerrs(i,1) = 30).
print/title = 'ERROR: Too many variables in
vars= list when using WS option. Covariates
not allowed.'.
end if.
do if (runerrs(i,1) = 31).
print/title = 'ERROR: mmodval and wmodval
can"t both be set to zero with contrast option.'.
end if.
do if (runerrs(i,1) = 32).
print/title = 'ERROR: You did not provide a
valid X variable name.'.
end if.
do if (runerrs(i,1) = 33).
print/title = 'ERROR: PROCESS requires all
variable names to be eight characters or
fewer.'.
print/title = '      Please shorten variable names
and reexecute.'/space=0.
end if.

```

```

do if (runerrs(i,1)=34).
print/title = 'ERROR: X and M cannot both be
specified as multicategorical.'.
end if.
do if (runerrs(i,1)=35).
print/title = 'ERROR: Multicategorical variable
specification available only for models 1 and
4.'.
end if.
do if (runerrs(i,1)=36).
print/title = 'ERROR: Categorical variables
cannot have more than 10 categories.'.
end if.
do if (runerrs(i,1)=37).
print/title = 'ERROR: Each group must have at
least two cases.'.
end if.
do if (runerrs(i,1)=38).
print/title = 'ERROR: Cluster option not
available with multicategorical variables.'.
end if.
do if (runerrs(i,1)=39).
print/title = 'ERROR: A maximum of 10
variables is allowed in the M= list.'.
end if.
do if (runerrs(i,1) = 41).
print/title = 'ERROR: MCM option not permitted
in model 4.'.
end if.
end loop.
do if (errs = 0).
do if (boot > 1 or mc > 0).
do if (bconoff = 1 and boot > 1).
print boot/title = 'Number of bootstrap samples
for bias corrected bootstrap confidence
intervals:'//format = F8.0.
end if.
do if (bconoff = 0 and boot > 1).

```

```

print boot/title = 'Number of bootstrap samples
for percentile bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (mc > 1).
print mc/title = 'Number of samples for Monte
Carlo confidence intervals:'/format = F8.0.
end if.
do if (booterr = 1).
compute badend = badend(1,2:ncol(badend)).
print badend/title = 'WARNING: Bootstrap CI
endpoints below not trustworthy. Decrease
confidence or increase bootstraps'/format =
F10.4.
end if.
end if.
print conf/title = 'Level of confidence for all
confidence intervals in output:'/format = F8.2.
do if ((center = 1 or ws=1) and (ncol(centvar) >
1)).
compute centvar = centvar(1,2:ncol(centvar)).
print centvar/title = 'NOTE: The following
variables were mean centered prior to
analysis:'/format = a8.
end if.
loop i = 1 to notes.
do if (note(i,1) = 1).
print/title = 'NOTE: Confidence level restricted
to between 50 and 99.9999%. 95% confidence
is provided in output'.
end if.
do if (note(i,1) = 2).
print/title = 'NOTE: Effect size measures for
indirect effects are not available for models
with dichotomous outcomes'.
end if.
do if (note(i,1) = 3).
print/title = 'NOTE: All standard errors for
continuous outcome models are based on the
HC3 estimator'.

```

```

end if.
do if (note(i,1) = 6).
print/title = 'NOTE: The number of bootstrap
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 7).
print/title = 'NOTE: The Johnson-Neyman
method is available only for Models 1 and 3'.
end if.
do if (note(i,1) = 8).
print/title = 'NOTE: The Johnson-Neyman
method cannot be used with a dichotomous
moderator'.
end if.
do if (note(i,1)=9).
print bad/title = 'NOTE: Some bootstrap
samples had to be replaced. The number of
such replacements was:'.
end if.
do if (note(i,1) = 11).
print nmiss/title = 'NOTE: Some cases were
deleted due to missing data. The number of
such cases was:'.
end if.
do if (note(i,1) = 12).
print /title = 'NOTE: Monte Carlo method
available only for models 4 and 5.
Bootstrapping was used instead.'.
end if.
do if (note(i,1) = 13).
print/title = 'NOTE: The number of Monte Carlo
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 19).
print/title = 'NOTE: Effect sizes not available for
within-subject analyses.'.
end if.
do if (note(i,1) = 16).

```

```

print/title = 'NOTE: Normal theory tests not
available for within-subject analyses.'.
end if.
do if (note(i,1) = 17).
print/title = 'NOTE: Monte Carlo confidence
intervals not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 18 and warnrep=0).
print/title = 'WARNING: You have requested
OLS estimation with a dichotomous criterion.'.
print/title = 'Interpret model coefficients and
inferential statistics with caution.'/space=0.
compute warnrep=1.
end if.
do if (note(i,1) = 20).
print/title = 'NOTE: Saving of bootstrap
estimates not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 22).
print/title = 'NOTE: Effect size option with
covariates requires covariates in models of M
and Y.'.
end if.
do if (note(i,1) = 23).
print/title = 'NOTE: Johnson-Neyman method
not available with multicategorical X'.
end if.
do if (note(i,1) = 24).
print/title = 'NOTE: MMODVAL option not
available with a multicategorical moderator'.
end if.
do if (note(i,1) = 25).
print/title = 'NOTE: Omnibus test for relative
indirect effects is not available with the MC
option.'.
end if.
do if (note(i,1) = 26).

```

| | | |
|-----------|----------------|---|
| | | <pre> print/title = 'NOTE: CONTRAST option not available with multicategorical X.'. end if. do if (note(i,1) = 27). print/title = 'NOTE: EFFSIZE option not available with multicategorical X.'. end if. do if (note(i,1) = 28). print/title = 'NOTE: NORMAL option not available with multicategorical X.'. end if. do if (note(i,1) = 31). print/title = 'NOTE: The TOTAL option is not available when Y is dichotomous.'. end if. do if (note(i,1) = 32). print/title = 'NOTE: Kappa-squared is disabled from output as of version 2.16.'. end if. do if (note(i,1) = 33). print/title = 'NOTE: COVMY option ignored in models 1, 2, and 3.'. end if. end loop. end if. end matrix. </pre> |
| Resources | Processor Time | 00:00:04.50 |
| | Elapsed Time | 00:00:04.50 |

Run MATRIX procedure:

***** PROCESS Procedure for SPSS Release 2.16.3 *****

Written by Andrew F. Hayes, Ph.D. www.afhayes.com
Documentation available in Hayes (2013). www.guilford.com/p/hayes3

Model = 4
Y = Operatio
X = Comm_Pra
M = Eco_Inno

Sample size

Outcome: Eco_Inno

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|-----------|--------|----------|-------|
| .8312 | .6909 | .3566 | 1908.4213 | 1.0000 | 854.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|-------|-------|---------|-------|-------|-------|
| constant | .5914 | .0753 | 7.8534 | .0000 | .4436 | .7392 |
| Comm_Pra | .8310 | .0190 | 43.6855 | .0000 | .7937 | .8683 |

Outcome: Operatio

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|-----------|--------|----------|-------|
| .8587 | .7374 | .3150 | 1197.8948 | 2.0000 | 853.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|-------|-------|---------|-------|-------|-------|
| constant | .3040 | .0733 | 4.1480 | .0000 | .1602 | .4479 |
| Eco_Inno | .3086 | .0322 | 9.5948 | .0000 | .2455 | .3718 |
| Comm_Pra | .6018 | .0322 | 18.7120 | .0000 | .5386 | .6649 |

***** DIRECT AND INDIRECT EFFECTS *****

Direct effect of X on Y

| Effect | SE | t | p | LLCI | ULCI |
|--------|-------|---------|-------|-------|-------|
| .6018 | .0322 | 18.7120 | .0000 | .5386 | .6649 |

Indirect effect of X on Y

| | Effect | Boot SE | BootLLCI | BootULCI |
|----------|--------|---------|----------|----------|
| Eco_Inno | .2565 | .0219 | .2163 | .3024 |

Normal theory tests for indirect effect

| Effect | se | Z | p |
|--------|-------|--------|-------|
| .2565 | .0274 | 9.3691 | .0000 |

***** ANALYSIS NOTES AND WARNINGS *****

Number of bootstrap samples for bias corrected bootstrap confidence intervals:

5000

Level of confidence for all confidence intervals in output:

95.00

----- END MATRIX -----

restore.

Matrix

| Notes | | |
|----------------|--------------------------------|----------------------|
| Output Created | | 20-OCT-2020 15:59:24 |
| Comments | | |
| Input | Active Dataset | DataSet1 |
| | Filter | <none> |
| | Weight | <none> |
| | Split File | <none> |
| | N of Rows in Working Data File | 856 |

Syntax

```
matrix.  
get dat/file = */variables =  
Financial_Performance Pro_Env_Strgy  
Eco_Innovation /names = vnames/missing =  
9999.  
compute ninit = nrow(dat).  
get dat/file = */variables =  
Financial_Performance Pro_Env_Strgy  
Eco_Innovation /names = vnames/missing =  
omit.  
get tmp/file = */variables =  
Financial_Performance /names =  
yname/missing = omit.  
get tmp2/file = */variables = Pro_Env_Strgy  
/names = xname/missing = omit.  
get tmp/file = */variables = Eco_Innovation  
/names = mnames/missing = omit.  
get tmp/file = */variables = w999999t z999999t  
v999999t q999999t.  
compute wname=tmp(1,1).  
do if (wname = ' ').  
compute wname = 'xxx'.  
end if.  
compute zname=tmp(1,2).  
do if (zname = ' ').  
compute zname = 'xxx'.  
end if.  
compute vname=tmp(1,3).  
do if (vname = ' ').  
compute vname = 'xxx'.  
end if.  
compute qname=tmp(1,4).  
do if (qname = ' ').  
compute qname = 'xxx'.  
end if.  
compute n = nrow(dat).  
compute p0=-.322232431088.  
compute p1 = -1.  
compute p2 = -.342242088547.  
compute p3 = -.0204231210245.
```

```

compute p4 = -.0000453642210148.
compute q0 = .0993484626060.
compute q1 = .588581570495.
compute q2 = .531103462366.
compute q3 = .103537752850.
compute q4 = .0038560700634.
compute badend = 0.
compute priorlo = -9999999.
compute priorhi = 9999999.
compute criterr = 0.
compute cluster = 0.
compute clsdmy = 0.
compute jndich = 0.
compute wvdich=0.
compute zqdich=0.
compute mod74dic=0.
compute warnrep=0.
compute contrast = trunc( 0 ).
do if (contrast > 2 or contrast < 0).
compute contrast = 0.
end if.
compute booterr = 0.
compute effsize = ( 0 = 1).
compute note = make(10,1,0).
compute notes = 1.
compute iterr = 0.
compute clsmtn = 0.
compute quantile = ( 0 = 1).
compute mmodval= 999.
compute jn = ( 0 = 1).
compute covcoeff=( 0 =1).
compute center = ( 0 = 1).
compute detail = ( 1 = 1).
compute coeffci = ( 1 = 1).
compute conf = 95.
compute longname = ( 1 =1).
compute bconoff=( 0 <> 1).
compute covmy = trunc( 0 ).
do if (covmy < 0 or covmy > 2).
compute covmy = 0.

```

```

end if.
do if (trunc(conf) >= 100 or (trunc(conf) <=
50)).
compute conf = 95.
compute note(notes,1) = 1.
compute notes = notes + 1.
end if.
do if (n < ninit).
compute nmiss = ninit-n.
compute note(notes,1) = 11.
compute notes = notes + 1.
end if.
compute errs = 0.
compute quantd = {0,0,0,0,0,0}.
compute quantc = {0,0,0,0,0,0}.
compute mcheck = 0.
compute ttt = 0.
compute plot = trunc( 0 ).
compute runerrs = make(50,1,0).
compute model = trunc( 4 ).
compute normal = 1.
compute ws= ( 0 =1).
compute olsdichm= ( 0 =1).
compute olsdichy= ( 0 =1).
compute mcx=trunc( 0 ).
compute mcm=trunc( 0 ).
do if (ws = 1).
do if (effsize = 1).
compute note(notes,1) = 19.
compute notes = notes + 1.
compute effsize=0.
end if.
do if (normal=1).
compute note(notes,1) = 16.
compute notes = notes + 1.
compute normal=0.
end if.
end if.
do if (model <> 4).
compute ws = 0.

```

```

end if.
do if (model > 0 and model < 4 and covmy <>
0).
compute covmy=0.
compute note(notes,1) = 33.
compute notes = notes + 1.
end if.
print/space=0.
do if (jn = 1 and model <> 1 and model <> 3).
compute note(notes,1) = 7.
compute notes = notes + 1.
end if.
do if ((model > 76) or (model < 1)).
compute model = 77.
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 19.
end if.
compute itprob=0.
do if (mcx < 0).
compute mcx=0.
end if.
do if (mcx > 4).
compute mcx=0.
end if.
do if (mcm < 0).
compute mcm=0.
end if.
do if (mcm > 4).
compute mcm=0.
end if.
do if (mcx > 0 and mcm > 0).
compute errs=errs+1.
compute runerrs(errs,1)=34.
compute criterr=1.
end if.
do if ((model <> 1 and model <> 4) and (mcx >
0 or mcm > 0)).
compute errs=errs+1.
compute runerrs(errs,1)=35.

```

```
compute criterr=1.  
end if.  
do if ((model = 4) and (mcm > 0)).  
compute errs=errs+1.  
compute runerrs(errs,1)=41.  
compute criterr=1.  
end if.  
do if ((mcx > 0) and jn <> 0).  
compute jn=0.  
compute note(notes,1) = 23.  
compute notes = notes + 1.  
end if.  
do if (mmodval <> 999 and mcm > 0).  
compute note(notes,1) = 24.  
compute notes = notes + 1.  
end if.  
compute toteff = 0.  
compute toteff = (( 0 = 1)*( 4 = 4 or 4 = 6)).  
compute varorder = 2.  
compute hc3 = ( 0 <> 0).  
compute cname = 'xxx'.  
compute centvar = {'xxx'}.  
compute nmeds = ncol(mnames).  
do if (longname=0).
```

compute criterr = 1.

compute errs = errs+1.

compute runerrs(errs,1) = 33.

1

2

3

end if.

compute modelm =

```
{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,2;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,3;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,5;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,6;  
1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,7;  
1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,8;  
1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,9;  
1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,3,10;  
1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,5,11;  
1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,0,5,12;  
1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5,13;  
0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,2,14;  
0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,2,15;  
0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,4,16;  
0,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,4,17;  
0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,6,18;  
0,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,6,19;  
0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,6,20;  
1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,7,21;  
1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,7,22;  
1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,23;  
1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,0,0,24; 1,1,1,1
```

,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,25;
1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,0,0,26;
1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,0,0,27;
1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,7,28;
1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,0,7,29;
1,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,30;
1,1,0,1,0,0,1,1,0,1,0,0,0,0,0,0,0,31;
1,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,0,32;
1,1,1,1,0,0,1,1,1,1,0,0,0,0,0,0,0,33;
1,1,1,1,0,0,1,0,0,1,0,0,0,0,0,0,0,34;
1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,35;
1,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,36;
1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,37;
1,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,38;
1,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,39;
1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,40;
1,0,0,1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,41;
1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,42;
1,0,0,1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,43;
1,0,0,1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,44;
1,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,45;
1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,46;
1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,47;
1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,48;
1,1,0,1,1,0,1,1,0,0,0,

0,0,0,0,0,0,0,0,0,49;
1,1,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,50;
1,1,1,1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,51;
1,1,1,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,52;
1,1,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,53;
1,1,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,54;
1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,55;
1,1,1,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,56;
1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,57;
1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,58;
1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,59;
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,60;
1,1,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,61;
1,1,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,62;

```

1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,0,0,0,0,63;
1,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,64;
1,0,0,1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,65;
1,0,0,1,0,0,0,0,0,1,0,0,1,0,0,0,0,0,0,66;
1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,67;
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,68;
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,69;
1,0,0,1,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,70;
1,0,0,1,0,0,1,0,0,1,0,0,1,1,1,0,0,0,0,71;
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,1,1,0,0,72;
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,1,1,0,0

,73; 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,74;
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,75;
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,1,0,0,0,76;
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,77}.
compute wm = modelm(model, 1).
compute zm = modelm(model, 2).
compute wzm = modelm(model, 3).
compute vy = modelm(model, 4).
compute qy = modelm(model, 5).
compute vqy = modelm(model, 6).
compute wy = modelm(model, 7).
compute zy = modelm(model, 8).
compute wzy = modelm(model, 9).
compute vxy = modelm(model, 10).
compute qxy = modelm(model, 11).
compute vqxy = modelm(model, 12).
compute wmy = modelm(model, 13).
compute wvmy = modelm(model, 14).
compute wvxy = modelm(model, 15).
compute zmy = modelm(model,16).
compute wzmy = modelm(model,17).
compute xmy = modelm(model,18).
compute imm=modelm(model,19).
do if (model = 74).
compute vdich=0.
end if.
do if (ncol(xname) <> 1).
compute errs = errs+1.

```

```

compute runerrs(errs,1) = 20.
compute criterr = 1.
end if.
do if (ncol(mnames) > 10).
compute errs = errs+1.
compute runerrs(errs,1) = 39.
compute criterr = 1.
end if.
do if (ncol(yname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 21.
compute criterr = 1.
end if.
do if (( 999 =0) and ( 999 =0) and (model=3 or
model=2) and (contrast=1)).
compute errs = errs+1.
compute runerrs(errs,1) = 31.
compute criterr = 1.
end if.
compute xlist = (wm or zm or wzm or wy or zy
or wzy or vxy or qxy or vqxy or wvxy or xmy).
compute mlist = (vy or qy or vqy or zmy or wmy
or wzmy or xmy or (model < 4)).
compute bad=0.
do if (criterr = 0).
compute werr = 0.
compute verr = 0.
compute qerr = 0.
compute zerr = 0.
compute yerr = 1.
compute xerr = 1.
compute wlist = (wm or wzm or wy or wzy or
wm or wvmy or wvmy or wvxy or wzmy).
do if (wlist = 1 and wname = 'xxx').
compute werr = 1.
compute wlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.

```

```

do if (wlist = 1 and ((wname = qname) or
(wname = vname) or (wname = zname) or
(wname = xname) or (wname = yname))).
compute werr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 12).
end if.
compute zlist = (zm or wzm or zy or wzy or
zmy or wzmy).
do if (zlist = 1 and zname = 'xxx').
compute zerr = 1.
compute zlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (zlist = 1 and ((zname = qname) or
(zname = vname) or (zname = wname) or
(zname = xname) or (zname = yname))).
compute zerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 13.
end if.
compute qlist = (qy or vqy or qxy or vqxy).
do if (qlist = 1 and qname = 'xxx').
compute qerr = 1.
compute qlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (qlist = 1 and ((qname = zname) or
(qname = vname) or (qname = wname) or
(qname = xname) or (qname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 14.
end if.
compute vlist = (vy or vqy or vxy or vqxy or
wvmy or wvxy).
do if (vlist = 1 and vname = 'xxx').
compute verr = 1.

```

```

compute vlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (vlist = 1 and ((vname = zname) or
(vname = qname) or (vname = wname) or
(vname = xname) or (vname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 15.
end if.
do if (wlist = 0 and wname <> 'xxx').
compute werr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 8.
end if.
do if (zlist = 0 and zname <> 'xxx').
compute zerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 9.
end if.
do if (qlist = 0 and qname <> 'xxx').
compute qerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 10.
end if.
do if (vlist = 0 and vname <> 'xxx').
compute verr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 11.
end if.
do if (hc3 = 1).
compute note(notes,1) = 3.
compute notes = notes+1.
end if.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).

```

```

compute cons = make(n,1,1).
compute temp = (n*sscp(dat))-
(t(csum(dat))*(csum(dat))).
compute temp = temp/(n*(n-1)).
compute temp = csum(((diag(temp))=0)).
compute temp2=1.
do if (ws = 1).
compute temp2 = (n*sscp(tmp2))-
(t(csum(tmp2))*(csum(tmp2))).
compute temp2 = temp2/(n*(n-1)).
end if.
do if ((temp > 0 and ws = 0) or (temp > 0 and
ws = 1 and temp2 > 0)).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 27.
end if.
compute nmeds = ncol(mnames).
compute mcmats=ident(nmeds*2).
compute mccoeff=make(nmeds*2,1,0).
compute sobel = make(nmeds,4,-999).
do if (model = 6 and nmeds > 4).
compute errs = errs+1.
compute runerrs(errs,1)=2.
end if.
do if (model < 4 and nmeds > 1).
compute errs = errs+1.
compute runerrs(errs,1) = 3.
end if.
compute nmods = (model = 74).
compute bad = 0.
compute intcnt = 1.
compute modvals = 0.
compute modvalsd = 0.
compute yintemp = {'int_1', 'int_2', 'int_3',
'int_4', 'int_5', 'int_6', 'int_7', 'int_8', 'int_9',
'int_10', 'int_11', 'int_12', 'int_13', 'int_14',
'int_15'}.
compute yintemp = {yintemp, 'int_16', 'int_17',
'int_18', 'int_19', 'int_20', 'int_21', 'int_22',

```

```

'int_23', 'int_24', 'int_25', 'int_26', 'int_27',
'int_28'}.
compute yintemp = {yintemp, 'int_29', 'int_30',
'int_31', 'int_32', 'int_33', 'int_34', 'int_35',
'int_36', 'int_37', 'int_38', 'int_39', 'int_40'}.
compute yintemp = {yintemp, 'int_41', 'int_42',
'int_43', 'int_44', 'int_45', 'int_46', 'int_47',
'int_48', 'int_49', 'int_50', 'int_51', 'int_52'}.
compute yintemp = {yintemp, 'int_53', 'int_54',
'int_55', 'int_56', 'int_57', 'int_58', 'int_59',
'int_60', 'int_61', 'int_62', 'int_63', 'int_64'}.
compute
cntname={'(C1)';'(C2)';'(C3)';'(C4)';'(C5)';'(C6)';'(
C7)';'(C8)';'(C9)';'(C10)';'(C11)';'(C12)';'(C13)';'(
C14)';'(C15)';'(C16)';'(C17)'}.
compute
cntname={cntname;'(C18)';'(C19)';'(C20)';'(C21
)';'(C22)';'(C23)';'(C24)';'(C25)';'(C26)';'(C27)';'(
C28)';'(C29)';'(C30)';'(C31)'}.
compute
cntname={cntname;'(C32)';'(C33)';'(C34)';'(C35
)';'(C36)';'(C37)';'(C38)';'(C39)';'(C40)';'(C41)';'(
C42)';'(C43)';'(C44)';'(C45)'}.
compute
cntname={cntname;'(C46)';'(C47)';'(C48)';'(C49
)';'(C50)';'(C51)';'(C52)';'(C53)';'(C54)';'(C55)';'(
C56)';'(C57)';'(C58)';'(C59)'}.
compute
cntname={cntname;'(C60)';'(C61)';'(C62)';'(C63
)';'(C64)';'(C65)';'(C66)';'(C67)';'(C68)';'(C69)';'(
C70)';'(C71)';'(C72)';'(C73)'}.
compute
cntname={cntname;'(C74)';'(C75)';'(C76)';'(C77
)';'(C78)';'(C79)';'(C80)';'(C81)';'(C82)';'(C83)';'(
C84)';'(C85)';'(C86)';'(C87)'}.
compute
cntname={cntname;'(C88)';'(C89)';'(C90)';'(C91
)';'(C92)';'(C93)';'(C94)';'(C95)';'(C96)';'(C97)';'(
C98)';'(C99)';'(C100)';'(C101)'}.

```



```

compute
cntname={cntname;'(C102)';'(C103)';'(C104)';'(
C105)'}.
compute apathnam={'a path'; 'a1 path'; 'a2
path'; 'a3 path'; 'a4 path'; 'a5 path'; 'a6 path';
'a7 path'; 'a8 path'; 'a9 path'; 'a10 path'}.
compute bpathnam={'b path'; 'b1 path'; 'b2
path'; 'b3 path'; 'b4path'; 'b5path'; 'b6 path'; 'b7
path'; 'b8 path'; 'b9 path'; 'b10 path'}.
compute modvnm = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute modvnm2 = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute mlab = { ' M1 ='; ' M2 ='; ' M3 ='; '
M4 ='; ' M5 ='; ' M6 ='; ' M7 ='; ' M8 ='; '
M9 ='; ' M10 ='}.
compute m = make(n,nmeds,1).
compute mmat = make(16,nmeds,0).
compute ymat = make(8,nmeds,0).
compute deco = make(10,1,0).
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
compute iterate = abs(trunc( 10000 )).
compute converge = abs( 0.00000001 ).
compute boot = abs(trunc( 5000 )).
do if (ws=1 and 0 > 0).
compute note(notes,1)=17.
compute notes=notes+1.
end if.
compute mc=(abs(trunc( 0 ))*(1-ws)).
do if (mc > 0 and model > 5).
do if (boot = 0).
compute boot=mc.
end if.
compute mc=0.
compute note(notes,1)=12.
compute notes = notes+1.
end if.

```

```

compute adjust = 0.
do if (mc > 0 and (model > 3 and model < 6)).
compute boot=0.
compute bconoff=0.
end if.
do if (boot > 0 and mc > 0).
compute mc = 0.
end if.
compute savboot = 0.
compute saveboot = ( 0 = 1).
do if (ws=1 and saveboot=1).
compute saveboot=0.
compute note(notes,1) = 20.
compute notes = notes + 1.
compute normal=0.
end if.
do if (saveboot =1 and boot > 0 and model >
3).
compute savboot = 1.
end if.
do if (boot <> 0 or mc <> 0).
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
loop.
compute cilow = rnd(bootsz*(1-(conf/100))/2).
compute cihigh =
trunc((bootsz*(conf/100)+(bootsz*(1-
(conf/100))/2)))+1.
do if (cilow < 1 or cihigh > bootsz).
compute
bootsz=trunc((bootsz+1000)/1000)*1000.
compute adjust = 1.
end if.
end loop if (cilow gt 0 and cihigh le bootsz).
compute boot=bootsz.
do if (mc > 0).
compute mc = bootsz.
end if.

```

```

do if (boot > 0 and mc > 0 and (model > 3 and
model < 6)).
compute boot=0.
end if.
do if (adjust = 1 and boot > 0).
compute note(notes,1)=6.
compute notes = notes+1.
end if.
do if (adjust = 1 and mc > 0).
compute note(notes,1)=13.
compute notes = notes+1.
end if.
end if.
do if (model = 6 and nmeds > 1).
compute mmpaths =
make((nmeds+2),(nmeds+2),0).
do if (nmeds = 2).
compute indboot = make(boot+1, 3, 999).
else if (nmeds = 3).
compute indboot = make(boot+1, 7, 999).
else if (nmeds = 4).
compute indboot = make(boot+1, 15, 999).
end if.
compute indlbl = {'Total:','Ind1 :','Ind2 :','Ind3
:','Ind4 :','Ind5 :','Ind6 :','Ind7 :','Ind8 :','Ind9
:','Ind10:','Ind11:','Ind12:','Ind13:','Ind14:','
Ind15:'}.
compute indlbl2 = {'Ind1','Ind2','Ind3','Ind4';
'Ind5','Ind6','Ind7','Ind8','Ind9','Ind10';
'Ind11','Ind12','Ind13','Ind14','Ind15'}.
compute indces = make(boot+1, 4, 999).
end if.
do if (model < 4).
compute boot = 0.
compute cmat = make(10,1,0).
compute zmat = make(10,1,0).
end if.
compute nvarch = make(1,ncol(dat),0).
compute wmatch = 0.
compute zmatch = 0.

```

```

compute vmatch = 0.
compute qmatch = 0.
compute mmatch=0.
compute minprobe=0.
compute maxprobe=0.
loop i = 1 to ncol(vnames).
do if (vnames(:,i)=yname).
compute y = dat(:,i).
compute nvarch(1,i)=1.
compute yerr = 0.
do if ((yname = xname) or (yname = wname) or
(yname = zname) or (yname = vname) or
(yname = qname)).
compute errs = errs+1.
compute runerrs(errs,1)=17.
end if.
end if.
do if (vnames(:,i)=xname).
compute x = dat(:,i).
compute nvarch(1,i)=1.
compute xdich = 1.
compute xerr=0.
loop jj = 1 to n.
do if ((x(jj,1) <> cmin(x)) and (x(jj,1) <>
cmax(x))).
compute xdich = 0.
break.
end if.
end loop.
compute xmean = csum(x)/n.
do if (center = 1 and (model < 4 or xlist > 0)
and mcx = 0).
compute meanvec = make(n,1, xmean).
compute x = x-meanvec.
compute centvar = {centvar, xname}.
end if.
compute xmean = csum(x)/n.
compute tmp = x-(cons*xmean).
compute xsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
do if (xdich = 0).

```

```

compute quantc(1,6) = 1.
compute matx = {xmean-xsd; xmean;
xmean+xsd}.
do if ( 999 = 999 and quantile = 0).
do if ((matx(1,1) < cmin(x)) and model = 74).
compute matx(1,1)=cmin(x).
compute minprobe=1.
end if.
do if ((matx(3,1) > cmax(x)) and model = 74).
compute matx(3,1)=cmax(x).
compute maxprobe=1.
end if.
end if.
do if (quantile = 1).
compute quantd(1,6) = 1.
compute quantc(1,6) = 0.
compute tmp = x.
compute tmp(GRADE(x)) = x.
compute matx =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (xdich = 1).
compute matx = {cmin(x); cmax(x)}.
do if (model=74).
compute matx=cmin(x).
compute mod74dic=1.
end if.
end if.
do if ( 999 <> 999).
compute matx = 999.
compute quantd(1,6) = 0.
compute quantc(1,6) = 0.
end if.
end if.
do if (werr = 0 and wlist = 1).
do if (vnames(:,i)=wname).
compute werr = 0.

```

```

compute wmatch = 1.
compute w = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(w)/n).
compute w = w-meanvec.
compute centvar = {centvar, wname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute wmean = csum(w)/n.
compute tmp = w-(cons*wmean).
compute wsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute wdich = 1.
loop jj = 1 to n.
do if ((w(jj,1) <> cmin(w)) and (w(jj,1) <>
cmax(w))).
compute wdich = 0.
break.
end if.
end loop.
do if (model = 3).
compute jndich=wdich.
compute jnmin=cmin(w).
compute jnmax=cmax(w).
end if.
do if (wdich = 0).
compute matw = {wmean-wsd; wmean;
wmean+wsd}.
do if ( 999 = 999 and quantile = 0).
do if (matw(1,1) < cmin(w)).
compute matw(1,1)=cmin(w).
compute minprobe=1.
end if.
do if (matw(3,1) > cmax(w)).
compute matw(3,1)=cmax(w).
compute maxprobe=1.
end if.
end if.
compute quantc(1,1) = 1.
do if (quantile = 1).

```

```

compute quantd(1,1) = 1.
compute quantc(1,1) = 0.
compute tmp = w.
compute tmp(GRADE(w)) = w.
compute matw =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (wdich = 1).
compute matw = {cmin(w); cmax(w)}.
compute wvdich=1.
compute cmaxw=cmax(w).
compute cminw=cmin(w).
end if.
do if ( 999 <> 999).
compute matw = 999.
compute quantd(1,1) = 0.
compute quantc(1,1) = 0.
end if.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm(1,1)=wname.
compute modmatp(1,1) = 1.
end if.
end if.
do if (zerr = 0 and zlist = 1).
do if (vnames(:,i)=zname).
compute zerr = 0.
compute zmatch = 1.
compute z = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(z)/n).
compute z = z-meanvec.
compute centvar = {centvar, zname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute zmean = csum(z)/n.

```

```

compute tmp = z-(cons*zmean).
compute zsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute zdich = 1.
loop jj = 1 to n.
do if ((z(jj,1) <> cmin(z)) and (z(jj,1) <>
cmax(z))).
compute zdich = 0.
break.
end if.
end loop.
do if (zdich = 0).
compute matz = {zmean-zsd; zmean;
zmean+zsd}.
do if ( 999 = 999 and quantile = 0).
do if (matz(1,1) < cmin(z)).
compute matz(1,1)=cmin(z).
compute minprobe=1.
end if.
do if (matz(3,1) > cmax(z)).
compute matz(3,1)=cmax(z).
compute maxprobe=1.
end if.
end if.
compute quantc(1,2) = 1.
do if (quantile = 1).
compute quantd(1,2) = 1.
compute quantc(1,2) = 0.
compute tmp = z.
compute tmp(GRADE(z)) = z.
compute matz = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (zdich = 1).
compute matz = {cmin(z); cmax(z)}.
compute cmaxz=cmax(z).
compute cminz=cmin(z).
compute zqdich=1.
end if.

```



```

do if ( 999 <> 999).
compute matz = 999.
compute quantd(1,2) = 0.
compute quantc(1,2) = 0.
end if.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm(1,2)=zname.
compute modmatp(1,2) = 1.
end if.
end if.
do if (verr = 0 and vlist = 1).
do if (vnames(:,i)=vname).
compute verr = 0.
compute vmatch = 1.
compute v = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(v)/n).
compute v = v-meanvec.
compute centvar = {centvar, vname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute vmean = csum(v)/n.
compute tmp = v-(cons*vmean).
compute vsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute vdich = 1.
loop jj = 1 to n.
do if ((v(jj,1) <> cmin(v)) and (v(jj,1) <>
cmax(v))).
compute vdich = 0.
break.
end if.
end loop.
do if (vdich = 0).
compute matv = {vmean-vsd; vmean;
vmean+vsd}.
do if ( 999 = 999 and quantile = 0).
do if (matv(1,1) < cmin(v)).
compute matv(1,1)=cmin(v).

```

```

compute minprobe=1.
end if.
do if (matv(3,1) > cmax(v)).
compute matv(3,1)=cmax(v).
compute maxprobe=1.
end if.
end if.
compute quantc(1,3) = 1.
do if (quantile = 1).
compute quantd(1,3) = 1.
compute quantc(1,3) = 0.
compute tmp = v.
compute tmp(GRADE(v)) = v.
compute matv = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (vdich = 1).
compute matv = {cmin(v); cmax(v)}.
compute wvdich=1.
compute cmaxv=cmax(v).
compute cminv=cmin(v).
end if.
do if ( 999 <> 999).
compute matv = 999.
compute quantd(1,3) = 0.
compute quantc(1,3) = 0.
end if.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm(1,3)=vname.
compute modmatp(1,3) = 1.
end if.
end if.
do if (qerr = 0 and qlist = 1).
do if (vnames(:,i)=qname).
compute qerr = 0.
compute qmatch = 1.
compute q = dat(:,i).

```

```

do if (center = 1).
compute meanvec = make(n,1,csum(q)/n).
compute q = q-meanvec.
compute centvar = {centvar, qname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute qmean = csum(q)/n.
compute tmp = q-(cons*qmean).
compute qsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute qdich = 1.
loop jj = 1 to n.
do if ((q(jj,1) <> cmin(q)) and (q(jj,1) <>
cmax(q))).
compute qdich = 0.
break.
end if.
end loop.
do if (qdich = 0).
compute matq = {qmean-qsd; qmean;
qmean+qsd}.
do if ( 999 = 999 and quantile = 0).
do if (matq(1,1) < cmin(q)).
compute matq(1,1)=cmin(q).
compute minprobe=1.
end if.
do if (matq(3,1) > cmax(q)).
compute matq(3,1)=cmax(q).
compute maxprobe=1.
end if.
end if.
compute quantc(1,4) = 1.
do if (quantile = 1).
compute quantd(1,4) = 1.
compute quantc(1,4) = 0.
compute tmp = q.
compute tmp(GRADE(q)) = q.
compute matq = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.

```

```

end if.
end if.
do if (qdich = 1).
compute matq = {cmin(q); cmax(q)}.
compute cmaxq=cmax(q).
compute cminq=cmin(q).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matq = 999.
compute quantd(1,4) = 0.
compute quantc(1,4) = 0.
end if.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm(1,4)=qname.
compute modmatp(1,4) = 1.
end if.
end if.
do if (vnames(:,i)=cname).
compute cld = dat(:,i).
compute cvname = vnames(:,i).
compute nvarch(1,i)=1.
compute clsmtch = 1.
end if.
loop j = 1 to ncol(mnames).
do if (vnames(:,i)=mnames(1,j)).
compute mmatch = mmatch + 1.
compute m(:,j)=dat(:,i).
do if (center = 1 and nvarch(1,i) = 0 and mlist >
0 and mcm = 0).
compute meanvec = make(n,1,csum(m(:,j))/n).
compute m(:,j) = m(:,j)-meanvec.
compute centvar = {centvar, mnames(1,j)}.
end if.
compute nvarch(1,i)=1.
compute dichm = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).

```

```

compute dichm = 0.
break.
end if.
end loop.
do if ((dichm = 1) and (olsdichm = 1)).
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichm = 1 and model > 3 and mcheck = 0
and olddichm = 0).
compute errs= errs+1.
compute runerrs(errs,1) = 1.
compute mcheck = 1.
end if.
do if ((model <= 3) and (ncol(mnames)=1)).
compute nmods = nmods + 1.
compute mmean = csum(m(:,j))/n.
compute tmp = m(:,j)-(cons*mmean).
compute msd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute mdich = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute mdich = 0.
break.
end if.
end loop.
do if (model = 1).
compute jndich=mdich.
compute jnmin=cmin(m(:,j)).
compute jnmax=cmax(m(:,j)).
end if.
do if (mdich = 0).
compute matm = {mmean-msd; mmean;
mmean+msd}.
do if ( 999 = 999 and quantile = 0).
do if (matm(1,1) < cmin(m(:,j))).
compute matm(1,1)=cmin(m(:,j)).
compute minprobe=1.
end if.

```

```

do if (matm(3,1) > cmax(m(:,j))).
compute matm(3,1)=cmax(m(:,j)).
compute maxprobe=1.
end if.
end if.
compute quantc(1,5) = 1.
do if (quantile = 1).
compute quantd(1,5) = 1.
compute quantc(1,5) = 0.
compute tmp = m(:,j).
compute tmp(GRADE(m(:,j))) = m(:,j).
compute matm = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (mdich = 1).
compute matm = {cmin(m); cmax(m)}.
end if.
do if ( 999 <> 999).
compute matm = 999.
compute quantd(1,5) = 0.
compute quantc(1,5) = 0.
end if.
compute modmatv(1,5)=nrow(matm).
compute modmat((1:nrow(matm)), 5) = matm.
compute modvnm(1,5)=mnames(1,j).
compute modmatp(1,5) = 1.
end if.
end if.
end loop.
end loop.
do if (minprobe=1).
compute note(notes,1) = 14.
compute notes = notes + 1.
end if.
do if (maxprobe=1).
compute note(notes,1) = 15.
compute notes = notes + 1.
end if.

```

```

do if (cname <> 'xxx' and clsmatch = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 23.
end if.
do if (cname <> 'xxx').
do if ((cname = zname) or (cname = vname)
or (cname = wname) or (cname = xname) or
(cname = yname) or (cname = qname)).
compute errs = errs+1.
compute runerrs(errs,1) = 24.
end if.
end if.
do if (wlist = 1 and werr = 0 and wmatch = 0).
compute werr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (zlist = 1 and zerr = 0 and zmatch = 0).
compute zerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (qlist = 1 and qerr = 0 and qmatch = 0).
compute qerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (vlist = 1 and verr = 0 and vmatch = 0).
compute verr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (yerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 16.
end if.
do if (xerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 32.
end if.

```

```

do if (model = 6 and nmeds < 2).
compute errs = errs+1.
compute runerrs(errs,1) = 18.
end if.
do if (mmatch < ncol(mnames)).
compute errs=errs+1.
compute runerrs(errs,1) = 25.
end if.
do if (cname <> 'xxx').
compute cld = design(cld).
compute cluster = ncol(cld).
compute cld = cld(:,2:ncol(cld)).
compute clsdmy = ncol(cld).
do if (clsdmy > 19).
compute errs = errs+1.
compute runerrs(errs,1) = 26.
end if.
do if (mcx > 0 or mcm > 0).
compute errs = errs+1.
compute runerrs(errs,1) = 38.
end if.
end if.
compute dichy = 1.
loop jj = 1 to n.
do if ((y(jj,1) <> cmin(y)) and (y(jj,1) <>
cmax(y))).
compute dichy = 0.
break.
end if.
end loop.
do if ((dichy = 1) and (olsdichy = 1)).
compute dichy=0.
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichy = 1).
do if (toteff=1).
compute toteff=0.
compute note(notes,1) = 31.
compute notes = notes + 1.

```



```

end if.
compute jncrit=xp2*xp2.
do if (ws = 1).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 28.
end if.
end if.
compute ncovs = ncol(dat)-rsum(nvarch).
do if (effsize = 1 and covmy <> 0 and ncovs >
0 and model > 3 and model < 7).
compute note(notes,1) = 22.
compute notes = notes + 1.
compute effsize = 0.
end if.
do if (ws = 1 and ncovs < nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 29.
end if.
do if (ws = 1 and ncovs > nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 30.
end if.
end if.
do if (errs = 0).
do if (rsum(quantd) > 0).
compute note(notes,1) = 4.
compute notes = notes+1.
end if.
do if (rsum(quantc) > 0).
compute note(notes,1) = 5.
compute notes = notes+1.
end if.
do if (ncovs > 0).
compute c = make(n,ncovs,0).
compute cnames = {'x'}.
compute j = 1.
loop i = 1 to ncol(vnames).

```

```

do if (nvarch(1,i)) = 0.
compute c(:,j) = dat(:,i).
compute nvarch(1,i)=1.
compute j=j+1.
compute cnames = {cnames, vnames(:,i)}.
end if.
end loop.
compute cnames = cnames(1,2:ncol(cnames)).
do if (ws = 1).
compute covmean=csum(c)/n.
loop i = 1 to ncovs.
compute meanvec = make(n,1, covmean(1,i)).
compute c(:,i) = c(:,i)-meanvec.
end loop.
compute centvar = {centvar, cnames}.
compute x=c(:,1:ncovs).
do if (ncovs=nmeds).
compute ncovs=0.
end if.
end if.
end if.
compute names = {yname, xname, mnames,
wname, zname, vname, qname}.
do if (ncovs > 0).
compute names = {names, cnames}.
end if.
do if (dichy = 1 and effsize = 1).
compute note(notes,1) = 2.
compute notes = notes+1.
end if.
do if (model > 3 and model < 6).
compute indeff=make(nmeds,1,0).
compute indboot=make(boot+1,nmeds,999).
do if (mc > 0).
compute indboot=make(mc+1,nmeds,999).
end if.
do if (effsize = 1 and dichy = 0).
compute rmeff=make(boot+1,nmeds+1,999).
compute abpseff=make(boot+1,nmeds+1,999).
compute abcseff=make(boot+1,nmeds+1,999).

```

```

compute pmeff=make(boot+1,nmeds+1,999).
compute r245 = make(boot+1,1,999).
compute kappa2 = make(boot+1,1,999).
end if.
end if.
do if (model = 6 and effsize = 1 and dichy = 0).
compute
rmeff=make(boot+1,ncol(indboot),999).
compute
abpseff=make(boot+1,ncol(indboot),999).
compute
abcseff=make(boot+1,ncol(indboot),999).
compute
pmeff=make(boot+1,ncol(indboot),999).
end if.
do if (nmods > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm(1,tmp) = modvnm(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:nmods).
compute modvnm=modvnm(:,1:nmods).
compute modmatv=modmatv(:,1:nmods).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvals =
make((modmatv(1,1)*modprod(1,1)),
nmods,0).
loop i = 1 to nmods.
compute strt = 1.

```

```

compute fnsh=0.
loop if (fnsh < nrow(modvals)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvals(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
do if (model = 74).
compute modvals=matx.
compute modvnm=xname.
end if.
compute vmat = make(8,nrow(modvals),0).
compute vmat(1,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute vmat(5,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute indeff=make(nrow(modvals),1,0).
do if (model <> 5).
compute indboot = make(((boot+1)*nmeds),
nrow(modvals), -99999999).
compute indbootp =
make(boot+1,(nmeds*nmods), -99999999).
do if (imm > 4 and imm < 8).
do if (imm = 7).
compute immmv1=matw.
compute immmv2=matv.
end if.
do if (imm = 5).
compute immmv1=matw.
compute immmv2=matz.
end if.
do if (imm = 6).
compute immmv1=matv.
compute immmv2=matq.
end if.

```

```

compute
indboop1=make((boot+1),(nmeds*(nrow(immmv2))),-9999999).
compute
indboop2=make((boot+1),(nmeds*(nrow(immmv1))),-9999999).
compute onesmv1=make(1,nrow(immmv1),1).
compute onesmv2=make(1,nrow(immmv2),1).
compute
nimvs=nrow(immmv1)+nrow(immmv2).
compute cmmval=0.
compute cmmlbs1={'xx'}.
compute cmmlbs2={'xx'}.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv2}.
compute
cmmlbt=make(nrow(immmv2),1,mnames(1,i)).
compute cmmlbs1={cmmlbs1;cmmlbt}.
end loop.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv1}.
compute
cmmlbt=make(nrow(immmv1),1,mnames(1,i)).
compute cmmlbs2={cmmlbs2;cmmlbt}.
end loop.
compute cmmval=cmmval(2:nrow(cmmval),:).
end if.
end if.
end if.
do if (nmods> 0).
loop i = 1 to ncol(modvals).
do if (modvnm(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm(1,i)=vname).
compute vcol = i.
end if.

```

```

do if (modvnm(1,i)=qname).
compute qcol = i.
end if.
end loop.
end if.
do if (dichy = 1).
compute omx = cmax(y).
compute omn = cmin(y).
compute y = (y = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute data = {cons,y,m,x}.
compute datamed = data.
compute datayed = data.
compute datanm = {'constant'; yname;
t(mnames); xname}.
compute datanmm = {'constant'; yname;
t(mnames); xname}.
compute datanmy = {'constant'; yname;
t(mnames); xname}.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
do if (model < 4 and errs = 0).
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' ', ' '}.
compute datayed = {datayed, x&*m}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(1,i) = 1.
compute vmat(2,i) = modvals(i,1).
end loop.
compute mmat = make(16,nmeds,1).
end if.
do if (model = 2 or model = 3).
compute int1 = x&*w.
compute datayed = {datayed, w, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.

```

```

compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,2).
compute vmat(3,i) = modvals(i,1).
compute vmat(4,i) = modvals(i,1)*modvals(i,2).
end loop.
end if.
do if (model = 3).
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*m.
compute int2 = x&*w&*m.
compute datayed = {datayed, int1, int2}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' X', wname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (model = 4 or model = 5).
compute vmat = make(8,1,1).
end if.
compute yintkey2 = yintkey.
do if (wm = 1).
compute int1 = x&*w.
compute datamed = {datamed, w,int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmm = {datanmm; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,wcol).
end loop.
do if (zm = 1).

```

```

compute int1 = x&*z.
compute datamed = {datamed, z, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(3,i) = modvals(i,zcol).
end loop.
end if.
do if (wzm = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*z.
compute int2 = x&*w&*z.
compute datamed = {datamed, int1, int2}.
loop i = 1 to nrow(modvals).
compute vmat(4,i) =
modvals(i,wcol)*modvals(i,zcol).
end loop.
end if.
end if.
compute mdatacol=ncol(datamed).
compute mintkey = yintkey.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
compute medints = intcnt-1.
do if (vy = 1 or xmy = 1).
compute mp = 1.
do if (xmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,1).

```



```

end loop.
end if.
do if (vy = 1).
compute datayed = {datayed, v}.
compute datanmy = {datanmy; vname}.
compute mmods = 1.
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,vcol).
end loop.
do if (qy = 1).
compute mp = 2.
compute datayed = {datayed,q}.
compute datanmy = {datanmy; qname}.
compute mmods = 2.
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,qcol).
end loop.
end if.
do if (vqy = 1).
compute mp = 3.
compute datayed = {datayed, v&*q}.
compute mmods = 3.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,vcol)*modvals(i,qcol).
end loop.
end if.
end if.
compute mints = make(n,(nmeds*mp),0).
loop i = 0 to (nmeds-1).
do if (i = 0 and vqy = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
vname, ' X', qname, ',', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))&*v.

```

```

compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, '', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (xmy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))&*x.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', xname, '', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (qy = 1).
compute mints(:,((i*mp)+2))=m(:,(i+1))&*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', qname, '', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqy = 1).
compute mints(:,((i*mp)+3))=m(:,(i+1))&*v&*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end loop.
compute datayed = {datayed, mints}.
end if.
compute mp = 1.
do if (wvmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,wcol)*modvals(i,vcol).
end loop.

```

```

end if.
compute mints2 = make(n,(nmeds*mp),0).
do if (wmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,wcol).
end loop.
do if (wy = 0 and model > 3).
compute datayed = {datayed, w}.
compute datanmy = {datanmy; wname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wvmy = 1).
compute datayed = {datayed,w&*v}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints2(:,((i*mp)+1))= m(:,(i+1))&*w.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wvmy = 1).
compute mints2(:,((i*mp)+2))=
m(:,(i+1))&*w&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints2}.
end if.
compute mp = 1.
do if (zmy = 1).
loop i = 1 to nrow(modvals).

```

```

compute vmat(6,i) = modvals(i,zcol).
end loop.
do if (wzmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,zcol)&*modvals(i,wcol).
end loop.
end if.
end if.
do if (zmy = 1).
compute mints3 = make(n,(nmeds*mp),0).
do if (zy = 0).
compute datayed = {datayed, z}.
compute datanmy = {datanmy; zname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wzmy = 1 and wzy = 0).
compute datayed = {datayed,w&*z}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints3(:,((i*mp)+1))= m(:,(i+1))&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wzmy = 1).
compute mints3(:,((i*mp)+2))=
m(:,(i+1))&*w&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', zname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.

```

```

end loop.
compute datayed = {datayed, mints3}.
end if.
compute decoc = 1.
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
do if (wy = 1 and model > 3).
compute datayed = {datayed, w, x&*w}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm2(1,1)=wname.
compute modmatp(1,1) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (zy = 1).
compute datayed = {datayed,z,x&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm2(1,2)=zname.
compute modmatp(1,2) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (wzy = 1).
compute datayed = {datayed,w&*z,x&*w&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.

```

```

compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vxy = 1).
compute datayed = {datayed, x&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm2(1,3)=vname.
compute modmatp(1,3) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (qxy = 1).
compute datayed = {datayed, x&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm2(1,4)=qname.
compute modmatp(1,4) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqxy = 1).
compute datayed = {datayed, x&*v&*q}.
compute decoc = decoc+1.

```

```

compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, 'X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end if.
do if (wvxy = 1).
compute datayed = {datayed, x&*w&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, 'X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute modvalsd = 0.
compute ttt = rsum(modmatp).
do if (rsum(modmatp) > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm2(1,tmp) = modvnm2(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:ttt).
compute modvnm2=modvnm2(:,1:ttt).
compute modmatv=modmatv(:,1:ttt).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.

```

```

end loop.
compute modvalsd =
make((modmatv(1,1)*modprod(1,1)), ttt,0).
loop i = 1 to ttt.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvalsd)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvalsd(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
end if.
do if (ttt > 0).
loop i = 1 to ncol(modvalsd).
do if (modvnm2(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm2(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm2(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm2(1,i)=qname).
compute qcol = i.
end if.
end loop.
compute directv = make(nrow(modvalsd),1,1).
do if (wy = 1).
compute directv ={directv, modvalsd(:,wcol)}.
end if.
do if (zy = 1).
compute directv ={directv, modvalsd(:,zcol)}.
end if.
do if (wzy = 1).

```



```

compute directv = {directv,
(modvalsd(:,wcol)*modvalsd(:,zcol))}.
end if.
do if (vxy = 1).
compute directv = {directv, modvalsd(:,vcol)}.
end if.
do if (qxy = 1).
compute directv = {directv, modvalsd(:,qcol)}.
end if.
do if (vqxy = 1).
compute directv = {directv,
(modvalsd(:,vcol)*modvalsd(:,qcol))}.
end if.
do if (wvxy = 1).
compute directv = {directv,
(modvalsd(:,vcol)*modvalsd(:,wcol))}.
end if.
end if.
compute ydatacol = ncol(datayed).
do if (ncovs > 0).
do if (covmy <> 2).
compute datamed = {datamed,c}.
end if.
do if (covmy <> 1).
compute datayed = {datayed,c}.
end if.
do if (covmy = 1 and model = 4 and mcx > 0).
compute datayed = {datayed,c}.
end if.
compute covmeans = csum(c)/n.
end if.
do if (cluster > 0).
compute datamed = {datamed,cld}.
compute datayed = {datayed, cld}.
compute cldmeans = csum(cld)/n.
end if.
compute mst = 3.
compute mnd = mst+nmeds-1.
compute ydatacol = ncol(datayed).
compute mdatacol = ncol(datamed).

```

```

do if (ncovs > 0).
compute datanmy = {datanmy; t(cnames)}.
do if (model > 3).
compute datanmm = {datanmm; t(cnames)}.
end if.
end if.
compute datanmy = {'constant';
datanmy(3:nrow(datanmy),1)}.
do if (model > 3).
compute datanmm = {'constant';
datanmm(3:nrow(datanmm),1)}.
end if.
compute amm = make(2,1,0).
compute abmm = make(2,1,0).
compute mnv = csum(datayed(:,2)/n).
compute mnv = make(n,1,mnv).
compute ssty = csum((datayed(:,2)-mnv)**2).
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
do if (ws=1).
compute stddevm=diag(sigma).
compute
stddevm=sqrt(stddevm(3:(nmeds+2),1)).
end if.
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute r2xy =
(sigma(2,(3+nmeds))/(stddevy*stddevx))**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))**2.
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).
do if (model = 4 and nmeds = 1 and cluster = 0
and ncovs = 0).
compute kappaa = sigma(2,3)*sigma(2,4).

```

```

compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
compute datatm = datamed.
compute dataty = datayed.
compute mdlrms2 = { '4' ; yname; xname}.
compute mdlrms = {'Model ='; ' Y ='; ' X ='
}.
do if (ws = 1).
compute mdlrms2 = { '4' ; yname}.
compute mdlrms = {'Model ='; ' Y ='}.
end if.
loop i = 1 to ncol(mnames).
compute mdlrms2 = {mdlrms2; mnames(1,i)}.
do if (i = 1 and ncol(mnames) = 1).
compute mdlrms = {mdlrms; ' M ='}.
else.
compute mdlrms = {mdlrms; mlab(i,1)}.
end if.
end loop.

```

```

do if (wname <> 'xxx').
compute mdlrms2 = {mdlrms2; wname}.
compute mdlrms = {mdlrms; '  W = '}.
end if.

do if (zname <> 'xxx').
compute mdlrms2 = {mdlrms2; zname}.
compute mdlrms = {mdlrms; '  Z = '}.
end if.

do if (vname <> 'xxx').
compute mdlrms2 = {mdlrms2; vname}.
compute mdlrms = {mdlrms; '  V = '}.
end if.

do if (qname <> 'xxx').
compute mdlrms2 = {mdlrms2; qname}.
compute mdlrms = {mdlrms; '  Q = '}.
end if.

do if (jn = 1 and model = 1 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.

do if (jn = 1 and model = 3 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.

compute mdpbe=0.
compute medte2=0.
do if ( 4 =4 and mcx > 0).
compute medte2=1.
end if.

do if (((mcx > 0) or (mcm > 0)) AND model =
1).
compute mdpbe=1.
compute dd={datayed(:,4),datayed(:,3)}.
do if (ncovs > 0).
compute dd={datayed(:,6:(5+ncovs)),dd}.
end if.
compute dd={datayed(:,2),dd}.
end if.

compute yes = ((nrow(modvals)=1) and
(contrast = 1) and (model = 3 or model = 2)).

```

```

compute yes2 = ((nrow(modvals)=1) and
(contrast = 1) and (model = 2)).
print/title = '***** PROCESS Procedure
for SPSS Release 2.16.3 *****'.
print/title = '      Written by Andrew F. Hayes,
Ph.D.      www.afhayes.com'.
print/title = '  Documentation available in
Hayes (2013).
www.guilford.com/p/hayes3'/space=0.
print mdlnms2/title =
*****
*****'/rnames = mdlnms/format =
a8.
do if (ncovs > 0).
print cnames/title = 'Statistical Controls: '/rlabels
= 'CONTROL=' /format a8.
end if.
print n/title = 'Sample size' /format F10.0.
do if (cluster > 0).
print cluster/rnames = cvname/title =
'Clustering Variable and Number of Clusters'.
end if.
do if (model > 3 and ( 'random' <> 'random')).
compute seedt= 'random'.
print seedt/title='Custom' /format=A12/rlabel =
'Seed:'.
end if.
do if (mdpbe=0 and medte2 = 0).
loop bt = 1 to (boot+1).
do if (bt = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef = 0.
do if (bt > 1).
loop.
compute v=trunc(uniform(n,1)*n)+1.
compute datayed = dataty(v,:).
do if (ws=1).

```

```

compute
bmeans=csum(datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed)))/n.
compute cntmat=make(n,nmeds,1).
compute bmeans2=bmeans.
do if (nmeds > 1).
compute bmeans2=mdiag(bmeans).
end if.
compute cntmat=cntmat*bmeans2.
compute datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed))=datayed(:,(ncol(data
yed)-nmeds+1):ncol(datayed))-cntmat.
end if.
compute detcheck=det(t(datayed)*datayed).
compute rk=(detcheck <> 0).
do if (model > 3).
compute datamed = datatm(v,:).
compute detcheck=det(t(datamed)*datamed).
do if (rk=1).
compute rk=(detcheck <> 0).
end if.
end if.
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
compute temp=diag(sigma).
compute bad = bad+(1-rk).
compute false = 1.
end loop if (rk = 1).
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nme
ds)).
do if (model = 4 and nmeds = 1 and ncovs = 0
and cluster = 0).
compute r2xy =
(sigma(2,4)/(stddevy*stddevx))&**2.

```

```

compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))&**2.
compute sstot = sigma(2,2)*(n-1).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
end if.
do if (model > 3).
loop im = 1 to nmeds.
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant';
datanmm((2+nmeds):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
do if (model = 6).
do if (im = 1).

```

```

compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
end if.
do if (im > 1).
compute xm={cons, datamed(:,3:(im+1)),
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant'; datanmm(2:im,1);
datanmm((mnd):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
compute
mmpaths((im+1),(2:im))=t(coeff(2:im,1)).
end if.
end if.
do if (ws = 1).
compute
coeff(1,1)=csum(datamed(:,(2+im)))/n.
end if.
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
do if (bt = 1).
compute resid=datamed(:,(2+im))-xm*coeff.
compute sse =cssq(resid).
compute mse = sse/(n-ncol(xm)).
compute mnv = csum(data(:,(2+im)))/n.
compute mnv = make(n,1,mnv).
compute sstm = csum((data(:,(2+im))-
mnv)&^2).
compute k3 = nrow(coeff).
do if (hc3 = 1).
compute h = xm(:,1).
loop i3=1 to n.
compute h(i3,1)= xm(i3,:)*invXtX*t(xm(i3,:)).
end loop.
loop i3=1 to k3.

```



```

compute xm(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xm(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xm(:,i3) = sqrt(mse)&*xm(:,i3).
end loop.
end if.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute hccov=invXtX*t(xm)*xm*invXtX.
compute mcmats(im,im)=hccov(2,2).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*hccov*lmat)*((t(lmat
)*coeff)))/dfnum.
compute coeff = coeff(1:(nrow(coeff)-
clsdmy),1).
compute mcoeff(im,1)=coeff(2,1).
compute standerr =
sqrt(diag(invXtX*t(xm)*xm*invXtX)).
compute standerr = standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1).
compute standerr(1,1)=stddevm(im,1)/sqrt(n).
end if.
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-
ncol(xm)))).
compute temp=(n-ncol(xm)).
do if (ws=1).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute temp=(n-1).
end if.
compute xd = abs(xp2).
compute temp = (temp* (exp((temp-
(5/6))*((xd/(temp-(2/3)+(.11/temp)))*(xd/(temp-
(2/3)+(.11/temp))))-1)).

```

```

compute temp1 = coeff-
sqrt(abs(temp))*standerr.
compute temp2 =
coeff+sqrt(abs(temp))*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (ws=1).
compute op=op(1,:).
compute xnmn =
apathnam(((nmeds>1)+1+(im-1)),1).
end if.
compute sobel(im,1) = coeff(2,1).
compute sobel(im,2)=standerr(2,1).
compute temp = mnames(1,im).
compute r2full = 1-(sse/sstm).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute summ = {sqrt(r2full), r2full, mse,
Fratio, dfnum, dfden, pfr}.
do if (detail = 1).
print temp/title =
*****
*****'/rlabels = 'Outcome:' /format
= A8.
do if (ws <> 1).
print summ/title = 'Model Summary' /clabels 'R',
'R-sq', 'MSE', 'F', 'df1', 'df2', 'p' /format = F10.4.
end if.
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
print op/title = 'Model' /rnames = xnmn /clabels
'coeff', 'se', 't', 'p', 'LLCI', 'ULCI' /format = F10.4.
do if (covcoeff=1 and ws <> 1).
compute
hccovtmp=hccov(1:nrow(op),1:nrow(op)).
compute cnamestp=t(xnmn).
print hccovtmp/title='Covariance matrix of
regression parameter
estimates' /rnames=xnmn /cnames=cnamestp /f
ormat= F10.4.

```

```

end if.
do if (nmods > 0 and nrow(mintkey) > 1).
print mintkey/title = 'Product terms key:'/format
= A8.
end if.
do if (ws = 1).
compute wsdf=n-1.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
end if.
end if.
compute ymat(1,im) = coeff(2,1).
do if (ws = 1).
compute ymat(1,im)=coeff(1,1).
end if.
do if (wm = 1).
compute ymat(2,im) = coeff(4,1).
do if (zm = 1).
compute ymat(3,im) = coeff(6,1).
do if (wzm = 1).
compute ymat(4, im) = coeff(8,1).
end if.
end if.
end if.
do if (model = 6).
compute mmpaths((im+1),1)=coeff((im+1),1).
end if.
end loop.
end if.
loop totlp = 1 to (1+(toteff*(bt = 1))).
do if (toteff = 1 and totlp =2).
compute xy =
{cons,datayed(:,(3+nmeds):ydatacol)}.
else.
compute xy={cons, datayed(:,3:ydatacol)}.
end if.
do if (dichy = 1).
compute pt2 =
make(nrow(datayed(:,2)),1,(csum(datayed(:,2))
/n)).

```

```

compute LL3 = datayed(:,2)*ln(pt2)+(1-
datayed(:,2))*ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xy),1,0).
compute LL1 = 0.
compute xy22=xy.
loop jjj = 1 to iterate.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.
compute coeff =
bt1+inv(t(xy22)*xy)*t(xy)*(datayed(:,2)-pt1).
compute pt1 = 1/(1+exp(-(xy*coeff))).
compute itprob = csum((pt1 <
.00000000000001) or (pt1 >
.99999999999999)).
do if (itprob = 0).
compute LL = datayed(:,2)*ln(pt1)+(1-
datayed(:,2))*ln(1-pt1).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
break.
end if.
compute bt1 = coeff.
compute LL1 = LL2.
end loop.
do if (jjj >= iterate and iterr = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
compute iterr = 1.
end if.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.
compute covmat = inv(t(xy22)*xy).
release xy22.
do if (totlp <> 2).

```

```

compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
end if.
do if (dichy = 0).
compute invXtX = inv(t(xy)*xy).
compute coeff =invXtX*t(xy)*datayed(:,2).
do if (effsize=1 and model > 3 and model < 7
and (ncovs > 0 or clsdmy > 0)).
compute xysdy =
{cons,datayed(:,(4+nmeds):ydatacol)}.
compute coeffsd
=inv(t(xysdy)*xysdy)*t(xysdy)*datayed(:,2).
compute resid=datayed(:,2)-(xysdy*coeffsd).
compute sdy cov =sqrt(csum(resid&*resid)/(n-
ncol(xysdy))).
compute xvaron=datayed(:,(3+nmeds)).
compute
coeffx2=inv(t(xysdy)*xysdy)*t(xysdy)*xvaron.
compute residx2=xvaron-(xysdy*coeffx2).
compute sdx cov
=sqrt(csum(residx2&*residx2)/(n-ncol(xysdy))).
end if.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4 and bt > 1).
compute resid=datayed(:,2)-xy*coeff.
compute sse = cssq(resid).
compute r2full = 1-(sse/sstot).
end if.
do if (bt =1).
compute resid=data(:,2)-xy*coeff.
compute k3 = nrow(coeff).
compute sse = cssq(resid).
compute mse = sse/(n-ncol(xy)).

```

```

compute
hcnof=coeff&/(sqrt(diag(mse*invXtX))).
do if (hc3 = 1).
compute h = xy(:,1).
loop i3=1 to n.
compute h(i3,1)= xy(i3,:)*invXtX*t(xy(i3,:)).
end loop.
loop i3=1 to k3.
compute xy(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xy(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xy(:,i3) = sqrt(mse)&*xy(:,i3).
end loop.
end if.
compute covmat = (invXtX*t(xy)*xy*invXtX).
end if.
end if.
do if (bt = 1).
do if (model = 2).
compute xy2={cons, datayed(:,3:ydatacol)}.
compute temp = ncol(xy2).
do if (temp > 6).
compute xy3=xy2(:,7:temp)}.
end if.
compute xy2={xy2(:,1:3), xy2(:,5)}.
do if (temp > 6).
compute xy2={xy2, xy3}.
release xy3.
end if.
compute invXtX = inv(t(xy2)*xy2).
compute coeff2 =invXtX*t(xy2)*datayed(:,2).
compute ssem2=cssq(datayed(:,2)-xy2*coeff2).
release xy2.
end if.
compute standerr = sqrt(diag(covmat)).
do if (totlp =1).

```

```

compute
mcmats((nmeds+1):ncol(mcmats),(nmeds+1):n
col(mcmats))=covmat(2:(1+nmeds),2:(1+nmed
s)).
end if.
compute standerr=standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1 and totlp=2).
compute standerr(1,1)=stddevy/sqrt(n).
end if.
compute coeffplt = coeff.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*coeff)))/dfnum).
compute coeff=coeff(1:(nrow(coeff)-clsdmy),1).
do if (totlp=1).
compute
mccoeff((nmeds+1):nrow(mccoeff))=coeff(2:(1+
nmeds),1).
end if.
compute bbbb=coeff(2,1).
do if totp = 1.
compute deco(1,1)=2+nmeds.
compute deco = deco(1:decoc,1).
compute covdirt = make((nrow(covmat)-
clsdmy),(ncol(covmat)-clsdmy),0).
compute covdirt = covmat(deco,:).
compute covdir =
make(nrow(covdirt),ncol(covdirt),0).
compute covdir = covdirt(:,t(deco)).
compute deco=coeff(deco,1).
do if (ttt > 0).
compute sedir =
sqrt(diag(directv*covdir*t(directv))).
compute directv = directv*deco.
end if.

```

```

compute sobel(:,3)=coeff(2:(1+nmeds),1).
compute sobel(:,4) = standerr(2:(1+nmeds),1).
compute sobel2 = sobel*sobel.
do if (varorder <> 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)*sobel2(:,4)+sobel2(:,3)*sobel2(:,2)).
end if.
do if (varorder = 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)*sobel2(:,4)+sobel2(:,3)*sobel2(:,2)+sobel2(:,2)*sobel2(:,4)).
end if.
compute sobel(:,1) = sobel(:,1)*sobel(:,3).
compute sobel(:,3) = sobel(:,1)/sobel(:,2).
compute sobel(:,4) = 2*(1-
cdfnorm(abs(sobel(:,3)))).
end if.
do if (dichy = 0).
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
do if (ws= 1 and totlp=2).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute dfden=n-1.
end if.
compute cnms = {'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute op = {coeff, standerr, tratio, p}.
end if.
do if (dichy = 1).
compute tratio = (coeff&/standerr).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute wald = tratio*tratio.
compute cnms = {'coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute temp=coeff-abs(xp2)*standerr.
compute op = {coeff, standerr, tratio, p, temp}.
compute temp=coeff+abs(xp2)*standerr.
compute op = {op, temp}.
end if.

```



```

do if (detail = 1).
do if (totlp = 2).
print yname/title = '*****'
TOTAL EFFECT MODEL
*****/rlabels =
'Outcome: '/format = A8.
end if.
do if (totlp <> 2).
print yname/title =
*****
*****/rlabels = 'Outcome: '/format
= A8.
end if.
end if.
do if (dichy = 1 and bt = 1 and totp = 1).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis: '/cnames = nmsd/format = F9.2.
end if.
do if (dichy = 0).
compute r2full = 1-(sse/ssty).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute jndf=dfden.
compute wsdf=dfden.
compute xd = abs(xp2).
compute jncrit = (dfden * (exp((dfden-
(5/6))*((xd/(dfden-
(2/3)+(.11/dfden)))*(xd/(dfden-
(2/3)+(.11/dfden)))))-1)).
compute summ = {sqrt(r2full), r2full, mse,
fratio, dfnum, dfden, pfr}.
compute temp1=coeff-sqrt(jncrit)*standerr.
compute temp2=coeff+sqrt(jncrit)*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (detail = 1).
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels =
'R', 'R-sq', 'MSE' 'F', 'df1', 'df2', 'p'/format =
F10.4.

```

```

end if.
end if.
end if.
do if (dichy = 1).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(coeff)-
1)).
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
do if (detail = 1).
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format =
F10.4.
end if.
end if.
do if (totlp = 2).
compute datanmy={'constant';
datanmy((nmeds+2):nrow(datanmy),1)}.
end if.
do if (detail = 1).
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
do if (ws=1).
compute op2=op(1:(1+nmeds),:).
compute datanmy2 = {'c' path'; bpathnam(
(1+(nmeds>1)): ((1+(nmeds>1))+(nmeds-
1)),1)}.
do if (totlp=2).
compute datanmy2={'c path'}.
compute op2=op(1,:).
compute wsdf=n-1.
end if.
print op2/title = 'Model'/rnames =
datanmy2/cnames = cnms/format = F10.4.
print wsdf/title=' '/rlabels = 'df = '/space=0.

```

```

end if.
do if (ws <> 1).
print op/title = 'Model'/rnames =
datanmy/cnames = cnms/format = F10.4.
do if (covcoeff=1).
compute
covmattp=covmat(1:nrow(op),1:nrow(op)).
compute cnamestp=t(datanmy).
print covmattp/title='Covariance matrix of
regression parameter
estimates'/rnames=datanmy/cnames=cnamestp
p/format= F10.4.
end if.
end if.
end if.
do if (ttt = 0 and totlp = 1).
compute deco = op((nmeds+2),:).
do if (ws = 1).
compute deco=op(1,:).
end if.
end if.
do if (ttt = 0 and totlp = 2).
compute decotot = op(2,:).
do if (ws = 1).
compute decotot=op(1,:).
end if.
end if.
do if (nmods > 0 and model > 4 and detail = 1
and nrow(yintkey) > 1)).
print yintkey/title = 'Product terms key: '/format
= A8.
end if.
do if (nmods > 0 and model < 4 and detail = 1).
print yintkey2/title = 'Product terms key: '/format
= A8.
do if ((model = 1 or model = 2) and dichy = 0).
compute temp={((op(4,3)**2)*(1-r2full))/dfden,
op(4,3)**2, 1, dfden, op(4,4)}.
do if (hc3=1).

```

```

compute temp(1,1)=((hcnot(4,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(2,1).
do if (model = 2).
compute temp={temp;((op(6,3)**2)*(1-
r2full))/dfden, op(6,3)**2, 1, dfden, op(6,4)}.
do if (hc3=1).
compute temp(2,1)=((hcnot(6,1)**2)*(1-
r2full))/dfden.
end if.
compute frat2=(dfden*(r2full-(1-
(ssem2/ssty))))/(2*(1-r2full)).
do if (hc3=1).
compute lmat=make(nrow(coeffplt),2,0).
compute lmat(4,1)=1.
compute lmat(6,2)=1.
compute frat2 =
(t(t(lmat)*coeffplt)*inv(t(lmat)*covmat*lmat)*((t(l
mat)*coeffplt))/2).
end if.
compute temp={temp;(r2full-(1-
(ssem2/ssty))),frat2,2,dfden,1-
fcdf(frat2,2,dfden)}.
compute rnms={rnms; yintkey2(3,1),'Both'}.
end if.
print temp/title = 'R-square increase due to
interaction(s):'/rnames=rnms/clabels = 'R2-
chng', 'F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (model = 3 and dichy = 0).
compute temp={(((op(8,3)**2)*(1-r2full))/dfden,
op(8,3)**2, dfden, op(8,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(8,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(5,1).

```

```

print temp/title = 'R-square increase due to
three-way interaction:.'/rnames=rnms/clabels =
'R2-chng', 'F(1,df2)', 'df2', 'p'/format = F10.4.
end if.
end if.
end if.
do if (model = 6 and totlp = 1).
compute
mmpaths(nrow(mmpaths),1)=coeff(nrow(mmpa
ths),1).
compute
mmpaths(nrow(mmpaths),(2:(nmeds+1)))=t(co
eff(2:(nmeds+1),1)).
end if.
do if (totlp = 1).
loop im = 1 to nmeds.
do if (model < 4).
compute ymat(1,im) = coeff(3,1)*(1-yes).
compute ymat(2,im) = coeff(4,1).
compute cmat(1,im) = covmat(3,3).
compute cmat(2,im) = covmat(4,4).
compute cmat(5,im) = covmat(3,4).
compute jnb1=coeff(3,1).
compute jnb3=coeff(4,1).
compute jnsb1=covmat(3,3).
compute jnsb3=covmat(4,4).
compute jnsb1b3=covmat(3,4).
do if (model = 2 or model = 3).
compute ymat(3,im) = coeff(6,1).
compute cmat(3,im) = covmat(6,6).
compute cmat(6,im) = covmat(3,6).
compute cmat(8,im) = covmat(4,6).
end if.
do if (model = 3).
compute ymat(4,im) = coeff(8,1).
compute cmat(4,im) = covmat(8,8).
compute cmat(7,im) = covmat(3,8).
compute cmat(9,im) = covmat(4,8).
compute cmat(10,im) = covmat(6,8).
compute jnb1=coeff(4,1).

```

```

compute jnb3=coeff(8,1).
compute jnsb1=covmat(4,4).
compute jnsb3=covmat(8,8).
compute jnsb1b3=covmat(4,8).
end if.
end if.
do if (model > 3).
compute ymat(5,im) = coeff((1+im),1).
end if.
do if (xmy = 1).
compute ymat(6,im) = coeff((2+nmeds+im),1).
end if.
do if (vy = 1).
compute ymat(6,im) = coeff((3+nmeds+im),1).
end if.
do if (qy = 1 and vy = 1).
compute ymat(6,im) = coeff((5+nmeds+((im-1)*2)),1).
compute ymat(7,im) = coeff((6+nmeds+((im-1)*2)),1).
end if.
do if (vqy = 1).
compute ymat(6,im) = coeff((6+nmeds+((im-1)*3)),1).
compute ymat(7,im) = coeff((7+nmeds+((im-1)*3)),1).
compute ymat(8,im) = coeff((8+nmeds+((im-1)*3)),1).
end if.
do if (wmy = 1).
compute ymat(7,im) = coeff((3+nmeds+im-wy),1).
end if.
do if (wmy = 1 and vy = 1).
compute ymat(7,im) =coeff((4+(nmeds*2)+im-wy),1).
end if.
do if (wmy = 1 and vy = 1 and wvmy = 1).
compute ymat(7,im) =
coeff((6+(nmeds*2)+((im-1)*2)-wy),1).

```

```

compute ymat(8,im) =
coeff((7+(nmeds*2)+((im-1)*2)-wy),1).
end if.
do if (wmy = 1 and zmy = 1).
compute ymat(6,im) = coeff((6-(wzy*3)+(wzm-
1)+(nmeds*2)+((im-1)*2)*wzm)+((im-1)*(1-
wzm))-((zy-wzy)*2),1).
compute ymat(7,im) = coeff((3-
zy+im+nmeds),1).
do if (wzmy = 1).
compute ymat(8,im) = coeff((7-
(wzy*3)+(nmeds*2)+((im-1)*2)),1).
end if.
end if.
do if (nmods > 0 and model <> 5).
loop indlp = 1 to nrow(modvals).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp)).
do if (model > 6).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp))*csum(ym
at(5:8,im)*vmat(5:8,indlp)).
end if.
end loop.
compute indboot((bt+(im-
1)*(boot+1)),:)=t(indeff).
do if (imm = 1 or imm = 3).
compute
indbootp(bt,im)=ymat(2,im)*ymat(5,im).
do if (imm=3).
compute
indbootp(bt,(im+nmeds))=ymat(3,im)*ymat(5,i
m).
do if (zdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxz-cminz).
end if.
end if.
do if (wdich=1).

```

```

compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
end if.
end if.
do if ((imm= 2 or imm = 4) or model = 74).
compute
indbootp(bt,im)=ymat(1,im)*ymat(6,im).
do if (imm=4).
compute
indbootp(bt,(im+nmeds))=ymat(1,im)*ymat(7,i
m).
do if (qdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxq-cminq).
end if.
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
end if.
end if.
do if (imm=7).
compute
indbootp(bt,im)=ymat(2,im)*ymat(6,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(2,im)*y
mat(6,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(6,im)*onesmv1)+(ymat(2,im)*y
mat(6,im)*t(immmv1)).
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).

```



```

compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxv-cminv).
end if.
end if.
do if (imm=5).
compute
indbootp(bt,im)=ymat(4,im)*ymat(5,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(4,im)*y
mat(5,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(3,im)*ymat(5,im)*onesmv1)+(ymat(4,im)*y
mat(5,im)*t(immmv1)).
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).
end if.
do if (zdich=1).

```

```

compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxz-
cminz).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxz-cminz).
end if.
end if.
do if (imm=6).
compute
indbootp(bt,im)=ymat(1,im)*ymat(8,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(1,im)*ymat(6,im)*onesmv2)+(ymat(1,im)*y
mat(8,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(7,im)*onesmv1)+(ymat(1,im)*y
mat(8,im)*t(immmv1)).
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxv-cminv).
end if.
do if (qdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxq-
cminq).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxq-cminq).

```

```

end if.
end if.
do if ((model = 58 or model = 59) and
wvdich=1).
compute
indbootp(bt,im)=(ymat(1,im)*ymat(7,im)*(cmax
w-cminw))+ymat(2,im)*ymat(5,im)*(cmaxw-
cminw))+ymat(2,im)*ymat(7,im)*((cmaxw*cma
xw)-(cminw*cminw))).
end if.
end if.
do if (model = 4 or model = 5).
compute
indboot(bt,im)=csum(ymat(1:4,im)*vmat(1:4,1
))*csum(ymat(5:8,im)*vmat(5:8,1)).
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.000000000000001.
end if.
compute sumind=rsum(indboot(bt,:)).
do if (im=nmeds).
compute
pmeff(bt,2:(im+1))=indboot(bt,:)/(sumind+coeff(
(2+nmeds),1)).
end if.
compute
rmeff(bt,(im+1))=indboot(bt,im)/coeff((2+nmeds
),1).
compute
abpseff(bt,(im+1))=indboot(bt,im)/stddevy.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*stddevx.
do if (ncovs > 0 or clsdmy > 0).
compute
abpseff(bt,(im+1))=indboot(bt,im)/sdyconv.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*sdxconv.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).

```

```

compute r245(bt,1) = r2my-(r2full-r2xy).
compute abmmr = 1.
do if (indboot(bt,im) < 0).
compute abmmr = cmin(abmm).
end if.
do if (indboot(bt,im) > 0).
compute abmmr = cmax(abmm).
end if.
compute kappa2(bt,1)=indboot(bt,im)/abmmr.
compute tmp = indboot(bt,im)/abmmr.
end if.
end if.
end if.
do if (model = 6).
do if (nmeds = 2).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(4,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3).
compute
indboot(bt,3)=mmpaths(3,1)*mmpaths(4,3).
end if.
do if (nmeds = 3).
compute indboot(bt,1) =
mmpaths(2,1)*mmpaths(5,2).
compute indboot(bt,2) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3).
compute indboot(bt,3) =
mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4).
compute indboot(bt,4) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4).
compute indboot(bt,5) =
mmpaths(3,1)*mmpaths(5,3).
compute indboot(bt,6) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4).
compute indboot(bt,7) =
mmpaths(4,1)*mmpaths(5,4).
end if.

```

```

do if (nmeds = 4).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(6,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(6,3).
compute
indboot(bt,3)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(6,4).
compute
indboot(bt,4)=mmpaths(2,1)*mmpaths(5,2)*mmpaths(6,5).
compute
indboot(bt,5)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(6,4).
compute
indboot(bt,6)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3)*mmpaths(6,5).
compute
indboot(bt,7)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,8)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,9)=mmpaths(3,1)*mmpaths(6,3).
compute
indboot(bt,10)=mmpaths(3,1)*mmpaths(4,3)*mmpaths(6,4).
compute indboot(bt,11)
=mmpaths(3,1)*mmpaths(5,3)*mmpaths(6,5).
compute indboot(bt,12) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,13) =
mmpaths(4,1)*mmpaths(6,4).
compute indboot(bt,14) =
mmpaths(4,1)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,15) =
mmpaths(5,1)*mmpaths(6,5).

```

```

end if.
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.00000000000001.
end if.
compute
pmeff(bt,:)=indboot(bt,:)/(rsum(indboot(bt,:))+m
mpaths(nrow(mmpaths),1)).
compute rmeff(bt,:) =
indboot(bt,:)/mmpaths(nrow(mmpaths),1).
compute abpseff(bt,:)=indboot(bt,:)/stddevy.
compute
abcseff(bt,:)=(stddevx*indboot(bt,:))/stddevy.
do if (ncovs > 0 or clsdmy > 0).
compute abpseff(bt,:)=indboot(bt,:)/sdy cov.
compute
abcseff(bt,:)=(sdxcov*indboot(bt,:))/sdy cov.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,:) = r2my-(r2full-r2xy).
end if.
end if.
end if.
end loop.
end if.
end loop.
do if (savboot = 1 and bt > 1).
compute bootstrp((bt-1),:)=bootcoef.
end if.
end loop.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
release datayed, dat, datamed, data, y, x,
datatm, dataty, mints2, cons, mnv, tmp.
do if (mc > 0).

```

```

compute x1 = sqrt(-
2*ln(uniform(mc,nrow(mcmats))))&*cos((2*3.14
159265358979)*uniform(mc,nrow(mcmats))).
compute x1=x1*chol(mcmats).
loop i=1 to nrow(x1).
compute x1(i,:)=x1(i,:)+t(mccoeff).
end loop.
loop i = 1 to nmeds.
compute
indboot(2:nrow(indboot),i)=x1(:,i)&*x1(:,(i+nme
ds)).
end loop.
end if.
do if (ttt = 0 and model > 3).
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
end if.
do if (model <> 74).
do if (dichy = 0).
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 't', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 't', 'p', 'LLCI', 'ULCI'/format =
F10.4.
else.
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 'Z', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 'Z', 'p', 'LLCI', 'ULCI' /format =
F10.4.

```

```

end if.
end if.
end if.
do if (ttt > 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
compute clbs = {modvnm2, 'Effect', 'SE', 't', 'p',
'LLCI', 'ULCI'}.
compute tratio = (directv&/sedir).
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
compute outp = {modvalsd, directv, sedir,
tratio, p}.
do if (dichy = 0).
compute temp1=directv-sqrt(jncrit)*sedir.
compute temp2=directv+sqrt(jncrit)*sedir.
compute outp = {outp, temp1, temp2}.
end if.
do if (dichy = 1).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute temp = directv-abs(xp2)*sedir.
compute outp = {outp, temp}.
compute temp = directv+abs(xp2)*sedir.
compute outp = {outp, temp}.
compute clbs = {modvnm2, 'Effect', 'SE', 'Z', 'p',
'LLCI', 'ULCI'}.
end if.
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
print outp/title = 'Conditional direct effect(s) of
X on Y at values of the moderator(s):'/format =
F10.4 /cnames = clbs.
end if.
do if (nmods > 0 and model <> 5).
do if (model < 4).
print/title =
*****
*****'.
compute zmat(1,1) = 1*(1-yes).
compute cfse = make(nrow(modvals),1,0).

```



```

loop #m = 1 to nrow(modvals).
do if (model = 1).
compute zmat(2,1)=modvals(#m,1)**2.
compute zmat(5,1)=2*modvals(#m,1).
end if.
do if (model = 2 or model = 3).
compute zmat(2,1)=modvals(#m,2)**2.
compute zmat(3,1)=modvals(#m,1)**2.
compute
zmat(4,1)=(modvals(#m,1)**2)*(modvals(#m,2)
**2)*(1-yes2).
compute zmat(5,1)=2*modvals(#m,2)*(1-yes).
compute zmat(6,1)=2*modvals(#m,1)*(1-yes).
compute
zmat(7,1)=2*modvals(#m,1)*modvals(#m,2)*(1-
yes).
compute
zmat(8,1)=2*modvals(#m,1)*modvals(#m,2).
compute
zmat(9,1)=2*modvals(#m,1)*(modvals(#m,2)**
2)*(1-yes2).
compute
zmat(10,1)=2*(modvals(#m,1)**2)*modvals(#m
,2)*(1-yes2).
end if.
compute cfse(#m,1)=sqrt(csum(zmat&*cmat)).
end loop.
end if.
do if (nmods > 0).
compute clbs = {modvnm, 'Effect'}.
loop im = 1 to nmeds.
compute obs = t(indboot(1+(im-1)*(boot+1),:)).
compute outp = {modvals, obs}.
do if (model < 4).
compute tstat = obs&/cfse.
do if (dichy = 0).
compute pval = 2*(1-tcdf(abs(tstat), (n-
ncol(xy))))).
compute temp=obs-sqrt(jncrit)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.

```

```

compute temp=obs+sqrt(jncrit)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 't', 'p', 'LLCI', 'ULCI'}.
compute jnclbs=clbs.
end if.
do if (dichy = 1).
compute pval = 2*(1-cdfnorm(abs(tstat))).
compute temp = obs-abs(xp2)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp = obs+abs(xp2)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute jnclbs=clbs.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indboot((1+(im-
1)*(boot+1)),:).
compute indboot2 = indboot((2+(im-
1)*(boot+1)):(1+(im-1)*(boot+1)+boot),:).
compute mnind = t(csum(indboot2)/boot).
compute stdind=t(sqrt((cssq(indboot2)-
((csum(indboot2)**2)/boot))/(boot-1))).
compute llci=make(1,ncol(indboot2),-999).
compute ulci=make(1,ncol(indboot2),-999).
loop #e = 1 to ncol(indboot2).

.
compute temp = indboot2(:,#e).
compute temp(GRADE( indboot2(:,#e) )) =
indboot2(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.

```

```

compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).

```

```

compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {modvals, obs, stdind, t(llci),
t(ulci)}.
compute clbs = {modvnm, 'Effect', 'Boot SE',
'BootLLCI', 'BootULCI'}.
end if.
compute mtemp = mnames(1, im).
compute rlbs =
make(nrow(modvals),1,mnames(1,im)).
do if (model < 4).
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
do if (yes=0).
print outp/title = 'Conditional effect of X on Y at
values of the moderator(s):'/cnames =
clbs/format = F10.4.
else.
compute outp=outp(:,3:ncol(outp)).
compute clbs=clbs(:,3:ncol(clbs)).
compute clbs(1,1)='Contrast'.
print outp/title = 'Contrast of conditional effects
of X on Y'/cnames = clbs/format = F10.4.
end if.
end if.
do if (model > 5 and mod74dic <> 1).
do if (im = 1).
print/title = 'Conditional indirect effect(s) of X on
Y at values of the moderator(s):'.
end if.
print outp/title = 'Mediator'/rnames =
rlbs/cnames = clbs/format = F10.4.
else if (mod74dic = 1 and model = 74).
do if (im = 1).
print/title = 'Indirect effect(s) of X on Y:'.
compute clbs3=clbs(1,2:ncol(clbs)).
end if.

```

```

compute outp3=outp(1,2:ncol(outp)).
print outp3/title = 'Mediator'/rnames =
rlbs/cnames = clbs3/format = F10.4.
end if.
end loop.
loop i = notes to 1 by -1.
do if (note(i,1) = 4 and yes=0).
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 5 and yes=0).
print/title = 'Values for quantitative moderators
are the mean and plus/minus one SD from
mean.'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 14 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    below the mean was replaced
with the minimum because one SD below the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.
do if (note(i,1) = 15 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    above the mean was replaced
with the maximum because one SD above the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.
end loop.
do if (model = 3 and yes=0).

```

```

compute jnvals=make(nrow(matw),7,0).
compute jnvals(:,1)=matw.
compute jnvals(:,2)=jnb1+jnb3*jnvals(:,1).
compute
jnvals(:,3)=sqrt(jnsb1+2*jnvals(:,1)*jnsb1b3+(jn
vals(:,1)&*jnvals(:,1))*jnsb3).
compute jnvals(:,4)=jnvals(:,2)/jnvals(:,3).
do if (dichy = 0).
compute jnvals(:,5)=2*(1-tcdf(abs(jnvals(:,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(:,5)=2*(1-
cdfnorm(abs(jnvals(:,4)))).
end if.
compute jnvals(:,6)=jnvals(:,2)-
sqrt(jncrit)*jnvals(:,3).
compute
jnvals(:,7)=jnvals(:,2)+sqrt(jncrit)*jnvals(:,3).
compute clbs={clbs(:,1),clbs(:,3:ncol(clbs))}.
do if (coeffci = 0).
compute jnvals =jnvals(:,1:(ncol(jnvals)-2)).
end if.
print jnvals/title = 'Conditional effect of X*M
interaction at values of W:/'cnames
=clbs/format = F10.4.
end if.
do if (jn = 1 and (model = 1 or model = 3) and
jndich = 0 and yes=0).
print/title = '***** JOHNSON-
NEYMAN TECHNIQUE
*****'.
compute ajn =(jncrit*jnsb3)-(jnb3*jnb3).
compute bjn = 2*((jncrit*jnsb1b3)-(jnb1*jnb3)).
compute cjn = (jncrit*jnsb1)-(jnb1*jnb1).
compute radarg = (bjn*bjn)-(4*ajn*cjn).
compute den = 2*ajn.
compute nrts = 0.
do if (radarg >= 0 and den <> 0).
compute x21 = (-bjn+sqrt(radarg))/den.

```

```

compute x22 = (-bjn-sqrt(radarg))/den.
compute roots = 0.
do if (x21 >= jnmin and x21 <= jnmax).
compute nrts = 1.
compute roots = {roots; x21}.
end if.
do if (x22 >= jnmin and x22 <= jnmax).
compute nrts = nrts + 1.
compute roots = {roots; x22}.
end if.
compute roots={roots,make(nrow(roots),2,0)}.
compute modtemp=m.
do if (model=3).
compute modtemp=w.
end if.
do if (nrts > 0).
compute roots = roots(2:nrow(roots),1:3).
compute roots(1,2)=(csum(modtemp <
roots(1,1))/n)*100.
compute roots(1,3)=(csum(modtemp >
roots(1,1))/n)*100.
do if (nrow(roots)=2).
compute roots(2,2)=(csum(modtemp <
roots(2,1))/n)*100.
compute roots(2,3)=(csum(modtemp >
roots(2,1))/n)*100.
end if.
print roots/title = 'Moderator value(s) defining
Johnson-Neyman significance
region(s)'/clabels = 'Value', '% below', '%
above'/format = F10.4.
compute jnvals=make((21+nrts),7,0).
loop i= 0 to 20.
compute jnvals((i+1),1)=jnmin+(i*((jnmax-
jnmin)/20)).
end loop.
loop i = 1 to nrts.
loop j = 2 to nrow(jnvals).
do if ((roots(i,1) > jnvals((j-1),1)) and (roots(i,1)
< jnvals(j,1))).

```

```

compute
jivals((j+1):(21+i),1)=jivals(j:(20+i),1).
compute jivals(j,1)=roots(i,1).
end if.
end loop.
end loop.
loop i = 1 to nrow(jivals).
compute jivals(i,2)=jnb1+jnb3*jivals(i,1).
compute
jivals(i,3)=sqrt(jnsb1+2*jivals(i,1)*jnsb1b3+(jivals(i,1)*jivals(i,1))*jnsb3).
compute jivals(i,4)=jivals(i,2)/jivals(i,3).
do if (dichy = 0).
compute jivals(i,5)=2*(1-tcdf(abs(jivals(i,4)),
jndf)).
end if.
do if (dichy = 1).
compute jivals(i,5)=2*(1-
cdfnorm(abs(jivals(i,4)))).
end if.
compute jivals(i,6)=jivals(i,2)-
sqrt(jncrit)*jivals(i,3).
compute
jivals(i,7)=jivals(i,2)+sqrt(jncrit)*jivals(i,3).
end loop.
do if (model = 1).
print jivals/title = 'Conditional effect of X on Y
at values of the moderator (M)'/cnames
=jnclbs/format = F10.4.
end if.
do if (model = 3).
compute
jnclbs={jnclbs(:,1),jnclbs(:,3:ncol(jnclbs))}.
print jivals/title = 'Conditional effect of X*M on
Y at values of the moderator (W)'/cnames
=jnclbs/format = F10.4.
end if.
end if.
do if (nrts = 0).

```



```

print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
else.
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
end if.
end if.
do if (model < 4 and (plot = 1)).
compute dataplot =
make((nrow(modvals)*nrow(matx)),(ncol(modv
als)+1),0).
compute tmp = 1.
loop i = 1 to nrow(modvals).
loop j = 1 to nrow(matx).
compute dataplot(tmp,:)={matx(j,1),
modvals(i,:)}.
compute tmp=tmp+1.
end loop.
end loop.
compute dataplot = {dataplot,
make(nrow(dataplot),(1+dichy),0)}.
compute dataplo2 = make(nrow(dataplot),1,1).
do if (model = 1).
compute dataplo2 = {dataplo2, dataplot(:,2),
dataplot(:,1), (dataplot(:,1)*dataplot(:,2))}.
end if.
do if (model = 2 or model = 3).
compute dataplo2 = {dataplo2, dataplot(:,3),
dataplot(:,1), (dataplot(:,1)*dataplot(:,3)),
dataplot(:,2), (dataplot(:,1)*dataplot(:,2))}.
do if (model = 3).
compute dataplo2 = {dataplo2,
(dataplot(:,2)*dataplot(:,3)),
(dataplot(:,1)*dataplot(:,2)*dataplot(:,3))}.
end if.
end if.

```

```

loop i = 1 to nrow(dataplot).
compute tmp=dataplo2(i,:).
do if (ncovs > 0).
compute tmp = {tmp, covmeans}.
end if.
do if (cluster > 0).
compute tmp = {tmp, cldmeans}.
end if.
compute dataplot(i,(ncol(dataplot)-
(dichy)))=tmp*coeffplt.
do if (dichy = 1).
compute
dataplot(i,(ncol(dataplot)))=exp(tmp*coeffplt)/(1
+exp(tmp*coeffplt)).
end if.
end loop.
compute clbs = {xname, modvnm, 'yhat'}.
do if (dichy = 1).
compute clbs = {xname, modvnm, 'ln(odds)',
'prob'}.
end if.
print/title =
!*****
*****!

```

```

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (model = 1).
do if (dichy = 0).
print/title = 'DATA LIST FREE/Pro_Env_Strgy
Eco_Innovation Financial_Performance.'
/space=1.
else if (dichy = 1).
print/title = 'DATA LIST FREE/Pro_Env_Strgy
Eco_Innovation logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH Financial_Performance BY
Eco_Innovation.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Financial_Performance BY
Pro_Env_Strgy.' /space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH logodds BY Eco_Innovation.' /space=0.

```

```

print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH prob BY Eco_Innovation.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Pro_Env_Strgy.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Pro_Env_Strgy.' /space=0.
end if.
end if.
end if.
do if (model = 2 or model = 3).
do if (dichy = 0).
print/title = 'DATA LIST FREE/Pro_Env_Strgy
Eco_Innovation Financial_Performance.'
/space=1.
end if.
do if (dichy = 1).
print/title = 'DATA LIST FREE/Pro_Env_Strgy
Eco_Innovation logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH Financial_Performance BY
Eco_Innovation/PANEL ROWVAR=.' /space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Financial_Performance BY
Pro_Env_Strgy/PANEL ROWVAR=.' /space=0.
end if.

```

```

end if.
do if (dichy = 1).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH logodds BY Eco_Innovation/PANEL
ROWVAR=.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH prob BY Eco_Innovation/PANEL
ROWVAR=.' /space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Pro_Env_Strgy/PANEL
ROWVAR=.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Pro_Env_Strgy/PANEL
ROWVAR=.' /space=0.
end if.
end if.
end if.
do if (ncovs > 0).
print/title = '* Estimates are based on setting
covariates to their sample means.'
end if.
end if.
end if.
do if ((imm > 0) or (model = 74 and
mod74dic=0) or ((model = 58 or model = 59)
and wvdich=1))).
do if (imm > 4 and imm < 8).
compute
indbootp={indbootp(:,1:nmeds),indboop1,indbo
op2}.
end if.
compute obsprod = t(indbootp(1,:)).
do if (boot > 0).

```

```

compute ones = make(boot,1,1).
compute estmte=indbootp(1,:).
compute indbootp = indbootp(2:(boot+1),:).
compute mnindp = t(csum(indbootp)/boot).
compute stdindp=t(sqrt((cssq(indbootp)-
((csum(indbootp)**2)/boot))/(boot-1))).
compute llcip=make(1,ncol(indbootp),-999).
compute ulcip=make(1,ncol(indbootp),-999).
loop #e = 1 to ncol(indbootp).

.
compute temp = indbootp(:,#e).
compute temp(GRADE( indbootp(:,#e) )) =
indbootp(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2)))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.

```

```

do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcip(1,#e)=llcit.
compute ulcip(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {obsprod, stdindp, t(llcip),
t(ulcip)}.
compute clbs = {'Effect', 'SE(Boot)', 'BootLLCI',
'BootULCI'}.
do if (model = 8 or model = 12).
print/title='-----'.
compute outpimm=outp(1:nmeds,:).
print/title = 'Indirect effect of highest order
product:.'/space=0.
print outpimm/title = 'Mediator'/cnames =
clbs/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).

```

```

print/title = '***** INDEX OF
MODERATED MEDIATION
*****'

end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'

end if.
compute clbs3 = {'Index', 'SE(Boot)',
'BootLLCI', 'BootULCI'}.
do if (imm < 5).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=outp((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'

compute outpimm=outp(1:nmeds,:).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
compute
outpimm=outp((nmeds+1):nrow(outp),:).

```



```

compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),clbs3}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3(1,1)=modvnm(1,1).
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
end if.
do if (boot = 0).
do if (model = 8 or model = 12).
print/title='-----'.
compute obsimm=obsprod(1:nmeds,:).
print/title = 'Indirect effect of highest order
product:.'/space=0.

```

```

print obsimm/title = 'Mediator'/clabels =
'Effect'/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
do if ((imm > 0 and imm < 5) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=obsprod((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
end loop.
end if.
do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'.
compute outpimm=obsprod(1:nmeds,1).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.

```

```

compute
outpimm=obsprod((nmeds+1):nrow(obsprod),:)
.

compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),'Index'}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3={modvnm(1,1),'Index'}.
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator: '/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
end if.
end if.
end if.
compute conmake=0.
compute concols = 0.
do if ((model > 3 and model < 7) and (contrast
> 0) and nmods = 0 and nmeds > 1 ).
compute concols =
((ncol(indboot)*(ncol(indboot)-1))/2).
compute indcon=make(nrow(indboot),concols,-
999).

```

```

compute conkey = { ' ', ' ', ' ', ' ' }.
compute temp=1.
compute conmake=1.
loop i = 1 to (ncol(indboot)-1).
loop j = (i+1) to (ncol(indboot)).
compute indcon(:,temp)=indboot(:,i)-
indboot(:,j).
do if (contrast = 2).
compute indcon(:,temp)=abs(indboot(:,i))-
abs(indboot(:,j)).
end if.
do if (model <> 6).
compute conkey={conkey; mnames(1,i),
'minus', mnames(1,j)}.
end if.
do if (model = 6).
compute conkey={conkey; indlbl2(i,1), 'minus',
indlbl2(j,1)}.
end if.
compute temp=temp+1.
end loop.
end loop.
end if.
do if (model = 4 or model = 5).
compute clbs = {'Effect'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obs = {csum(obs); obs}.
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute rlbs = {rlbs;
cntname(1:ncol(indcon),1)}.
end if.
compute outp = obs.
compute outp2=outp.
do if (effsize = 1 and dichy = 0).
compute
pmeff(:,1)=rsum(pmeff(:,2:ncol(pmeff))).
compute
rmeff(:,1)=rsum(rmeff(:,2:ncol(rmeff))).

```

```

compute
abpseff(:,1)=rsum(abpseff(:,2:ncol(abpseff))).
compute
abcseff(:,1)=rsum(abcseff(:,2:ncol(abcseff))).
compute eff = {pmeff, rmeff, abpseff, abcseff}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute eff = {eff, r245}.
compute r245obs = {r245(1,1);r245(1,1)}.
compute kappa2ob = kappa2(1,1).
end if.
compute pmobs = t(pmeff(1,1:(nmeds+1))).
compute rmobs = t(rmeff(1,1:(nmeds+1))).
compute psobs = t(abpseff(1,1:(nmeds+1))).
compute csobs = t(abcseff(1,1:(nmeds+1))).
do if (contrast = 0).
compute outp2 = {obs, psobs, csobs, pmobs,
rmobs}.
end if.
do if (contrast > 0).
compute obs2=obs(1:nrow(psobs),:).
compute outp2 = {obs2, psobs, csobs, pmobs,
rmobs}.
end if.
compute clbs = {'ab', 'ab_ps', 'ab_cs', 'ab/c',
'ab/c'}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute outp2 = {outp2, r245obs}.
compute clbs = {clbs, 'R-sq_med'}.
do if (boot = 0 and mc = 0).
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
do if (boot = 0 and mc = 0).
do if (nmeds = 1).
compute outp2 = outp2(2,:).
compute rlbs = rlbs(2,1).

```

```

end if.
do if (model=5).
compute outp2=outp2(:,1:3).
end if.
print outp2/title = 'Indirect effect(s) of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1 and nmeds >
1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0 or mc > 0).
compute indboot = {rsum(indboot), indboot}.
compute bootSZ=boot.
do if (mc > 0).
compute bootSZ=mc.
end if.
compute ones = make(bootSZ,1,1).
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(bootSZ+1),:).
compute mnind = t(csum(indboot)/bootSZ).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/bootSZ))/(bootSZ-1))).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).
.
compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.

```

```

compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.

```

```

compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
do if (effsize = 1 and dichy = 0 and boot > 0).
compute estmte=eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).

.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.

```



```

compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
end if.
end if.
do if (boot > 0 or mc > 0).
compute outp = {obs, stdind, t(llci), t(ulci)}.
do if (nmeds = 1).
compute outp = outp(2,:).
compute rlbs = rlbs(2,1).

```

```

end if.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
do if (mc > 0).
compute clbs = {'Effect', 'MC SE', 'MC LLCI',
'MC ULCI'}.
end if.
print outp/title = 'Indirect effect of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (dichy = 0 and effsize = 1 and boot > 0).
compute outp = {psobs,
stdindf((2*(nmeds+1)+1):(3*(nmeds+1)),1),
t(llcif(1,(2*(nmeds+1)+1):(3*(nmeds+1)))),
t(ulcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))}.
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Partially standardized indirect
effect of X on Y'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute outp = {csobs,
stdindf((3*(nmeds+1)+1):(4*(nmeds+1)),1),
t(llcif(1,(3*(nmeds+1)+1):(4*(nmeds+1)))),
t(ulcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))}.
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Completely standardized
indirect effect of X on Y'/rnames = rlbs/cnames
= clbs/format = F10.4.
compute outp = {pmobs,
stdindf(1:(nmeds+1),1),t(llcif(1,1:(nmeds+1))),
t(ulcif(1,1:(nmeds+1))) }.
do if (model = 4).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.

```

```

print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
compute outp = {rmobs, stdindf(((nmeds+1)+1)
:(2*(nmeds+1)) ,1),
t(llcif(1,((nmeds+1)+1):(2*(nmeds+1))))),
t(ulcif(1,((nmeds+1)+1):(2*(nmeds+1))))}.
do if (nmeds = 1).
compute outp = outp(2,.).
end if.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
do if (nmeds = 1 and cluster = 0 and ncovs = 0
and model = 4).
compute r245obs = r245obs(1,1).
compute outp = {r245obs,
stdindf((4*(nmeds+1)+1):(4*(nmeds+1)+1),1),
t(llcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1))),
t(ulcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1)))}.
print outp/title = 'R-squared mediation effect
size (R-sq_med)'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
end if.
do if (normal = 1).
do if (nmeds = 1).
print sobel/title = 'Normal theory tests for
indirect effect'/clabels = 'Effect', 'se', 'Z',
'p'/format = F10.4.
end if.
do if (nmeds > 1).
compute rlbs2 = rlbs(2:nrow(rlbs),1).
print sobel/title = 'Normal theory tests for
specific indirect effects'/rnames = rlbs2/clabels
= 'Effect', 'se', 'Z', 'p'/format = F10.4.

```

```

end if.
end if.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
do if (model = 6).
compute clbs = {'eff'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obsfsum=csum(obs).
compute obs = {obsfsum; obs}.
compute indlbl = indlbl(1:nrow(obs),1).
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute indlbl = {indlbl;
cntname(1:ncol(indcon),1)}.
end if.
compute obs2=obs.
do if (boot = 0).
do if (dichy = 0 and effsize = 1).
compute clbs = {'eff', 'eff_ps', 'eff_cs', 'eff/c',
'eff/c'}.
do if (ncovs = 0).
compute obs = {obs, obs/stddevy,
(obs*stddevx/stddevy), obs/ctot,
obs/mmpaths(nrow(mmpaths),1)}.
end if.
do if (ncovs > 0 or clsdmy > 0).
compute obs = {obs, obs/sdy cov,
(obs*sdx cov/sdy cov),
obs/(obsfsum+mmpaths(nrow(mmpaths),1)),
obs/mmpaths(nrow(mmpaths),1)}.
end if.
compute obs2=obs.
do if (contrast > 0).

```

```

compute obs2 = obs(1:(nrow(obs)-concols),:).
end if.
end if.
print obs2/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute indboot = {rsum(indboot), indboot}.
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(boot+1),:).
compute mnind = t(csum(indboot)/boot).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/boot))/(boot-1))).
compute temp = nrow(indboot).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).
.
compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).

```

```

compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.

```

```

compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute obs = {obs, stdind, t(llci), t(ulci)}.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
print obs/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (effsize = 1 and dichy = 0).
compute indboot=indboot(:,1:(ncol(indboot)-
concols)).
compute eff = {rsum(abpseff), abpseff,
rsum(abcseff), abcseff, rsum(pmeff), pmef,
rsum(rmeff), rmeff}.
compute effobs = eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).

.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(effobs(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.

```

```

end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.

```



```

end loop.
compute temp2 = stdindf(1:ncol(indboot),1).
compute temp3 = effobs(:,1:ncol(indboot)).
compute temp2low = llcif(1,1:ncol(indboot)).
compute temp2hi = ulcif(1,1:ncol(indboot)).
compute outp = {t(temp3), temp2,t(temp2low),
t(temp2hi)}.
print outp/title = 'Partially standardized indirect
effect of X on Y'/cnames = clbs/rnames =
indlbl/format = F10.4.
compute temp2 =
stdindf((ncol(indboot)+1):(2*ncol(indboot)),1).
compute temp3 =
effobs(:,(ncol(indboot)+1):(2*ncol(indboot))).
compute temp2low =
llcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute temp2hi =
ulcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute outp = {t(temp3), temp2,t(temp2low),
t(temp2hi)}.
print outp/title = 'Completely standardized
indirect effect of X on Y'/cnames = clbs/rnames
= indlbl/format = F10.4.
compute temp2 =
stdindf((2*(ncol(indboot))+1):(3*ncol(indboot)),1
).
compute temp3 =
effobs(:,(2*(ncol(indboot))+1):(3*ncol(indboot)))
.
compute temp2low =
llcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute temp2hi =
ulcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute outp = {t(temp3), temp2,t(temp2low),
t(temp2hi)}.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
compute temp =
eff(:,(3*(ncol(indboot))+1):(4*ncol(indboot))).

```

```

compute temp2 =
stdindf((3*(ncol(indboot))+1):(4*ncol(indboot)),1
).
compute temp3 =
effobs(:,(3*(ncol(indboot))+1):(4*ncol(indboot)))
.
compute templow =
llcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temphi =
ulcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
end if.
end if.
do if (nmeds = 2).
compute effkey = {xname, '->', mnames(1,1), '-
>', yname, ',', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, ',', ' '}.
compute effkey = {indlbl(2:4,1), effkey}.
end if.
do if (nmeds = 3).
compute effkey = {xname, '->', mnames(1,1), '-
>', yname, ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname, '
', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,3), '->', yname, '
', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, ',', ',', ',', ' '}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, '',' ',' ',' '}.
compute effkey = {indbl(2:8,1), effkey}.
end if.
do if (nmeds = 4).
compute effkey = {xname, '->' , mnames(1,1), '-
>', yname, '',' ',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,4), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname, '',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,4), '->', yname, '',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, '',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', mnames(1,4), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, '',' ',' ',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,4), '->', yname, '
',' ',' ',' '}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, ' ', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, ' ', ' ', ' ', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', mnames(1,4), '->', yname, '
', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,4), '->', yname, ' ', ' ', ' ', ' ', ' ', ' '}.
compute effkey = {indbl(2:16,1), effkey}.
end if.
print effkey/title = 'Indirect effect key'/format =
A8.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
end if.
DO IF (medte2=1).
compute
name=t(mdlnms2(2:nrow(mdlnms2),1)).
compute tmpdat=datayed(:,2:ncol(datayed)).
compute dd=tmpdat(:,1).
compute
dd={dd,tmpdat(:,(2+nmeds)),tmpdat(:,2:(1+nme
ds)))}.
compute stratify=0.
do if (ncovs > 0).
compute cname = cnames.
compute
dd={dd,tmpdat(:,(3+nmeds):(2+nmeds+ncovs))
}.
end if.
compute method = 1.
compute samples = boot.

```

```

do if (mc > 0).
compute samples = mc.
compute method=2.
end if.
compute omnibus = 1.
compute booterr = 0.
do if (mcx < 0 or mcx > 4).
compute mcx = 1.
end if.
compute ovals = ncol(design(dd(:,1))).
compute nte = notes.
compute criterr = 0.
compute xskip = 0.
compute dumok = 0.
compute badend = 0.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute priorlo = -9999999.
compute priorhi = 9999999.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute reps = samples.
do if (contrast = 1).
compute note(nte,1) = 26.
compute nte=nte+1.
end if.
do if (effsize = 1).
compute note(nte,1) = 27.
compute nte=nte+1.
end if.
do if (normal = 1).
compute note(nte,1) = 28.
compute nte=nte+1.
end if.
do if (method = 2).
do if (omnibus = 1).
compute note(nte,1) = 25.

```

```

compute nte=nte+1.
end if.
compute samples = 0.
do if (reps < 1000).
compute reps = 1000.
end if.
end if.
do if (method = 1).
compute samples = reps.
end if.
compute temp = ncol(dd).
compute n = nrow(dd).
compute ones = make(n,1,1).
compute temp=dd(:,2).
compute nxd = ncol(temp).
compute nx = nxd.
do if (nx = 1 and mcx > 0).
compute temp = dd.
compute temp(GRADE(dd(:,2)),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,2)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute mnvls = cmin(t(nnvls)).
do if (mnvls < 2).
compute errs = errs + 1.
compute runerrs(errs,1) = 37.
compute criterr = 1.
end if.
do if (nvls > 9).
compute errs = errs+1.
compute runerrs(errs,1) = 36.
compute criterr = 1.
end if.
do if (criterr = 0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,2).
compute temp = 2.
loop i = 2 to n.

```

```

do if (dd(i,2) <> nnvls((temp-1),1)).
compute nnvls(temp,1)=dd(i,2).
compute temp = temp+1.
end if.
end loop.
do if (mcx > 0).
compute x = dummy(:,2:ncol(dummy)).
compute nx = ncol(x).
compute minus1 = make(1,ncol(x),-1).
compute xdes=make((nx+1),3,0).
compute xdes(1,1)=dd(1,2).
compute xdes(1,2)=1.
compute temp = 2.
loop k = 2 to n.
do if (dd(k,2) <> dd((k-1),2)).
compute xdes(temp,2) = k.
compute xdes(temp,1) = dd(k,2).
compute xdes((temp-1),3) = k-1.
compute temp=temp+1.
end if.
end loop.
compute xdes((temp-1),3)=n.
compute xdes = {xdes, (xdes(:,3)-xdes(:,2)+1)}.
do if (stratify <> 0).
compute note(nte,1) = 7.
compute nte=nte+1.
end if.
do if (mcx = 4).
loop k = 1 to n.
do if (rsum(x(k,:)) = 0).
compute x(k,:) = minus1.
end if.
end loop.
end if.
do if (mcx = 2 or mcx = 3).
compute note(nte,1)=4.
compute nte = nte+1.
loop k = 1 to n.
do if (rsum(x(k,:)) > 0).
loop i = 1 to ncol(x).

```

```

do if (x(k,i) = 0).
compute x(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcx = 3).
compute conmat1={-8,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1; 0,0,0,0,0,-
3,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.
loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i, :)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute x(k,:)=conmat1((rsum(x(k,:))+1),:).
end loop.
end if.
end if.
compute xdata = x.
compute xskip = 1.
compute xname = ddd(1,1:nx).
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcx = 4).
compute dummat(1,:) = minus1.
end if.
do if (mcx = 2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.

```



```

do if (mcx = 3).
compute dummat=conmat1.
end if.
compute dummat={nnvls, dummat}.
compute xname2={name(1,2), xname}.
end if.
end if.
end if.
compute nm=nmeds.
compute r2b = make(nm,1,-999).
compute indeff = make((nx*nm),1,-999).
compute cov =
make((nx*nm+nm),(nx*nm+nm),0).
compute mns=make(nrow(cov),1,0).
compute ny = ncol(dd)-nm-nxd-ncovs.
compute y = dd(:,1).
compute ydata = y.
compute yname = name(1,1).
do if (xskip = 0).
compute x = dd(:,2:(1+nxd)).
compute xdata = x.
compute xname = name(1,2:(1+nxd)).
compute indlbs = t(xname).
end if.
compute m = dd(:,(2+nxd):(1+nxd+nm)).
compute mdata = m.
compute mlab = { ' M1 = ' ; ' M2 = ' ; ' M3 = ' ; '
M4 = ' ; ' M5 = ' ; ' M6 = ' ; ' M7 = ' ; ' M8 = ' ; '
M9 = ' ; ' M10 = ' }.
compute mlab = {mlab ; ' M11 = ' ; ' M12 = ' ; '
M13 = ' ; ' M14 = ' ; ' M15 = ' }.
compute mname =
name(1,(2+nxd):(1+nxd+nm)).
do if (ncovs > 0).
compute c =
dd(:,(2+nxd+nm):(1+nxd+nm+ncovs)).
compute x = {x, c}.
compute xdata = x.
compute cdata = c.
compute xname = {xname, cname}.

```

```

end if.
compute nms={yname; t(mname);
t(xname(1,1:(ncol(xname)-ncovs)))}.
do if (dumok = 1).
compute nms={yname; t(mname); name(1,2)}.
end if.
do if (nm < 16).
do if (dumok = 1).
print dummat/title = 'Coding of categorical X
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
compute
boots=make((samples+1),((nx+1)*nm),-999).
do if (errs = 0 and criterr = 0).
loop k = 1 to (samples+1).
do if (k = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef=0.
do if (k > 1).
compute u=trunc(uniform(n,1)*n)+1.
compute xdata = x(u,:).
compute mdata = m(u,:).
compute ydata = y(u,:).
do if (ncovs > 0).
compute cdata = c(u,:).
end if.
end if.
loop i = 1 to nm.
compute start = 1+(i-1)*nx.
compute xmat = {ones, xdata}.
do if (covmy=2).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = mdata(:,i).
compute amat =
inv(t(xmat)*xmat)*t(xmat)*ymat.

```

```

compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*amat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-(((1-r2)*(n-1)/(n-ncol(xmat)))).
compute mse=ssresid/(n-ncol(xmat)).
compute r2cha = r2.
compute adjr2cha = adjr2.
do if (ncovs > 0 and covmy <> 2).
compute xmat = {ones, cdata}.
compute atmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*atmat)**2).
compute r2step1 = (sstotal-ssresid2)/sstotal.
compute r2cha = r2-r2step1.
compute adjr2cha = adjr2- (1-(((1-r2step1)*(n-
1)/(n-ncol(xmat)))).
compute xmat = {ones, xdata}.
end if.
compute bootcoef={bootcoef,t(amat)}.
compute r2b(i,1)=adjr2cha.
do if (k = 1).
.
compute hcinvtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).

```

```

loop i3=1 to k3.
compute xmat(:,i3) = sqrt( mse )&* xmat(:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute tratio = amat&/sqrt(diag(covmat)).
compute rnms = mname(1,i).
print rnms/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.
compute lmat = ident(nrow(amat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*amat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*amat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/'format = F10.4
/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'.
compute op = {amat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*(xp2/(temp-
(2/3)+(.11/temp)))*(xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.

```

```

print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute cov(start:(start+(nx-
1)),start:(start+(nx-
1)))=covmat(2:(ncol(covmat)-(ncovs*(covmy <>
2))),2:(ncol(covmat)-(ncovs*(covmy <> 2)))).
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns(start:(start+(nx-
1)),1)=amat(2:(1+nx)).
end loop.
compute xmat = {ones,mdata,xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (k = 1).
print yname/title =
*****
*****/rlabels = 'Outcome:'/'format
= A8.
do if (ovals = 2).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:'/'cnames = nmsd/'format = F9.2.
end if.
end if.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(ymat)/n)).
compute LL3 = ymat*&ln(pt2)+(1-ymat)*&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).

```

```

compute LL1 = 0.

.

compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= ymat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp&*ln(pt1lp)+(1-ylp)&*ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.

```

```

do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute bootcoef={bootcoef,t(bmat)}.
compute LL1 = LL2.
compute covmat=varb.
end if.
do if (ovals <> 2).
compute bmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*bmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute bootcoef={bootcoef,t(bmat)}.
end if.
do if (k = 1).
do if (ovals = 2).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).

```

```

compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms2/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute
opdirsum=outp((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
end if.
do if (ovals <> 2).

.

compute hcinvXtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,)*hcinvXtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.
end if.
compute hcdat= xmat.

```



```

compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(bmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/(ncol(xmat)-1).
compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat((2+nmeds):(2+nmeds+(nx-1)),:)= lmat2.
compute fratiodr =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/nx.
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute df2=(n-ncol(xmat)).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), df2, pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = bmat&/sqrt(diag(covmat)).
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute op = {bmat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat)))) }.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*(xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.

```

```

print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms2/format =
F10.4.
compute
opdirsum=op((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
compute xmat = {ones,mdata}.
do if (ncovs > 0).
compute xmat = {xmat, cdata}.
end if.
compute btmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*btmat)**2).
compute r2direct = r2-((sstotal-
ssresid2)/sstotal).
compute pfr = 1-fcdf(fratiodr,(nx),(n-1-nx-nm-
ncovs)).
compute r2direct = {r2direct, fratiodr, nx, (n-1-
nx-nm-ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms2/cnames=rnms2/form
at= F10.4.
end if.
end if.
do if (k = 1 and toteff = 1).
print yname/title = '*****
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:.'/format = A8.
compute xmat = {ones, xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.

```

```

do if (ovals = 2).
compute pt2 = make(n,1,(csum(yamat)/n)).
compute LL3 = yamat*&ln(pt2)+(1-yamat)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= yamat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).

```

```

compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute LL1 = LL2.
compute covmat=varb.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(bmat)-
1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms = {'Constant'; t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).
compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.

```

```

print outp/title = 'Logistic Regression
Model'/rnames = rnms/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute optotsum=outp(2:(1+nx),:).
end if.
do if (ovals <> 2).
compute cmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*cmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).

.

compute hcinvtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvtX*t( xmat )* xmat
*hcinvtX
.
compute lmat = ident(nrow(cmat)).

```

```

compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lmat)*cmat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio, (ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/format = F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'.
compute tratio = cmat&/sqrt(diag(covmat)).
compute op = {cmat, sqrt(diag(covmat)), tratio, (2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-(5/6))*((xp2/(temp-(2/3)+(.11/temp)))*xp2/(temp-(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-sqrt(abs(temp))*op(:,2).
compute temp2 = op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't', 'p', 'LLCI', 'ULCI'/rnames = rnms/format = F10.4.
compute optotsum=op(2:(1+nx),:).
compute ssresid2 = ssresid.
compute r2total = r2.
compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat(2:(1+nx),:) = lmat2.
compute fratio3 =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lmat)*cmat)))/nx.
do if (ncovs > 0).
compute xmat = {ones, cdata}.

```

```

compute ctmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*ctmat)**2).
compute r2total = r2-((sstotal-ssresid2)/sstotal).
end if.
compute pfr = 1-fcdf(fratio3,(nx),(n-1-nx-
ncovs)).
compute r2tot2 = {r2total, fratio3, nx, (n-1-nx-
ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns((nrow(mns)-
nm+1):nrow(mns),1)=bmat(2:(1+nm),1).
compute tmp = 1.
loop i = 1 to (nm*nx).
compute
boots(k,i)=mns(i,1)*mns(((nm*nx)+tmp),1).
do if ((i/nx)=trunc(i/nx)).
compute
boots(k,((nm*nx)+tmp))=r2b(tmp,1)*mns(((nm*
nx)+tmp),1).
compute tmp = tmp+1.
end if.
end loop.
do if (savboot = 1 and k > 1).
compute bootstrp((k-1),:)=bootcoef.
end if.
end loop.
do if (method = 1).
compute llci=make(1,ncol(boots),-999).
compute ulci=make(1,ncol(boots),-999).
compute indeff = t(boots(1,1:(nm*nx))).
compute indeff22=t(boots(1,:)).

```

```

compute omni = t(boots(1,((nm*nx)+1) :
ncol(boots))).
do if (nrow(boots) > 1).
compute boots = boots(2:nrow(boots),:).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
loop i = 1 to ncol(boots).

.

compute temp = boots(:,i).
compute temp(GRADE( boots(:,i) )) = boots(:,i).
compute badlo = 0.
compute badhi = 0.
do if ( (indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(indeff22(i,1)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.

```



```

compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,i)=llcit.
compute ulci(1,i)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute llci = t(llci).
compute ulci= t(ulci).
compute llci={llci,ulci}.
end if.
end if.
do if (method = 2).
compute tmp=1.
loop i = 1 to (nm*nx).
compute
indeff(i,1)=mns(i,1)*mns(((nm*nx)+tmp),1).
do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = sqrt(-
2*ln(uniform(reps,nrow(cov))))&*cos((2*3.1415
9265358979)*uniform(reps,nrow(cov))).
compute boots=boots*chol(cov).

```

```

loop i = 1 to ncol(boots).
compute boots(:,i)=boots(:,i)+mns(i,1).
end loop.
compute tmp=1.
loop i = 1 to (nm*nx).
compute
boots(:,i)=boots(:,i)*boots(:,((nm*nx)+tmp)).
compute temp = boots(:,i).
compute temp(GRADE(boots(:,i))) = boots(:,i).
compute boots(:,i) = temp.
do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = boots(:,1:(nm*nx)).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
compute llci =
t(((boots(cilow,:));(boots(cihigh,:)))).
end if.
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
compute rnmssum=rnms(2:(1+nx),1).
do if (ovals <> 2).
print optotsum/title = 'Relative total effects of X
of Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.
print r2tot2/title = 'Omnibus test of total effect of
X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print optotsum/title = 'Relative total effects of X
on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.

```

```

print/title= '=====' /space=0.
end if.
compute
rnmssum=rnms2((2+nmeds):(1+nmeds+nx),1).
do if (ovals <> 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.
print r2direct/title = 'Omnibus test of direct
effect of X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '=====' /space=0.
do if (omnibus = 1 and (method = 1 or method
= 3)).
compute indlbs = {indlbs; 'Omnibus'}.
end if.
loop i = 1 to nm.
compute op = mname(1,i).
print op/title = 'Relative indirect effect(s) of X on
Y through:.'/format = A8/space=0.
do if (method = 2).
compute clbs = {'Effect', 'SE MC', 'MC LLCI',
'MC ULCI'}.
compute op = {indeff, bootse(1:(nm*nx)), llici}.
end if.
do if (method = 1).
compute clbs = {'Effect', 'SE(boot)', 'LLCI',
'ULCI'}.
compute op = indeff.
do if (samples > 0).
compute op = {indeff, bootse(1:(nm*nx),:),
llici(1:(nm*nx),:)}).
end if.
end if.

```

```

compute temp = op((((i-1)*nx)+1):(nx*i),:).
do if ((method = 1) and omnibus = 1).
do if (samples = 0).
compute temp={temp;omni(i,1)}.
end if.
do if (samples > 0).
compute temp = {temp; omni(i,1),
bootse((((nm*nx)+i),:), llci((((nm*nx)+i),:))}.
end if.
end if.
print temp/title = ' '/rnames = indlbs/cnames =
clbs/format = F10.4 /space=0.
print/title = '-----'.
end loop.
end if.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
END IF.
DO IF (mdpbe=1).
compute nm={yname,xname,mnames}.
do if (ncovs > 0).
compute
nm={yname,cnames,xname,mnames}.
end if.
compute tpx=xname.
compute x = dd(:,2:ncol(dd)).
compute tpy = yname.
compute tempy= dd(:,1).
compute n = nrow(dd).
compute nx=1.

```

```

compute criterr=0.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute ddd1 = {'int_1', 'int_2', 'int_3', 'int_4',
'int_5', 'int_6', 'int_7', 'int_8', 'int_9'}.
compute dumok = 0.
compute mcfoc=mcx.
compute mcmmod=mcm.
compute mcloc=(mcfoc>0).
compute con = make(n,1,1).
compute ncovs=ncol(x)-2.
do if (mcfoc > 0 or mcmmod > 0).
compute temp = dd.
compute temp(GRADE(dd(:,(ncol(dd)-
mcloc))),:)= dd.
compute dd = temp.
compute dummy = design(dd(:,ncol(dd)-
mcloc)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute toosmall=rsum(nnvls < 2).
compute mnvls = cmin(t(nnvls)).
do if ((rsum(nnvls < 2)) > 0).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=37.
end if.
do if (nvls > 10).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=36.
end if.
do if (criterr=0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,ncol(dd)-mcloc).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,ncol(dd)-mcloc) <> nnvls((temp-
1),1)).

```

```

compute nnvls(temp,1)=dd(i,ncol(dd)-mcloc).
compute temp = temp+1.
end if.
end loop.
compute dummy = dummy(:,2:ncol(dummy)).
do if (mcfoc=4 or mcmmod=4).
compute minus1=make(1,ncol(dummy),-1).
loop k = 1 to n.
do if (rsum(dummy(k,:)) = 0).
compute dummy(k,:)=minus1.
end if.
end loop.
end if.
do if (mcfoc = 2 or mcfoc = 3) or (mcmmod = 2 or
mcmmod = 3)).
loop k = 1 to n.
do if (rsum(dummy(k,:)) > 0).
loop i = 1 to ncol(dummy).
do if (dummy(k,i) = 0).
compute dummy(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcfoc = 3 or mcmmod=3).
compute conmat1={-8,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1; 0,0,0,0,0,-
3,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.
loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,:)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute
dummy(k,:)=conmat1((rsum(dummy(k,:))+1),:).
end loop.

```

```

end if.
end if.
compute nx = ncol(dummy).
compute xname = ddd(1,1:nx).
compute xdata = dummy.
compute xname = ddd(1,1:nx).
compute xname2={nm(1,(ncol(nm)-mcloc)),
xname}.
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcfoc = 2 or mcmmod=2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcfoc = 3 or mcmmod=3).
compute dummat=conmat1.
end if.
do if (mcfoc = 4 or mcmmod = 4).
compute dummat(1,:)=minus1.
end if.
compute dummat={nnvls, dummat}.
end if.
end if.
do if (criterr=0).
compute y = dd(:,1).
compute ovals = ncol(design(dd(:,1))).
compute itprob = 0.
do if (ovals = 2).
compute omx = cmax(y(:,1)).
compute omn = cmin(y(:,1)).
compute y(:,1) = (y(:,1) = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute sstotal = csum((y-(csum(y)/n))&**2).
compute outv = t(nm(1,1)).

```

```

compute mdtr = nm(1,ncol(dd)).
compute fciv = nm(1,(ncol(dd)-1)).
do if (mcfoc <> 0 or mcmod <> 0).
compute inter=make(n,nx,0).
loop i = 1 to nx.
compute inter(:,i)=dummy(:,i)*dd(:,ncol(dd)-(1-mcloc)).
end loop.
do if (ncovs = 0).
compute x={con,dd(:,ncol(dd)-(1-mcloc)),dummy}.
compute x={x,inter}.
compute nms={nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
do if (ncovs > 0).
compute
x={con,dd(:,2:(1+ncovs)),dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={t(nm(1,2:(1+ncovs)));
nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
end if.
compute xmns = csum(x)/n.
compute focvals = {1,0,0}.
compute highwarn = 0.
compute lowwarn = 0.
do if (ncol(x) > (2*nx+2)).
compute covmns = {1, xmns(1,2:(ncol(x)-1-
(2*nx)))}.
compute focvals = {covmns,0,0}.
end if.
compute dfres = n-ncol(x).
do if (ovals = 2).
compute pt2 = make(n,1,(csum(y)/n)).
compute LL3 = y*ln(pt2)+(1-y)*ln(1-pt2).

```



```

compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(x),1,0).
compute LL1 = 0.

.

compute pt1lp= pt1.
compute xlp= x.
compute ylp= y.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)*&ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.

```

```

compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
else if (ovals <> 2).
compute invXtX=inv(t(x)*x).
compute b = invXtX*t(x)*y.
compute resid = y-(x*b).
compute ssresid=csum(resid&*resid).
compute msresid=ssresid/dfres.
compute varb = msresid*invXtX.
compute
seint=sqrt(varb(nrow(varb),nrow(varb))).
compute k3 = nrow(b).
do if (hc3 = 1).
compute xhc=x.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)= xhc(i3,:)*invXtX*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).
end loop.
compute varb=(invXtX*t(xhc)*xhc*invXtX).
end if.
compute seb = sqrt(diag(varb)).
compute r2 = 1-(ssresid/sstotal).
do if (mcfoc <> 0 or mcmod <> 0).
compute ytp=y.
compute xtp=x(:,1:(ncol(x)-nx)).
compute btp = inv(t(xtp)*xtp)*t(xtp)*ytp.
compute residt = ytp-(xtp*btp).
compute ssresidt=csum(residt&*residt).
compute r2noint = 1-(ssresidt/sstotal).
end if.
compute pr = ncol(x)-1.

```

```

compute lmat = ident(nrow(b)).
compute lmat = lmat(:,2:ncol(lmat)).
compute f =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
pr).
compute pf = 1-fcdf(f,pr,dfres).
compute pf = {sqrt(r2),r2,msresid,f,pr,dfres,pf}.
compute xd=abs(xp2).
end if.
do if (itprob = 0).
compute tstat = b&/seb.
do if (ovals <> 2).
compute p = 2*(1-tcdf(abs(tstat), dfres)).
else if (ovals = 2).
compute p = 2*(1-cdfnorm(abs(tstat))).
end if.
compute outp = {b,seb,tstat,p}.
compute nms = {'constant'; nms; 'interact'}.
do if (ovals <> 2).
compute tval = (dfres* (exp((dfres-
(5/6))*((xd/(dfres-(2/3)+(.11/dfres)))*(xd/(dfres-
(2/3)+(.11/dfres)))))-1)).
compute outp={outp,(b-
sqrt(tval)&*seb),(b+sqrt(tval)&*seb)}.
end if.
compute bb = tval.
do if (dumok = 1).
do if (mcfoc > 0).
print dummat/title = 'Coding of categorical X
variable for analysis: '/cnames = xname2/format
= F5.2.
else if (mcmod > 0).
print dummat/title = 'Coding of categorical M
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
print yname/title =
*****

```

```

*****'/rlabels = 'Outcome:'/format
= A8.
do if (ovals = 2).
compute nmsd = {outv, 'Analysis'}.
print rcd/title = 'Coding of binary Y for
analysis:'/cnames = nmsd/format = F10.4.
compute outp = {outp,(outp(:,1)-
sqrt(tval)&*outp(:,2)),(outp(:,1)+sqrt(tval)&*outp
(:,2))}.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
print outp/title = 'Logistic Regression
Model'/rnames = nms/clabels 'Coeff' 'se' 'Z' 'p'
'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
compute varbtmp=varb.
compute btmp=b.
compute LL2f=LL2.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=x(:,1:(ncol(x)-nx)).
compute bt1 = make(ncol(xlp),1,0).
.
compute pt1lp= pt1lp.

```

```

compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(y-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*ln(pt1lp)+(1-ylp)*ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL2f.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(LLdiff,nx).

```

```

compute rcha={LLdiff,nx,pchi}.
end if.
do if (ovals <> 2).
print pf/title = 'Model Summary'/clabels = 'R' 'R-
sq' 'MSE' 'F' 'df1' 'df2' 'p'/format F10.4.
compute lmat=make(nrow(b),1,0).
compute lmat(nrow(lmat),1)=1.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
1).
compute
rcha=((b(nrow(b),1)/seint)*(b(nrow(b),1)/seint))*
(1-r2)/dfres.
compute rcha = {rcha, fcha, 1, dfres,
outp((pr+1),4)}.
print outp/title = 'Model'/rnames = nms/clabels
'coeff' 'se' 't' 'p' 'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
do if (mcfoc > 0 or mcmod > 0).
compute rcha=r2-r2noint.
compute lmat=make((nrow(b)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
nx).
compute pvalr2c=1-fcdf(fcha,nx,dfres).
compute rcha = {rcha, fcha, nx, dfres,pvalr2c}.
end if.
end if.
do if (mcfoc > 0 or mcmod > 0).
compute intkey = {'a', 'b', 'c', 'd', 'e'}.
loop i = 1 to nx.
compute intkey={intkey; ddd1(1,i), ' : ', ddd(1,i),
' X ', nm(1,ncol(dd)-(1-mcloc))}.

```

```

end loop.
compute intkey=intkey(2:nrow(intkey),:).
print intkey/title='Product terms key: '/format =
a8.
end if.
do if (ovals <> 2).
print rcha/title = 'R-square increase due to
interaction: '/clabels 'R2-chng' 'F' 'df1' 'df2'
'p'/format F10.4.
else if (ovals = 2).
print rcha/title = 'Likelihood ratio test for
interaction: '/clabels 'Chi-sq' 'df' 'p'/format F10.4.
end if.
print/title =
*****
*****!
compute mdvar = x(:,(ncol(x)-1)).
do if (mcfoc = 0 and mcmod = 0).
compute g1 = b((ncol(x)-2),1).
compute g3 = b(ncol(x),1).
compute vg1 = varb((ncol(x)-2),(ncol(x)-2)).
compute vg3 = varb(ncol(x),ncol(x)).
compute covg1g3 = varb((ncol(x)-2), ncol(x)).
end if.
do if (mcfoc > 0 or mcmod > 0).
compute mdvar = x(:,(ncol(x)-(2*nx))).
end if.
compute mdmin = cmin(mdvar).
compute mdmax = cmax(mdvar).
compute fvar = x(:,(ncol(x)-2)).
compute nval = ncol(design(mdvar)).
compute fvmin = cmin(fvar).
compute fvmax = cmax(fvar).
do if (mmodval = 999).
compute mnmd = csum(mdvar)/n.
compute tmp = make(n,1,mnmd).
compute sdmd = sqrt(csum(((mdvar-
tmp)**2))/(n-1)).
compute probeval = {mnmd-sdmd; mnmd;
mnmd+sdmd}.

```

```

do if (probeval(1,1) < mdmin).
compute lowwarn = 1.
compute probeval(1,1)=mdmin.
end if.
do if (probeval(nrow(probeval),1) > mdmax).
compute highwarn = 1.
compute probeval(nrow(probeval),1)=mdmax.
end if.
do if (quantile = 1).
compute tmp = mdvar.
compute tmp(GRADE(mdvar(:,1)),:)= mdvar.
compute mdvar = tmp.
compute
probeval={mdvar(trunc(0.1*n),1);mdvar(trunc(0
.25*n),1);mdvar(trunc(0.5*n),1);mdvar(trunc(0.7
5*n),1);mdvar(trunc(0.9*n),1)}.
end if.
end if.
do if (nval = 2).
compute probeval = make(2,1,0).
compute probeval(1,1)=cmin(mdvar).
loop i = 1 to n.
do if (mdvar(i,1) <> probeval(1,1)).
compute probeval(2,1) = mdvar(i,1).
BREAK.
end if.
end loop.
end if.
do if (mmodval <> 999).
compute probeval = mmodval.
end if.
compute outp = make(nrow(probeval),7,0).
do if (mcfoc > 0 or mcmmod > 0).
compute focvals=make(1,ncol(x)+1,1).
do if (mcfoc > 0).
print/title = 'Conditional effect of X on Y at
values of the moderator:'.
print/title = ' '/space=0.
compute rn2=mdtr.
compute matt=make(nx,6,0).

```



```

end if.
loop jj=1 to nrow(probeval).
do if (mcfoc > 0).
loop ii=1 to nx.
compute g1=b((ncol(x)-(2*nx)+ii),1).
compute g3=b((ncol(x)-nx+ii),1).
compute vg1=varb((ncol(x)-(2*nx)+ii),(ncol(x)-
(2*nx)+ii)).
compute vg3=varb((ncol(x)-nx+ii),(ncol(x)-
nx+ii)).
compute covg1g3=varb((ncol(x)-
(2*nx)+ii),(ncol(x)-nx+ii)).
compute x2 = probeval(jj,1).
compute w1 = g1+g3*x2.
compute varw1 =
vg1+(2*x2*covg1g3)+((x2*x2)*vg3).
compute sew1 = sqrt(varw1).
compute t1 = w1/sew1.
do if (ovals <> 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-tcdf(abs(t1), dfres)).
compute cnms = {'Coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)= {w1,sew1,t1,p,llci,ulci}.
else if (ovals = 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-cdfnorm(abs(t1))).
compute cnms = {'Coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)= {w1,sew1,t1,p,llci,ulci}.
end if.
end loop.
compute rnms=t(xname).
compute mdvalpr=probeval(jj,1).
print mdvalpr/title = 'Moderator
value:.'/rnames=rnn2/format= F10.4 /space=0.

```

```

print matt/title='
'/cnames=cnms/rnames=rnms/format= F10.4
/space=0.
compute xprob=x(:,(ncol(x)-(2*nx)))-mdvalpr.
loop kk = 1 to nx.
compute xprob={xprob,
(xprob(:,1)&*x(:,(ncol(x)-(2*nx)+kk)))}.
end loop.
compute xprob={x(:,1:((ncol(x)-(2*nx))-
1)),xprob}.
do if (ovals <> 2).
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute residc = y-(xprob*bmultc).
compute ssresidc=csum(residc*residc).
compute r2c = r2-(1-(ssresidc/sstotal)).
compute fcha2=(dfres*r2c)/(nx*(1-r2)).
do if (hc3 = 1).
loop kk = 1 to nx.
compute xprob={xprob, x(:,(ncol(x)-
(2*nx)+kk))}.
end loop.
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute k3 = nrow(bmultc).
compute xhc=xprob.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)=
xhc(i3,:)*inv(t(xprob)*xprob)*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).
end loop.
compute
varbc=(inv(t(xprob)*xprob)*t(xhc)*xhc*inv(t(xpr
ob)*xprob)).
compute lmat=make((nrow(bmultc)-nx),nx,0).
compute lmat2=ident(nx).

```

```

compute lmat={lmat;lmat2}.
compute fcha2 =
(t(t(lmat)*bmultc)*inv(t(lmat)*varbc*lmat)*((t(lm
at)*bmultc)))/nx).
end if.
compute pvalr2cc=1-fcdf(fcha2,nx,dfres).
compute rcha2 = {r2c, fcha2, nx,
dfres,pvalr2cc}.
print rcha2/title = 'Test of equality of conditional
means at this value of the moderator'/clabels
'R2-chng' 'F' 'df1' 'df2' 'p'/format F10.4.
else if (ovals = 2).
compute btmp=b.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=xprob.
compute bt1 = make(ncol(xlp),1,0).

.
compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.

```

```

end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&*ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL4.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(lldiff,nx).
compute rcha2 = {LLdiff, nx,pchi}.
print rcha2/title = 'Test of equality of log odds
conditioned on this moderator value'/clabels
'Chi-sq' 'df' 'p'/format F10.4.
end if.
end if.
compute
tttt=make(nrow(dummat),1,probeval(jj,1)).
compute tttt={tttt,dummat(:,2:ncol(dummat))}.
loop kkk=1 to nx.
compute tttt={tttt,tttt(:,1)&*tttt(:,1+kkk)}.
end loop.
compute ones=make(nrow(dummat),1,1).
do if (ncol(x) > (2*nx+2)).
compute
covmnmat=make(nrow(tttt),ncol(covmns),0).
loop kkk=1 to nrow(tttt).
compute covmnmat(kkk,:)=covmns.
end loop.
compute tttt={covmnmat,tttt}.

```

```

else.
compute tttt={ones,tttt}.
end if.
compute focvals={focvals;dummat(:,1),tttt}.
do if (mcfoc > 0).
compute yhat={dummat(:,1),(tttt*b)}.
compute cmnms={fciv, 'yhat'}.
do if (ovals <> 2).
print yhat/title = 'Estimated conditional means
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (ovals = 2).
print yhat/title = 'Estimated conditional log odds
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (jj <> nrow(probeval)).
print/title='-----'/space=0.
end if.
end if.
end loop.
compute focvals=focvals(2:nrow(focvals),:).
compute yhat=focvals(:,2:ncol(focvals))*b.
compute
focvals={focvals(:,1),focvals(:,(ncol(focvals)-
(2*nx))),yhat}.
compute cnms={fciv,mdtr,'yhat'}.
do if (mcmod > 0).
compute cnms={mdtr,fciv,'yhat'}.
end if.
end if.
do if (mcmod > 0).
compute outp=make((nx+1),7,0).
compute bcatm={b((ncol(x)-
(2*nx)),1);b((ncol(x)-nx+1):ncol(x),1)}.
compute outp(:,1)=dummat(:,1).
compute bcatcov=varb((ncol(x)-
nx):ncol(x),(ncol(x)-nx):ncol(x)).

```

```

compute bcatcov(1,1)=varb((ncol(x)-
(2*nx)),(ncol(x)-(2*nx))).
compute
bcatcov(2:nrow(bcatcov),1)=varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx)).
compute
bcatcov(1,2:nrow(bcatcov))=t(varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx))).
loop i = 1 to nrow(dummat).
compute
catmval={1,dummat(i,2:ncol(dummat))}.
compute condeff=catmval*bcatm.
compute
condse=sqrt(catmval*bcatcov*t(catmval)).
compute outp(i,2:3)={condeff,condse}.
end loop.
compute outp(:,4)=outp(:,2)&/outp(:,3).
compute outp(:,5) = 2*(1-tcdf(abs(outp(:,4)),
dfres)).
compute outp(:,6) = (outp(:,2)-
sqrt(tval)&*outp(:,3)).
compute outp(:,7) =
(outp(:,2)+sqrt(tval)&*outp(:,3)).
do if (ovals <> 2).
compute cnmms = {xname2(1,1), 'coeff', 'se',
't', 'p', 'LLCI', 'ULCI'}.
end if.
do if (ovals = 2).
compute outp(:,5) = 2*(1-
cdfnorm(abs(outp(:,4)))).
compute cnmms = {xname2(1,1), 'coeff', 'se',
'Z', 'p', 'LLCI', 'ULCI'}.
end if.
print outp/title = 'Conditional Effect of Focal
Predictor in Groups Defined by the Moderator
Variable: '/cnames=cnmms/format = F10.4.
end if.
do if (nval > 2 and (mmodval = 999) and
mcmmod = 0).
do if (quantile <> 1).

```

```

print/title = 'Moderator values are the sample
mean and plus/minus one SD from mean'.
else.
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
end if.
do if (highwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD above the mean was'.
print/title = 'replaced with the maximum
because one SD above the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
do if (lowwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD below the mean was'.
print/title = 'replaced with the minimum
because one SD below the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
end if.
do if (nval = 2 and (mmodval = 999)).
print/title = 'The moderator variable is
dichotomous'.
end if.
do if (mmodval <> 999 and mcmmod = 0 and
((mmodval < mdmin) or (mmodval > mdmax))).
print/title = 'Warning: Moderator value specified
is outside of the range of the data'.
end if.
do if (plot = 1).
do if (mcfoc = 0 and mcmmod=0).
compute cnms = {t(nms((ncol(x)-2):(ncol(x)-
1),1)), 'yhat'}.
end if.
do if (ovals = 2).

```

```

compute prob =
exp(focvals(:,3))/(1+exp(focvals(:,3))).
compute focvals = {focvals, prob}.
compute cnms = {cnms, 'prob'}.
end if.
print/title =
*****
*****'

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (ovals = 2).
do if (mcfoc=0).
print/title = 'DATA LIST FREE/Eco_Innovation
Pro_Env_Strgy logodds prob.' /space=1.
end if.
do if (mcmod=0).
print/title = 'DATA LIST FREE/Pro_Env_Strgy
Eco_Innovation logodds prob.' /space=1.
end if.
else if (ovals <> 2).
do if (mcfoc=0).
print/title = 'DATA LIST FREE/Eco_Innovation
Pro_Env_Strgy Financial_Performance.'
/space=1.
end if.

```



```

do if (mcmmod=0).
print/title = 'DATA LIST FREE/Pro_Env_Strgy
Eco_Innovation Financial_Performance.'
/space=1.
end if.
end if.
print/title = 'BEGIN DATA.' /space=0.
print focvals/title = ' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (ovals <> 2 and mcmmod=0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Financial_Performance BY
Pro_Env_Strgy.' /space=0.
end if.
do if (ovals <> 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH Financial_Performance BY
Eco_Innovation.' /space=0.
end if.
do if (ovals = 2 and mcmmod=0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Pro_Env_Strgy.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Pro_Env_Strgy.' /space=0.
end if.
do if (ovals = 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH logodds BY Eco_Innovation.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH prob BY Eco_Innovation.' /space=0.
end if.
do if (ncovs > 0).
print/title = 'NOTE: For data above, covariates
are set to their sample means.'.

```

```

end if.
end if.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
end if.
END IF.
end if.
do if (bad > 0).
compute note(notes,1) = 9.
compute notes = notes + 1.
end if.
print/title = '***** ANALYSIS
NOTES AND WARNINGS
*****'.
loop i = 1 to errs.
do if (runerrs(i,1) = 1).
print/title = 'ERROR: One of your declared
mediators is dichotomous. This procedure
can"t be used.'.
end if.
do if (runerrs(i,1) = 2).
print/title = 'ERROR: For model 6, this
procedure limits the number of mediators to
four.'.
end if.
do if (runerrs(i,1) = 3).
print/title = 'ERROR: For models 1, 2, and 3,
only a single variable can be listed in the M
list.'.
end if.
do if (runerrs(i,1) = 4).

```

```
print/title = 'ERROR: You requested a model
involving W but did not provide a valid W
variable name.'.
end if.
do if (runerrs(i,1) = 5).
print/title = 'ERROR: You requested a model
involving Z but did not provide a valid Z
variable name.'.
end if.
do if (runerrs(i,1) = 6).
print/title = 'ERROR: You requested a model
involving Q but did not provide a valid Q
variable name.'.
end if.
do if (runerrs(i,1) = 7).
print/title = 'ERROR: You requested a model
involving V but did not provide a valid V
variable name.'.
end if.
do if (runerrs(i,1) = 8).
print/title = 'ERROR: You specified a W
variable for a model that does not need it'.
end if.
do if (runerrs(i,1) = 9).
print/title = 'ERROR: You specified a Z variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 10).
print/title = 'ERROR: You specified a Q variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 11).
print/title = 'ERROR: You specified a V variable
for a model that does not need it.'.
end if.
do if (runerrs(i,1) = 12).
print/title = 'ERROR: The variable specified for
W has already been assigned.'.
end if.
do if (runerrs(i,1) = 13).
```

```

print/title = 'ERROR: The variable specified for
Z has already been assigned.'.
end if.
do if (runerrs(i,1) = 14).
print/title = 'ERROR: The variable specified for
Q has already been assigned.'.
end if.
do if (runerrs(i,1) = 15).
print/title = 'ERROR: The variable specified for
V has already been assigned.'.
end if.
do if (runerrs(i,1) = 16).
print/title = 'ERROR: You did not provide a
valid Y variable name.'.
end if.
do if (runerrs(i,1) = 17).
print/title = 'ERROR: The variable specified for
Y has already been assigned.'.
end if.
do if (runerrs(i,1) = 18).
print/title = 'ERROR: Model 6 requires more
than one mediator.'.
end if.
do if (runerrs(i,1) = 19).
print/title = 'ERROR: You have not specified a
valid model number.'.
end if.
do if (runerrs(i,1) = 20).
print/title = 'ERROR: At least one and only one
variable must be listed for X.'.
end if.
do if (runerrs(i,1) = 21).
print/title = 'ERROR: At least one and only one
variable must be listed for Y.'.
end if.
do if (runerrs(i,1) = 22).
print/title = 'ERROR: Iteration didn"t converge
to a solution. Interpret results with caution.'.
end if.
do if (runerrs(i,1) = 23).

```

```
print/title = 'ERROR: You specified a clustering
variable that does not exist in your variable
list.'.
end if.
do if (runerrs(i,1) = 24).
print/title = 'ERROR: You specified a clustering
variable that has already been assigned.'.
end if.
do if (runerrs(i,1) = 25).
print/title = 'ERROR: One or more of your M
variables is not listed in the variables list.'.
end if.
do if (runerrs(i,1) = 26).
print/title = 'ERROR: A maximum of 20 cluster
units is allowed. Use multilevel modeling
instead.'.
end if.
do if (runerrs(i,1) = 27).
print/title = 'ERROR: One of the variables in
your model is a constant.'.
end if.
do if (runerrs(i,1) = 28).
print/title = 'ERROR: Dichotomous Y is not
permitted with WS option.'.
end if.
do if (runerrs(i,1) = 29).
print/title = 'ERROR: Insufficient number of
variables in vars= list when using WS option.'.
end if.
do if (runerrs(i,1) = 30).
print/title = 'ERROR: Too many variables in
vars= list when using WS option. Covariates
not allowed.'.
end if.
do if (runerrs(i,1) = 31).
print/title = 'ERROR: mmodval and wmodval
can"t both be set to zero with contrast option.'.
end if.
do if (runerrs(i,1) = 32).
```

```

print/title = 'ERROR: You did not provide a
valid X variable name.'.
end if.
do if (runerrs(i,1) = 33).
print/title = 'ERROR: PROCESS requires all
variable names to be eight characters or
fewer.'.
print/title = '      Please shorten variable names
and reexecute.'/space=0.
end if.
do if (runerrs(i,1)=34).
print/title = 'ERROR: X and M cannot both be
specified as multicategorical.'.
end if.
do if (runerrs(i,1)=35).
print/title = 'ERROR: Multicategorical variable
specification available only for models 1 and
4.'.
end if.
do if (runerrs(i,1)=36).
print/title = 'ERROR: Categorical variables
cannot have more than 10 categories.'.
end if.
do if (runerrs(i,1)=37).
print/title = 'ERROR: Each group must have at
least two cases.'.
end if.
do if (runerrs(i,1)=38).
print/title = 'ERROR: Cluster option not
available with multicategorical variables.'.
end if.
do if (runerrs(i,1)=39).
print/title = 'ERROR: A maximum of 10
variables is allowed in the M= list.'.
end if.
do if (runerrs(i,1) = 41).
print/title = 'ERROR: MCM option not permitted
in model 4.'.
end if.
end loop.

```

```

do if (errs = 0).
do if (boot > 1 or mc > 0).
do if (bconoff = 1 and boot > 1).
print boot/title = 'Number of bootstrap samples
for bias corrected bootstrap confidence
intervals:'/'format = F8.0.
end if.
do if (bconoff = 0 and boot > 1).
print boot/title = 'Number of bootstrap samples
for percentile bootstrap confidence
intervals:'/'format = F8.0.
end if.
do if (mc > 1).
print mc/title = 'Number of samples for Monte
Carlo confidence intervals:'/'format = F8.0.
end if.
do if (booterr = 1).
compute badend = badend(1,2:ncol(badend)).
print badend/title = 'WARNING: Bootstrap CI
endpoints below not trustworthy. Decrease
confidence or increase bootstraps'/'format =
F10.4.
end if.
end if.
print conf/title = 'Level of confidence for all
confidence intervals in output:'/'format = F8.2.
do if ((center = 1 or ws=1) and (ncol(centvar) >
1)).
compute centvar = centvar(1,2:ncol(centvar)).
print centvar/title = 'NOTE: The following
variables were mean centered prior to
analysis:'/'format = a8.
end if.
loop i = 1 to notes.
do if (note(i,1) = 1).
print/title = 'NOTE: Confidence level restricted
to between 50 and 99.9999%. 95% confidence
is provided in output'.
end if.
do if (note(i,1) = 2).

```

```

print/title = 'NOTE: Effect size measures for
indirect effects are not available for models
with dichotomous outcomes'.
end if.
do if (note(i,1) = 3).
print/title = 'NOTE: All standard errors for
continuous outcome models are based on the
HC3 estimator'.
end if.
do if (note(i,1) = 6).
print/title = 'NOTE: The number of bootstrap
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 7).
print/title = 'NOTE: The Johnson-Neyman
method is available only for Models 1 and 3'.
end if.
do if (note(i,1) = 8).
print/title = 'NOTE: The Johnson-Neyman
method cannot be used with a dichotomous
moderator'.
end if.
do if (note(i,1)=9).
print bad/title = 'NOTE: Some bootstrap
samples had to be replaced. The number of
such replacements was:'.
end if.
do if (note(i,1) = 11).
print nmiss/title = 'NOTE: Some cases were
deleted due to missing data. The number of
such cases was:'.
end if.
do if (note(i,1) = 12).
print /title = 'NOTE: Monte Carlo method
available only for models 4 and 5.
Bootstrapping was used instead.'.
end if.
do if (note(i,1) = 13).

```



```
print/title = 'NOTE: The number of Monte Carlo
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 19).
print/title = 'NOTE: Effect sizes not available for
within-subject analyses.'.
end if.
do if (note(i,1) = 16).
print/title = 'NOTE: Normal theory tests not
available for within-subject analyses.'.
end if.
do if (note(i,1) = 17).
print/title = 'NOTE: Monte Carlo confidence
intervals not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 18 and warnrep=0).
print/title = 'WARNING: You have requested
OLS estimation with a dichotomous criterion.'.
print/title = 'Interpret model coefficients and
inferential statistics with caution.'/space=0.
compute warnrep=1.
end if.
do if (note(i,1) = 20).
print/title = 'NOTE: Saving of bootstrap
estimates not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 22).
print/title = 'NOTE: Effect size option with
covariates requires covariates in models of M
and Y.'.
end if.
do if (note(i,1) = 23).
print/title = 'NOTE: Johnson-Neyman method
not available with multicategorical X'.
end if.
do if (note(i,1) = 24).
```

| | | |
|-----------|----------------|--|
| | | <pre> print/title = 'NOTE: MMODVAL option not available with a multicategorical moderator'. end if. do if (note(i,1) = 25). print/title = 'NOTE: Omnibus test for relative indirect effects is not available with the MC option.'. end if. do if (note(i,1) = 26). print/title = 'NOTE: CONTRAST option not available with multicategorical X.'. end if. do if (note(i,1) = 27). print/title = 'NOTE: EFFSIZE option not available with multicategorical X.'. end if. do if (note(i,1) = 28). print/title = 'NOTE: NORMAL option not available with multicategorical X.'. end if. do if (note(i,1) = 31). print/title = 'NOTE: The TOTAL option is not available when Y is dichotomous.'. end if. do if (note(i,1) = 32). print/title = 'NOTE: Kappa-squared is disabled from output as of version 2.16.'. end if. do if (note(i,1) = 33). print/title = 'NOTE: COVMY option ignored in models 1, 2, and 3.'. end if. end loop. end if. end matrix. </pre> |
| Resources | Processor Time | 00:00:04.44 |
| | Elapsed Time | 00:00:04.47 |

Run MATRIX procedure:

***** PROCESS Procedure for SPSS Release 2.16.3 *****

Written by Andrew F. Hayes, Ph.D. www.afhayes.com
Documentation available in Hayes (2013). www.guilford.com/p/hayes3

Model = 4
Y = Financia
X = Pro_Env_
M = Eco_Inno

Sample size
856

Outcome: Eco_Inno

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|-----------|--------|----------|-------|
| .7582 | .5748 | .4904 | 1154.6120 | 1.0000 | 854.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|-------|-------|---------|-------|-------|--------|
| constant | .9317 | .0866 | 10.7643 | .0000 | .7618 | 1.1016 |
| Pro_Env_ | .7270 | .0214 | 33.9796 | .0000 | .6850 | .7690 |

Outcome: Financia

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|-----------|--------|----------|-------|
| .8876 | .7878 | .2573 | 1583.7928 | 2.0000 | 853.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|-------|-------|---------|-------|-------|-------|
| constant | .2082 | .0668 | 3.1163 | .0019 | .0771 | .3393 |
| Eco_Inno | .7534 | .0248 | 30.3948 | .0000 | .7047 | .8020 |
| Pro_Env_ | .1864 | .0238 | 7.8417 | .0000 | .1397 | .2330 |

***** DIRECT AND INDIRECT EFFECTS *****

Direct effect of X on Y

| Effect | SE | t | p | LLCI | ULCI |
|--------|-------|--------|-------|-------|-------|
| .1864 | .0238 | 7.8417 | .0000 | .1397 | .2330 |

Indirect effect of X on Y

| | Effect | Boot SE | BootLLCI | BootULCI |
|----------|--------|---------|----------|----------|
| Eco_Inno | .5477 | .0235 | .5030 | .5947 |

Normal theory tests for indirect effect

| Effect | se | Z | p |
|--------|-------|---------|-------|
| .5477 | .0242 | 22.6487 | .0000 |

***** ANALYSIS NOTES AND WARNINGS *****

Number of bootstrap samples for bias corrected bootstrap confidence intervals:
5000

Level of confidence for all confidence intervals in output:
95.00

----- END MATRIX -----

restore.

Matrix

| Notes | | |
|----------------|--------------------------------|----------------------|
| Output Created | | 20-OCT-2020 16:00:00 |
| Comments | | |
| Input | Active Dataset | DataSet1 |
| | Filter | <none> |
| | Weight | <none> |
| | Split File | <none> |
| | N of Rows in Working Data File | 856 |

Syntax

```
matrix.  
get dat/file = */variables =  
Financial_Performance Plan_Orga_Pract  
Eco_Innovation /names = vnames/missing =  
9999.  
compute ninit = nrow(dat).  
get dat/file = */variables =  
Financial_Performance Plan_Orga_Pract  
Eco_Innovation /names = vnames/missing =  
omit.  
get tmp/file = */variables =  
Financial_Performance /names =  
yname/missing = omit.  
get tmp2/file = */variables = Plan_Orga_Pract  
/names = xname/missing = omit.  
get tmp/file = */variables = Eco_Innovation  
/names = mnames/missing = omit.  
get tmp/file = */variables = w999999t z999999t  
v999999t q999999t.  
compute wname=tmp(1,1).  
do if (wname = ' ').  
compute wname = 'xxx'.  
end if.  
compute zname=tmp(1,2).  
do if (zname = ' ').  
compute zname = 'xxx'.  
end if.  
compute vname=tmp(1,3).  
do if (vname = ' ').  
compute vname = 'xxx'.  
end if.  
compute qname=tmp(1,4).  
do if (qname = ' ').  
compute qname = 'xxx'.  
end if.  
compute n = nrow(dat).  
compute p0=-.322232431088.  
compute p1 = -1.  
compute p2 = -.342242088547.  
compute p3 = -.0204231210245.
```

```

compute p4 = -.0000453642210148.
compute q0 = .0993484626060.
compute q1 = .588581570495.
compute q2 = .531103462366.
compute q3 = .103537752850.
compute q4 = .0038560700634.
compute badend = 0.
compute priorlo = -9999999.
compute priorhi = 9999999.
compute criterr = 0.
compute cluster = 0.
compute clsdmy = 0.
compute jndich = 0.
compute wvdich=0.
compute zqdich=0.
compute mod74dic=0.
compute warnrep=0.
compute contrast = trunc( 0 ).
do if (contrast > 2 or contrast < 0).
compute contrast = 0.
end if.
compute booterr = 0.
compute effsize = ( 0 = 1).
compute note = make(10,1,0).
compute notes = 1.
compute iterr = 0.
compute clsmtn = 0.
compute quantile = ( 0 = 1).
compute mmodval= 999.
compute jn = ( 0 = 1).
compute covcoeff=( 0 =1).
compute center = ( 0 = 1).
compute detail = ( 1 = 1).
compute coeffci = ( 1 = 1).
compute conf = 95.
compute longname = ( 1 =1).
compute bconoff=( 0 <> 1).
compute covmy = trunc( 0 ).
do if (covmy < 0 or covmy > 2).
compute covmy = 0.

```

```

end if.
do if (trunc(conf) >= 100 or (trunc(conf) <=
50)).
compute conf = 95.
compute note(notes,1) = 1.
compute notes = notes + 1.
end if.
do if (n < ninit).
compute nmiss = ninit-n.
compute note(notes,1) = 11.
compute notes = notes + 1.
end if.
compute errs = 0.
compute quantd = {0,0,0,0,0,0}.
compute quantc = {0,0,0,0,0,0}.
compute mcheck = 0.
compute ttt = 0.
compute plot = trunc( 0 ).
compute runerrs = make(50,1,0).
compute model = trunc( 4 ).
compute normal = 1.
compute ws= ( 0 =1).
compute olsdichm= ( 0 =1).
compute olsdichy= ( 0 =1).
compute mcx=trunc( 0 ).
compute mcm=trunc( 0 ).
do if (ws = 1).
do if (effsize = 1).
compute note(notes,1) = 19.
compute notes = notes + 1.
compute effsize=0.
end if.
do if (normal=1).
compute note(notes,1) = 16.
compute notes = notes + 1.
compute normal=0.
end if.
end if.
do if (model <> 4).
compute ws = 0.

```

```

end if.
do if (model > 0 and model < 4 and covmy <>
0).
compute covmy=0.
compute note(notes,1) = 33.
compute notes = notes + 1.
end if.
print/space=0.
do if (jn = 1 and model <> 1 and model <> 3).
compute note(notes,1) = 7.
compute notes = notes + 1.
end if.
do if ((model > 76) or (model < 1)).
compute model = 77.
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 19.
end if.
compute itprob=0.
do if (mcx < 0).
compute mcx=0.
end if.
do if (mcx > 4).
compute mcx=0.
end if.
do if (mcm < 0).
compute mcm=0.
end if.
do if (mcm > 4).
compute mcm=0.
end if.
do if (mcx > 0 and mcm > 0).
compute errs=errs+1.
compute runerrs(errs,1)=34.
compute criterr=1.
end if.
do if ((model <> 1 and model <> 4) and (mcx >
0 or mcm > 0)).
compute errs=errs+1.
compute runerrs(errs,1)=35.

```



```
compute criterr=1.  
end if.  
do if ((model = 4) and (mcm > 0)).  
compute errs=errs+1.  
compute runerrs(errs,1)=41.  
compute criterr=1.  
end if.  
do if ((mcx > 0) and jn <> 0).  
compute jn=0.  
compute note(notes,1) = 23.  
compute notes = notes + 1.  
end if.  
do if (mmodval <> 999 and mcm > 0).  
compute note(notes,1) = 24.  
compute notes = notes + 1.  
end if.  
compute toteff = 0.  
compute toteff = (( 0 = 1)*( 4 = 4 or 4 = 6)).  
compute varorder = 2.  
compute hc3 = ( 0 <> 0).  
compute cname = 'xxx'.  
compute centvar = {'xxx'}.  
compute nmeds = ncol(mnames).  
do if (longname=0).
```

```
compute criterr = 1.
```

```
compute errs = errs+1.
```

```
compute runerrs(errs,1) = 33.
```

1

2

3

```
end if.  
compute modelm =  
{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,6;  
1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,7;  
1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8;  
1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,9;  
1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,10;  
1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,11;  
1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,0,12;  
1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,13;  
0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,14;  
0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,15;  
0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,16;  
0,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,17;  
0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,18;  
0,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,19;  
0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,20;  
1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,21;
```

1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,7,22;
1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,23;
1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,0,0,24; 1,1,1,1

,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,25;
1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,0,0,26;
1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,0,0,27;
1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,7,28;
1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,0,7,29;
1,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,30;
1,1,0,1,0,0,1,1,0,1,0,0,0,0,0,0,0,31;
1,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,0,32;
1,1,1,1,0,0,1,1,1,1,0,0,0,0,0,0,0,33;
1,1,1,1,0,0,1,0,0,1,0,0,0,0,0,0,0,34;
1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,35;
1,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,36;
1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,37;
1,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,38;
1,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,39;
1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,0,40;
1,0,0,1,1,0,1,0,0,1,1,0,0,0,0,0,0,41;
1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,42;
1,0,0,1,1,1,1,0,0,1,1,1,0,0,0,0,0,43;
1,0,0,1,1,1,1,0,0,1,0,0,0,0,0,0,0,44;
1,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,45;
1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,46;
1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,47;
1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,48;
1,1,0,1,1,0,1,1,0,0,0,

0,0,0,0,0,0,0,0,49;
1,1,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,50;
1,1,1,1,1,0,1,1,1,0,0,0,0,0,0,0,0,51;
1,1,1,1,1,0,0,0,0,1,1,0,0,0,0,0,0,52;
1,1,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,53;
1,1,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,54;
1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,55;
1,1,1,1,1,1,0,0,0,1,1,1,0,0,0,0,0,56;
1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,57;
1,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,58;

```
1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,59;  
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,60;  
1,1,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,61;  
1,1,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,62;  
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,0,0,0,0,63;  
1,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,64;  
1,0,0,1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,65;  
1,0,0,1,0,0,0,0,0,1,0,0,1,0,0,0,0,0,0,66;  
1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,67;  
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,68;  
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,69;  
1,0,0,1,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,70;  
1,0,0,1,0,0,1,0,0,1,0,0,1,1,1,0,0,0,0,71;  
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,1,1,0,0,72;  
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,1,1,0,0
```

```
,73; 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,74;
```

```
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,75;
```

```
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,1,0,0,0,76;
```

```
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,77}.
```

```
compute wm = modelm(model, 1).
```

```
compute zm = modelm(model, 2).
```

```
compute wzm = modelm(model, 3).
```

```
compute vy = modelm(model, 4).
```

```
compute qy = modelm(model, 5).
```

```
compute vqy = modelm(model, 6).
```

```
compute wy = modelm(model, 7).
```

```
compute zy = modelm(model, 8).
```

```
compute wzy = modelm(model, 9).
```

```
compute vxy = modelm(model, 10).
```

```
compute qxy = modelm(model, 11).
```

```
compute vqxy = modelm(model, 12).
```

```
compute wmy = modelm(model, 13).
```

```
compute wvmy = modelm(model, 14).
```

```
compute wvxy = modelm(model, 15).
```

```
compute zmy = modelm(model,16).
```

```
compute wzmy = modelm(model,17).
```

```
compute xmy = modelm(model,18).
```

```
compute imm=modelm(model,19).
```

```
do if (model = 74).
```

```

compute vdich=0.
end if.
do if (ncol(xname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 20.
compute criterr = 1.
end if.
do if (ncol(mnames) > 10).
compute errs = errs+1.
compute runerrs(errs,1) = 39.
compute criterr = 1.
end if.
do if (ncol(yname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 21.
compute criterr = 1.
end if.
do if (( 999 =0) and ( 999 =0) and (model=3 or
model=2) and (contrast=1)).
compute errs = errs+1.
compute runerrs(errs,1) = 31.
compute criterr = 1.
end if.
compute xlist = (wm or zm or wzm or wy or zy
or wzy or vxy or qxy or vqxy or wvxy or xmy).
compute mlist = (vy or qy or vqy or zmy or wmy
or wzmy or xmy or (model < 4)).
compute bad=0.
do if (criterr = 0).
compute werr = 0.
compute verr = 0.
compute qerr = 0.
compute zerr = 0.
compute yerr = 1.
compute xerr = 1.
compute wlist = (wm or wzm or wy or wzy or
wm or wvmy or wvmy or wvxy or wzmy).
do if (wlist = 1 and wname = 'xxx').
compute werr = 1.
compute wlist = 0.

```

```

compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (wlist = 1 and ((wname = qname) or
(wname = vname) or (wname = zname) or
(wname = xname) or (wname = yname))).
compute werr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 12.
end if.
compute zlist = (zm or wzm or zy or wzy or
zmy or wzmy).
do if (zlist = 1 and zname = 'xxx').
compute zerr = 1.
compute zlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (zlist = 1 and ((zname = qname) or
(zname = vname) or (zname = wname) or
(zname = xname) or (zname = yname))).
compute zerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 13.
end if.
compute qlist = (qy or vqy or qxy or vqxy).
do if (qlist = 1 and qname = 'xxx').
compute qerr = 1.
compute qlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (qlist = 1 and ((qname = zname) or
(qname = vname) or (qname = wname) or
(qname = xname) or (qname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 14.
end if.

```



```

compute vlist = (vy or vqy or vxy or vqxy or
wvmy or wvxy).
do if (vlist = 1 and vname = 'xxx').
compute verr = 1.
compute vlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (vlist = 1 and ((vname = zname) or
(vname = qname) or (vname = wname) or
(vname = xname) or (vname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 15.
end if.
do if (wlist = 0 and wname <> 'xxx').
compute werr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 8.
end if.
do if (zlist = 0 and zname <> 'xxx').
compute zerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 9.
end if.
do if (qlist = 0 and qname <> 'xxx').
compute qerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 10.
end if.
do if (vlist = 0 and vname <> 'xxx').
compute verr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 11.
end if.
do if (hc3 = 1).
compute note(notes,1) = 3.
compute notes = notes+1.
end if.
compute alpha2 = (1-(conf/100))/2.

```

```

compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute cons = make(n,1,1).
compute temp = (n*sscp(dat))-
(t(csum(dat))*(csum(dat))).
compute temp = temp/(n*(n-1)).
compute temp = csum((((diag(temp))=0)).
compute temp2=1.
do if (ws = 1).
compute temp2 = (n*sscp(tmp2))-
(t(csum(tmp2))*(csum(tmp2))).
compute temp2 = temp2/(n*(n-1)).
end if.
do if ((temp > 0 and ws = 0) or (temp > 0 and
ws = 1 and temp2 > 0)).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 27.
end if.
compute nmeds = ncol(mnames).
compute mcmats=ident(nmeds*2).
compute mccoeff=make(nmeds*2,1,0).
compute sobel = make(nmeds,4,-999).
do if (model = 6 and nmeds > 4).
compute errs = errs+1.
compute runerrs(errs,1)=2.
end if.
do if (model < 4 and nmeds > 1).
compute errs = errs+1.
compute runerrs(errs,1) = 3.
end if.
compute nmods = (model = 74).
compute bad = 0.
compute intcnt = 1.
compute modvals = 0.
compute modvalsd = 0.
compute yintemp = {'int_1', 'int_2', 'int_3',
'int_4', 'int_5', 'int_6', 'int_7', 'int_8', 'int_9',

```

```

'int_10', 'int_11', 'int_12', 'int_13', 'int_14',
'int_15'}.
compute yintemp = {yintemp, 'int_16', 'int_17',
'int_18', 'int_19', 'int_20', 'int_21', 'int_22',
'int_23', 'int_24', 'int_25', 'int_26', 'int_27',
'int_28'}.
compute yintemp = {yintemp, 'int_29', 'int_30',
'int_31', 'int_32', 'int_33', 'int_34', 'int_35',
'int_36', 'int_37', 'int_38', 'int_39', 'int_40'}.
compute yintemp = {yintemp, 'int_41', 'int_42',
'int_43', 'int_44', 'int_45', 'int_46', 'int_47',
'int_48', 'int_49', 'int_50', 'int_51', 'int_52'}.
compute yintemp = {yintemp, 'int_53', 'int_54',
'int_55', 'int_56', 'int_57', 'int_58', 'int_59',
'int_60', 'int_61', 'int_62', 'int_63', 'int_64'}.
compute
cntname={'(C1)';'(C2)';'(C3)';'(C4)';'(C5)';'(C6)';'(
C7)';'(C8)';'(C9)';'(C10)';'(C11)';'(C12)';'(C13)';(
C14)';'(C15)';'(C16)';'(C17)'}.
compute
cntname={cntname;'(C18)';'(C19)';'(C20)';'(C21
)';'(C22)';'(C23)';'(C24)';'(C25)';'(C26)';'(C27)';(
C28)';'(C29)';'(C30)';'(C31)'}.
compute
cntname={cntname;'(C32)';'(C33)';'(C34)';'(C35
)';'(C36)';'(C37)';'(C38)';'(C39)';'(C40)';'(C41)';(
C42)';'(C43)';'(C44)';'(C45)'}.
compute
cntname={cntname;'(C46)';'(C47)';'(C48)';'(C49
)';'(C50)';'(C51)';'(C52)';'(C53)';'(C54)';'(C55)';(
C56)';'(C57)';'(C58)';'(C59)'}.
compute
cntname={cntname;'(C60)';'(C61)';'(C62)';'(C63
)';'(C64)';'(C65)';'(C66)';'(C67)';'(C68)';'(C69)';(
C70)';'(C71)';'(C72)';'(C73)'}.
compute
cntname={cntname;'(C74)';'(C75)';'(C76)';'(C77
)';'(C78)';'(C79)';'(C80)';'(C81)';'(C82)';'(C83)';(
C84)';'(C85)';'(C86)';'(C87)'}.

```

```

compute
cntname={cntname;'(C88)';'(C89)';'(C90)';'(C91
)';'(C92)';'(C93)';'(C94)';'(C95)';'(C96)';'(C97)';'(
C98)';'(C99)';'(C100)';'(C101)'}.
compute
cntname={cntname;'(C102)';'(C103)';'(C104)';'(
C105)'}.
compute apathnam={'a path'; 'a1 path'; 'a2
path'; 'a3 path'; 'a4 path'; 'a5 path'; 'a6 path';
'a7 path'; 'a8 path'; 'a9 path'; 'a10 path'}.
compute bpathnam={'b path'; 'b1 path'; 'b2
path'; 'b3 path'; 'b4path'; 'b5path'; 'b6 path'; 'b7
path'; 'b8 path'; 'b9 path'; 'b10 path'}.
compute modvnm = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute modvnm2 = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute mlab = { ' M1 ='; ' M2 ='; ' M3 ='; '
M4 ='; ' M5 ='; ' M6 ='; ' M7 ='; ' M8 ='; '
M9 ='; ' M10 ='}.
compute m = make(n,nmeds,1).
compute mmat = make(16,nmeds,0).
compute ymat = make(8,nmeds,0).
compute deco = make(10,1,0).
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
compute iterate = abs(trunc( 10000 )).
compute converge = abs( 0.00000001 ).
compute boot = abs(trunc( 5000 )).
do if (ws=1 and 0 > 0).
compute note(notes,1)=17.
compute notes=notes+1.
end if.
compute mc=(abs(trunc( 0 ))*(1-ws)).
do if (mc > 0 and model > 5).
do if (boot = 0).
compute boot=mc.
end if.

```

```

compute mc=0.
compute note(notes,1)=12.
compute notes = notes+1.
end if.
compute adjust = 0.
do if (mc > 0 and (model > 3 and model < 6)).
compute boot=0.
compute bconoff=0.
end if.
do if (boot > 0 and mc > 0).
compute mc = 0.
end if.
compute savboot = 0.
compute saveboot = ( 0 = 1).
do if (ws=1 and saveboot=1).
compute saveboot=0.
compute note(notes,1) = 20.
compute notes = notes + 1.
compute normal=0.
end if.
do if (saveboot =1 and boot > 0 and model >
3).
compute savboot = 1.
end if.
do if (boot <> 0 or mc <> 0).
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
loop.
compute cilow = rnd(bootsz*(1-(conf/100))/2).
compute cihigh =
trunc((bootsz*(conf/100)+(bootsz*(1-
(conf/100))/2)))+1.
do if (cilow < 1 or cihigh > bootsz).
compute
bootsz=trunc((bootsz+1000)/1000)*1000.
compute adjust = 1.
end if.
end loop if (cilow gt 0 and cihigh le bootsz).

```

```

compute boot=bootsz.
do if (mc > 0).
compute mc = bootsz.
end if.
do if (boot > 0 and mc > 0 and (model > 3 and
model < 6)).
compute boot=0.
end if.
do if (adjust = 1 and boot > 0).
compute note(notes,1)=6.
compute notes = notes+1.
end if.
do if (adjust = 1 and mc > 0).
compute note(notes,1)=13.
compute notes = notes+1.
end if.
end if.
do if (model = 6 and nmeds > 1).
compute mmpaths =
make((nmeds+2),(nmeds+2),0).
do if (nmeds = 2).
compute indboot = make(boot+1, 3, 999).
else if (nmeds = 3).
compute indboot = make(boot+1, 7, 999).
else if (nmeds = 4).
compute indboot = make(boot+1, 15, 999).
end if.
compute indlbl = {'Total:','Ind1 :','Ind2 :','Ind3
:','Ind4 :','Ind5 :','Ind6 :','Ind7 :','Ind8 :','Ind9
:','Ind10:','Ind11:','Ind12:','Ind13:','Ind14:','
Ind15:'}.
compute indlbl2 = {'Ind1','Ind2','Ind3','Ind4';
'Ind5','Ind6','Ind7','Ind8','Ind9','Ind10';
'Ind11','Ind12','Ind13','Ind14','Ind15'}.
compute indces = make(boot+1, 4, 999).
end if.
do if (model < 4).
compute boot = 0.
compute cmat = make(10,1,0).
compute zmat = make(10,1,0).

```

```

end if.
compute nvarch = make(1,ncol(dat),0).
compute wmatch = 0.
compute zmatch = 0.
compute vmatch = 0.
compute qmatch = 0.
compute mmatch=0.
compute minprobe=0.
compute maxprobe=0.
loop i = 1 to ncol(vnames).
do if (vnames(:,i)=yname).
compute y = dat(:,i).
compute nvarch(1,i)=1.
compute yerr = 0.
do if ((yname = xname) or (yname = wname) or
(yname = zname) or (yname = vname) or
(yname = qname)).
compute errs = errs+1.
compute runerrs(errs,1)=17.
end if.
end if.
do if (vnames(:,i)=xname).
compute x = dat(:,i).
compute nvarch(1,i)=1.
compute xdich = 1.
compute xerr=0.
loop jj = 1 to n.
do if ((x(jj,1) <> cmin(x)) and (x(jj,1) <>
cmax(x))).
compute xdich = 0.
break.
end if.
end loop.
compute xmean = csum(x)/n.
do if (center = 1 and (model < 4 or xlist > 0)
and mcx = 0).
compute meanvec = make(n,1, xmean).
compute x = x-meanvec.
compute centvar = {centvar, xname}.
end if.

```

```

compute xmean = csum(x)/n.
compute tmp = x-(cons*xmean).
compute xsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
do if (xdich = 0).
compute quantc(1,6) = 1.
compute matx = {xmean-xsd; xmean;
xmean+xsd}.
do if ( 999 = 999 and quantile = 0).
do if ((matx(1,1) < cmin(x)) and model = 74).
compute matx(1,1)=cmin(x).
compute minprobe=1.
end if.
do if ((matx(3,1) > cmax(x)) and model = 74).
compute matx(3,1)=cmax(x).
compute maxprobe=1.
end if.
end if.
do if (quantile = 1).
compute quantd(1,6) = 1.
compute quantc(1,6) = 0.
compute tmp = x.
compute tmp(GRADE(x)) = x.
compute matx =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (xdich = 1).
compute matx = {cmin(x); cmax(x)}.
do if (model=74).
compute matx=cmin(x).
compute mod74dic=1.
end if.
end if.
do if ( 999 <> 999).
compute matx = 999.
compute quantd(1,6) = 0.
compute quantc(1,6) = 0.
end if.

```



```

end if.
do if (werr = 0 and wlist = 1).
do if (vnames(:,i)=wname).
compute werr = 0.
compute wmatch = 1.
compute w = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(w)/n).
compute w = w-meanvec.
compute centvar = {centvar, wname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute wmean = csum(w)/n.
compute tmp = w-(cons*wmean).
compute wsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute wdich = 1.
loop jj = 1 to n.
do if ((w(jj,1) <> cmin(w)) and (w(jj,1) <>
cmax(w))).
compute wdich = 0.
break.
end if.
end loop.
do if (model = 3).
compute jndich=wdich.
compute jnmin=cmin(w).
compute jnmax=cmax(w).
end if.
do if (wdich = 0).
compute matw = {wmean-wsd; wmean;
wmean+wsd}.
do if ( 999 = 999 and quantile = 0).
do if (matw(1,1) < cmin(w)).
compute matw(1,1)=cmin(w).
compute minprobe=1.
end if.
do if (matw(3,1) > cmax(w)).
compute matw(3,1)=cmax(w).
compute maxprobe=1.

```

```

end if.
end if.
compute quantc(1,1) = 1.
do if (quantile = 1).
compute quantd(1,1) = 1.
compute quantc(1,1) = 0.
compute tmp = w.
compute tmp(GRADE(w)) = w.
compute matw =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (wdich = 1).
compute matw = {cmin(w); cmax(w)}.
compute wvdich=1.
compute cmaw=cmax(w).
compute cminw=cmin(w).
end if.
do if ( 999 <> 999).
compute matw = 999.
compute quantd(1,1) = 0.
compute quantc(1,1) = 0.
end if.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm(1,1)=wname.
compute modmatp(1,1) = 1.
end if.
end if.
do if (zerr = 0 and zlist = 1).
do if (vnames(:,i)=zname).
compute zerr = 0.
compute zmatch = 1.
compute z = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(z)/n).
compute z = z-meanvec.
compute centvar = {centvar, zname}.

```

```

end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute zmean = csum(z)/n.
compute tmp = z-(cons*zmean).
compute zsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute zdich = 1.
loop jj = 1 to n.
do if ((z(jj,1) <> cmin(z)) and (z(jj,1) <>
cmax(z))).
compute zdich = 0.
break.
end if.
end loop.
do if (zdich = 0).
compute matz = {zmean-zsd; zmean;
zmean+zsd}.
do if ( 999 = 999 and quantile = 0).
do if (matz(1,1) < cmin(z)).
compute matz(1,1)=cmin(z).
compute minprobe=1.
end if.
do if (matz(3,1) > cmax(z)).
compute matz(3,1)=cmax(z).
compute maxprobe=1.
end if.
end if.
compute quantc(1,2) = 1.
do if (quantile = 1).
compute quantd(1,2) = 1.
compute quantc(1,2) = 0.
compute tmp = z.
compute tmp(GRADE(z)) = z.
compute matz = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (zdich = 1).
compute matz = {cmin(z); cmax(z)}.

```

```

compute cmaxz=cmax(z).
compute cminz=cmin(z).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matz = 999.
compute quantd(1,2) = 0.
compute quantc(1,2) = 0.
end if.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm(1,2)=zname.
compute modmatp(1,2) = 1.
end if.
end if.
do if (verr = 0 and vlist = 1).
do if (vnames(:,i)=vname).
compute verr = 0.
compute vmatch = 1.
compute v = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(v)/n).
compute v = v-meanvec.
compute centvar = {centvar, vname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute vmean = csum(v)/n.
compute tmp = v-(cons*vmean).
compute vsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute vdich = 1.
loop jj = 1 to n.
do if ((v(jj,1) <> cmin(v)) and (v(jj,1) <>
cmax(v))).
compute vdich = 0.
break.
end if.
end loop.
do if (vdich = 0).

```

```

compute matv = {vmean-vsd; vmean;
vmean+vsd}.
do if ( 999 = 999 and quantile = 0).
do if (matv(1,1) < cmin(v)).
compute matv(1,1)=cmin(v).
compute minprobe=1.
end if.
do if (matv(3,1) > cmax(v)).
compute matv(3,1)=cmax(v).
compute maxprobe=1.
end if.
end if.
compute quantc(1,3) = 1.
do if (quantile = 1).
compute quantd(1,3) = 1.
compute quantc(1,3) = 0.
compute tmp = v.
compute tmp(GRADE(v)) = v.
compute matv = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (vdich = 1).
compute matv = {cmin(v); cmax(v)}.
compute wvdich=1.
compute cmaxv=cmax(v).
compute cminv=cmin(v).
end if.
do if ( 999 <> 999).
compute matv = 999.
compute quantd(1,3) = 0.
compute quantc(1,3) = 0.
end if.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm(1,3)=vname.
compute modmatp(1,3) = 1.
end if.
end if.

```

```

do if (qerr = 0 and qlist = 1).
do if (vnames(:,i)=qname).
compute qerr = 0.
compute qmatch = 1.
compute q = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(q)/n).
compute q = q-meanvec.
compute centvar = {centvar, qname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute qmean = csum(q)/n.
compute tmp = q-(cons*qmean).
compute qsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute qdich = 1.
loop jj = 1 to n.
do if ((q(jj,1) <> cmin(q)) and (q(jj,1) <>
cmax(q))).
compute qdich = 0.
break.
end if.
end loop.
do if (qdich = 0).
compute matq = {qmean-qsd; qmean;
qmean+qsd}.
do if ( 999 = 999 and quantile = 0).
do if (matq(1,1) < cmin(q)).
compute matq(1,1)=cmin(q).
compute minprobe=1.
end if.
do if (matq(3,1) > cmax(q)).
compute matq(3,1)=cmax(q).
compute maxprobe=1.
end if.
end if.
compute quantc(1,4) = 1.
do if (quantile = 1).
compute quantd(1,4) = 1.
compute quantc(1,4) = 0.

```

```

compute tmp = q.
compute tmp(GRADE(q)) = q.
compute matq = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (qdich = 1).
compute matq = {cmin(q); cmax(q)}.
compute cmaxq=cmax(q).
compute cminq=cmin(q).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matq = 999.
compute quantd(1,4) = 0.
compute quantc(1,4) = 0.
end if.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm(1,4)=qname.
compute modmatp(1,4) = 1.
end if.
end if.
do if (vnames(:,i)=cname).
compute cld = dat(:,i).
compute cvname = vnames(:,i).
compute nvarch(1,i)=1.
compute clsmtch = 1.
end if.
loop j = 1 to ncol(mnames).
do if (vnames(:,i)=mnames(1,j)).
compute mmatch = mmatch + 1.
compute m(:,j)=dat(:,i).
do if (center = 1 and nvarch(1,i) = 0 and mlist >
0 and mcm = 0).
compute meanvec = make(n,1,csum(m(:,j))/n).
compute m(:,j) = m(:,j)-meanvec.
compute centvar = {centvar, mnames(1,j)}.
end if.

```

```

compute nvarch(1,i)=1.
compute dichm = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute dichm = 0.
break.
end if.
end loop.
do if ((dichm = 1) and (olsdichm = 1)).
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichm = 1 and model > 3 and mcheck = 0
and olsdichm = 0).
compute errs= errs+1.
compute runerrs(errs,1) = 1.
compute mcheck = 1.
end if.
do if ((model <= 3) and (ncol(mnames)=1)).
compute nmods = nmods + 1.
compute mmean = csum(m(:,j))/n.
compute tmp = m(:,j)-(cons*mmean).
compute msd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute mdich = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute mdich = 0.
break.
end if.
end loop.
do if (model = 1).
compute jndich=mdich.
compute jnmin=cmin(m(:,j)).
compute jnmax=cmax(m(:,j)).
end if.
do if (mdich = 0).
compute matm = {mmean-msd; mmean;
mmean+msd}.

```



```

do if ( 999 = 999 and quantile = 0).
do if (matm(1,1) < cmin(m(:,j))).
compute matm(1,1)=cmin(m(:,j)).
compute minprobe=1.
end if.
do if (matm(3,1) > cmax(m(:,j))).
compute matm(3,1)=cmax(m(:,j)).
compute maxprobe=1.
end if.
end if.
compute quantc(1,5) = 1.
do if (quantile = 1).
compute quantd(1,5) = 1.
compute quantc(1,5) = 0.
compute tmp = m(:,j).
compute tmp(GRADE(m(:,j))) = m(:,j).
compute matm = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (mdich = 1).
compute matm = {cmin(m); cmax(m)}.
end if.
do if ( 999 <> 999).
compute matm = 999.
compute quantd(1,5) = 0.
compute quantc(1,5) = 0.
end if.
compute modmatv(1,5)=nrow(matm).
compute modmat((1:nrow(matm)), 5) = matm.
compute modvnm(1,5)=mnames(1,j).
compute modmatp(1,5) = 1.
end if.
end if.
end loop.
end loop.
do if (minprobe=1).
compute note(notes,1) = 14.
compute notes = notes + 1.

```

```

end if.
do if (maxprobe=1).
compute note(notes,1) = 15.
compute notes = notes + 1.
end if.
do if (cname <> 'xxx' and clsmatch = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 23.
end if.
do if (cname <> 'xxx').
do if ((cname = zname) or (cname = vname)
or (cname = wname) or (cname = xname) or
(cname = yname) or (cname = qname)).
compute errs = errs+1.
compute runerrs(errs,1) = 24.
end if.
end if.
do if (wlist = 1 and werr = 0 and wmatch = 0).
compute werr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (zlist = 1 and zerr = 0 and zmatch = 0).
compute zerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (qlist = 1 and qerr = 0 and qmatch = 0).
compute qerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (vlist = 1 and verr = 0 and vmatch = 0).
compute verr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (yerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 16.

```

```

end if.
do if (xerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 32.
end if.
do if (model = 6 and nmeds < 2).
compute errs = errs+1.
compute runerrs(errs,1) = 18.
end if.
do if (mmatch < ncol(mnames)).
compute errs=errs+1.
compute runerrs(errs,1) = 25.
end if.
do if (cname <> 'xxx').
compute cld = design(cld).
compute cluster = ncol(cld).
compute cld = cld(:,2:ncol(cld)).
compute clsdmy = ncol(cld).
do if (clsdmy > 19).
compute errs = errs+1.
compute runerrs(errs,1) = 26.
end if.
do if (mcx > 0 or mcm > 0).
compute errs = errs+1.
compute runerrs(errs,1) = 38.
end if.
end if.
compute dichy = 1.
loop jj = 1 to n.
do if ((y(jj,1) <> cmin(y)) and (y(jj,1) <>
cmax(y))).
compute dichy = 0.
break.
end if.
end loop.
do if ((dichy = 1) and (olsdichy = 1)).
compute dichy=0.
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.

```

```

do if (dichy = 1).
do if (toteff=1).
compute toteff=0.
compute note(notes,1) = 31.
compute notes = notes + 1.
end if.
compute jncrit=xp2*xp2.
do if (ws = 1).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 28.
end if.
end if.
compute ncovs = ncol(dat)-rsum(nvarch).
do if (effsize = 1 and covmy <> 0 and ncovs >
0 and model > 3 and model < 7).
compute note(notes,1) = 22.
compute notes = notes + 1.
compute effsize = 0.
end if.
do if (ws = 1 and ncovs < nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 29.
end if.
do if (ws = 1 and ncovs > nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 30.
end if.
end if.
do if (errs = 0).
do if (rsum(quantd) > 0).
compute note(notes,1) = 4.
compute notes = notes+1.
end if.
do if (rsum(quantc) > 0).
compute note(notes,1) = 5.
compute notes = notes+1.
end if.

```

```

do if (ncovs > 0).
compute c = make(n,ncovs,0).
compute cnames = {'x'}.
compute j = 1.
loop i = 1 to ncol(vnames).
do if (nvarch(1,i)) = 0.
compute c(:,j) = dat(:,i).
compute nvarch(1,i)=1.
compute j=j+1.
compute cnames = {cnames, vnames(:,i)}.
end if.
end loop.
compute cnames = cnames(1,2:ncol(cnames)).
do if (ws = 1).
compute covmean=csum(c)/n.
loop i = 1 to ncovs.
compute meanvec = make(n,1, covmean(1,i)).
compute c(:,i) = c(:,i)-meanvec.
end loop.
compute centvar = {centvar, cnames}.
compute x=c(:,1:ncovs).
do if (ncovs=nmeds).
compute ncovs=0.
end if.
end if.
end if.
compute names = {yname, xname, mnames,
wname, zname, vname, qname}.
do if (ncovs > 0).
compute names = {names, cnames}.
end if.
do if (dichy = 1 and effsize = 1).
compute note(notes,1) = 2.
compute notes = notes+1.
end if.
do if (model > 3 and model < 6).
compute indeff=make(nmeds,1,0).
compute indboot=make(boot+1,nmeds,999).
do if (mc > 0).
compute indboot=make(mc+1,nmeds,999).

```

```

end if.
do if (effsize = 1 and dichy = 0).
compute rmeff=make(boot+1,nmeds+1,999).
compute abpseff=make(boot+1,nmeds+1,999).
compute abcseff=make(boot+1,nmeds+1,999).
compute pmeff=make(boot+1,nmeds+1,999).
compute r245 = make(boot+1,1,999).
compute kappa2 = make(boot+1,1,999).
end if.
end if.
do if (model = 6 and effsize = 1 and dichy = 0).
compute
rmeff=make(boot+1,ncol(indboot),999).
compute
abpseff=make(boot+1,ncol(indboot),999).
compute
abcseff=make(boot+1,ncol(indboot),999).
compute
pmeff=make(boot+1,ncol(indboot),999).
end if.
do if (nmods > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm(1,tmp) = modvnm(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:nmods).
compute modvnm=modvnm(:,1:nmods).
compute modmatv=modmatv(:,1:nmods).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.

```

```

compute modvals =
make((modmatv(1,1)*modprod(1,1)),
nmods,0).
loop i = 1 to nmods.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvals)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvals(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
do if (model = 74).
compute modvals=matx.
compute modvnm=xname.
end if.
compute vmat = make(8,nrow(modvals),0).
compute vmat(1,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute vmat(5,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute indeff=make(nrow(modvals),1,0).
do if (model <> 5).
compute indboot = make(((boot+1)*nmeds),
nrow(modvals), -99999999).
compute indbootp =
make(boot+1,(nmeds*nmods), -99999999).
do if (imm > 4 and imm < 8).
do if (imm = 7).
compute immmv1=matw.
compute immmv2=matv.
end if.
do if (imm = 5).
compute immmv1=matw.
compute immmv2=matz.
end if.

```

```

do if (imm = 6).
compute immmv1=matv.
compute immmv2=matq.
end if.
compute
indboop1=make((boot+1),(nmeds*(nrow(immmv2))),-9999999).
compute
indboop2=make((boot+1),(nmeds*(nrow(immmv1))),-9999999).
compute onesmv1=make(1,nrow(immmv1),1).
compute onesmv2=make(1,nrow(immmv2),1).
compute
nimvs=nrow(immmv1)+nrow(immmv2).
compute cmmval=0.
compute cmmlbs1={'xx'}.
compute cmmlbs2={'xx'}.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv2}.
compute
cmmlbt=make(nrow(immmv2),1,mnames(1,i)).
compute cmmlbs1={cmmlbs1;cmmlbt}.
end loop.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv1}.
compute
cmmlbt=make(nrow(immmv1),1,mnames(1,i)).
compute cmmlbs2={cmmlbs2;cmmlbt}.
end loop.
compute cmmval=cmmval(2:nrow(cmmval),:).
end if.
end if.
end if.
do if (nmods> 0).
loop i = 1 to ncol(modvals).
do if (modvnm(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm(1,i)=zname).
compute zcol = i.

```



```

end if.
do if (modvnm(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm(1,i)=qname).
compute qcol = i.
end if.
end loop.
end if.
do if (dichy = 1).
compute omx = cmax(y).
compute omn = cmin(y).
compute y = (y = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute data = {cons,y,m,x}.
compute datamed = data.
compute datayed = data.
compute datanm = {'constant'; yname;
t(mnames); xname}.
compute datanmm = {'constant'; yname;
t(mnames); xname}.
compute datanmy = {'constant'; yname;
t(mnames); xname}.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
do if (model < 4 and errs = 0).
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' ', ' '}.
compute datayed = {datayed, x&*m}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(1,i) = 1.
compute vmat(2,i) = modvals(i,1).
end loop.
compute mmat = make(16,nmeds,1).
end if.
do if (model = 2 or model = 3).
compute int1 = x&*w.

```

```

compute datayed = {datayed, w, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,2).
compute vmat(3,i) = modvals(i,1).
compute vmat(4,i) = modvals(i,1)*modvals(i,2).
end loop.
end if.
do if (model = 3).
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*m.
compute int2 = x&*w&*m.
compute datayed = {datayed, int1, int2}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' X', wname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (model = 4 or model = 5).
compute vmat = make(8,1,1).
end if.
compute yintkey2 = yintkey.
do if (wm = 1).
compute int1 = x&*w.
compute datamed = {datamed, w,int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmm = {datanmm; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).

```

```

compute vmat(2,i) = modvals(i,wcol).
end loop.
do if (zm = 1).
compute int1 = x*z.
compute datamed = {datamed, z, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(3,i) = modvals(i,zcol).
end loop.
end if.
do if (wzm = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w*z.
compute int2 = x*w*z.
compute datamed = {datamed, int1, int2}.
loop i = 1 to nrow(modvals).
compute vmat(4,i) =
modvals(i,wcol)*modvals(i,zcol).
end loop.
end if.
end if.
compute mdatacol=ncol(datamed).
compute mintkey = yintkey.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
compute medints = intcnt-1.
do if (vy = 1 or xmy = 1).
compute mp = 1.

```

```

do if (xmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,1).
end loop.
end if.
do if (vy = 1).
compute datayed = {datayed, v}.
compute datanmy = {datanmy; vname}.
compute mmods = 1.
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,vcol).
end loop.
do if (qy = 1).
compute mp = 2.
compute datayed = {datayed,q}.
compute datanmy = {datanmy; qname}.
compute mmods = 2.
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,qcol).
end loop.
end if.
do if (vqy = 1).
compute mp = 3.
compute datayed = {datayed, v&*q}.
compute mmods = 3.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,vcol)*modvals(i,qcol).
end loop.
end if.
end if.
compute mints = make(n,(nmeds*mp),0).
loop i = 0 to (nmeds-1).
do if (i = 0 and vqy = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
vname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.

```

```

do if (vy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (xmy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))*x.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', xname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (qy = 1).
compute mints(:,((i*mp)+2))=m(:,(i+1))*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqy = 1).
compute mints(:,((i*mp)+3))=m(:,(i+1))*v*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end loop.
compute datayed = {datayed, mints}.
end if.
compute mp = 1.
do if (wvmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).

```

```

compute vmat(8,i) =
modvals(i,wcol)*modvals(i,vcol).
end loop.
end if.
compute mints2 = make(n,(nmeds*mp),0).
do if (wmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,wcol).
end loop.
do if (wy = 0 and model > 3).
compute datayed = {datayed, w}.
compute datanmy = {datanmy; wname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wvmy = 1).
compute datayed = {datayed,w&*v}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints2(:,((i*mp)+1))= m(:,(i+1))&*w.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wvmy = 1).
compute mints2(:,((i*mp)+2))=
m(:,(i+1))&*w&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints2}.
end if.

```

```

compute mp = 1.
do if (zmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,zcol).
end loop.
do if (wzmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,zcol)&*modvals(i,wcol).
end loop.
end if.
end if.
do if (zmy = 1).
compute mints3 = make(n,(nmeds*mp),0).
do if (zy = 0).
compute datayed = {datayed, z}.
compute datanmy = {datanmy; zname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wzmy = 1 and wzy = 0).
compute datayed = {datayed,w&*z}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints3(:,((i*mp)+1))= m(:,(i+1))&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wzmy = 1).
compute mints3(:,((i*mp)+2))=
m(:,(i+1))&*w&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', zname}.

```

```

compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints3}.
end if.
compute decoc = 1.
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
do if (wy = 1 and model > 3).
compute datayed = {datayed, w, x&*w}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm2(1,1)=wname.
compute modmatp(1,1) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (zy = 1).
compute datayed = {datayed,z,x&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm2(1,2)=zname.
compute modmatp(1,2) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.

```



```

do if (wzy = 1).
compute datayed = {datayed,w&*z,x&*w&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vxy = 1).
compute datayed = {datayed, x&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm2(1,3)=vname.
compute modmatp(1,3) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (qxy = 1).
compute datayed = {datayed, x&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm2(1,4)=qname.
compute modmatp(1,4) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.

```

```

compute intcnt = intcnt+1.
do if (vqxy = 1).
compute datayed = {datayed, x&*v&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, 'X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end if.
do if (wvxy = 1).
compute datayed = {datayed, x&*w&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, 'X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute modvalsd = 0.
compute ttt = rsum(modmatp).
do if (rsum(modmatp) > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm2(1,tmp) = modvnm2(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:ttt).
compute modvnm2=modvnm2(:,1:ttt).
compute modmatv=modmatv(:,1:ttt).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.

```

```

loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvalsd =
make((modmatv(1,1)*modprod(1,1)), ttt,0).
loop i = 1 to ttt.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvalsd)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvalsd(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
end if.
do if (ttt > 0).
loop i = 1 to ncol(modvalsd).
do if (modvnm2(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm2(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm2(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm2(1,i)=qname).
compute qcol = i.
end if.
end loop.
compute directv = make(nrow(modvalsd),1,1).
do if (wy = 1).
compute directv ={directv, modvalsd(:,wcol)}.
end if.

```

```

do if (zy = 1).
compute directv = {directv, modvalsd(:,zcol)}.
end if.
do if (wzy = 1).
compute directv = {directv,
(modvalsd(:,wcol)*modvalsd(:,zcol))}.
end if.
do if (vxy = 1).
compute directv = {directv, modvalsd(:,vcol)}.
end if.
do if (qxy = 1).
compute directv = {directv, modvalsd(:,qcol)}.
end if.
do if (vqxy = 1).
compute directv = {directv,
(modvalsd(:,vcol)*modvalsd(:,qcol))}.
end if.
do if (wvxy = 1).
compute directv = {directv,
(modvalsd(:,vcol)*modvalsd(:,wcol))}.
end if.
end if.
compute ydatacol = ncol(datayed).
do if (ncovs > 0).
do if (covmy <> 2).
compute datamed = {datamed,c}.
end if.
do if (covmy <> 1).
compute datayed = {datayed,c}.
end if.
do if (covmy = 1 and model = 4 and mcx > 0).
compute datayed = {datayed,c}.
end if.
compute covmeans = csum(c)/n.
end if.
do if (cluster > 0).
compute datamed = {datamed,cld}.
compute datayed = {datayed, cld}.
compute cldmeans = csum(cld)/n.
end if.

```

```

compute mst = 3.
compute mnd = mst+nmeds-1.
compute ydatacol=ncol(datayed).
compute mdatacol=ncol(datamed).
do if (ncovs > 0).
compute datanmy = {datanmy; t(cnames)}.
do if (model > 3).
compute datanmm = {datanmm; t(cnames)}.
end if.
end if.
compute datanmy = {'constant';
datanmy(3:nrow(datanmy),1)}.
do if (model > 3).
compute datanmm = {'constant';
datanmm(3:nrow(datanmm),1)}.
end if.
compute amm = make(2,1,0).
compute abmm = make(2,1,0).
compute mnv = csum(datayed(:,2)/n).
compute mnv = make(n,1,mnv).
compute ssty = csum((datayed(:,2)-mnv)**2).
compute sigma = (n*sscp(datayed)-
(t(csum(datayed))*(csum(datayed)))).
compute sigma = sigma/(n*(n-1)).
do if (ws=1).
compute stddevm=diag(sigma).
compute
stddevm=sqrt(stddevm(3:(nmeds+2),1)).
end if.
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute r2xy =
(sigma(2,(3+nmeds))/(stddevy*stddevx))**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))**2.
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).

```

```

do if (model = 4 and nmeds = 1 and cluster = 0
and ncovs = 0).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
compute datatm = datamed.
compute dataty = datayed.
compute mdlrms2 = { '4' ; yname; xname}.
compute mdlrms = {'Model ='; ' Y ='; ' X ='
}.
do if (ws = 1).
compute mdlrms2 = { '4' ; yname}.
compute mdlrms = {'Model ='; ' Y ='}.
end if.
loop i = 1 to ncol(mnames).
compute mdlrms2 = {mdlrms2; mnames(1,i)}.
do if (i = 1 and ncol(mnames) = 1).
compute mdlrms = {mdlrms; ' M ='}.
else.

```

```

compute mdlInms = {mdlInms; mlab(i,1)}.
end if.
end loop.
do if (wname <> 'xxx').
compute mdlInms2 = {mdlInms2; wname}.
compute mdlInms = {mdlInms; '  W = '}.
end if.
do if (zname <> 'xxx').
compute mdlInms2 = {mdlInms2; zname}.
compute mdlInms = {mdlInms; '  Z = '}.
end if.
do if (vname <> 'xxx').
compute mdlInms2 = {mdlInms2; vname}.
compute mdlInms = {mdlInms; '  V = '}.
end if.
do if (qname <> 'xxx').
compute mdlInms2 = {mdlInms2; qname}.
compute mdlInms = {mdlInms; '  Q = '}.
end if.
do if (jn = 1 and model = 1 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
do if (jn = 1 and model = 3 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
compute mdpbe=0.
compute medte2=0.
do if ( 4 =4 and mcx > 0).
compute medte2=1.
end if.
do if (((mcx > 0) or (mcm > 0)) AND model =
1).
compute mdpbe=1.
compute dd={datayed(:,4),datayed(:,3)}.
do if (ncovs > 0).
compute dd={datayed(:,6:(5+ncovs)),dd}.
end if.
compute dd={datayed(:,2),dd}.

```

```

end if.
compute yes = ((nrow(modvals)=1) and
(contrast = 1) and (model = 3 or model = 2)).
compute yes2 = ((nrow(modvals)=1) and
(contrast = 1) and (model = 2)).
print/title = '***** PROCESS Procedure
for SPSS Release 2.16.3 *****'.
print/title = '      Written by Andrew F. Hayes,
Ph.D.      www.afhayes.com'.
print/title = '  Documentation available in
Hayes (2013).
www.guilford.com/p/hayes3'/space=0.
print mdlnms2/title =
*****
*****'/rnames = mdlnms/format =
a8.
do if (ncovs > 0).
print cnames/title = 'Statistical Controls: '/rlabels
= 'CONTROL=' /format a8.
end if.
print n/title = 'Sample size' /format F10.0.
do if (cluster > 0).
print cluster/rnames = cvname/title =
'Clustering Variable and Number of Clusters'.
end if.
do if (model > 3 and ( 'random' <> 'random')).
compute seedt= 'random'.
print seedt/title='Custom' /format=A12/rlabel =
'Seed:'.
end if.
do if (mdpbe=0 and medte2 = 0).
loop bt = 1 to (boot+1).
do if (bt = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef = 0.
do if (bt > 1).
loop.
compute v=trunc(uniform(n,1)*n)+1.

```



```

compute datayed = dataty(v,:).
do if (ws=1).
compute
bmeans=csum(datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed)))/n.
compute cntmat=make(n,nmeds,1).
compute bmeans2=bmeans.
do if (nmeds > 1).
compute bmeans2=mdia(bmeans).
end if.
compute cntmat=cntmat*bmeans2.
compute datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed))=datayed(:,(ncol(data
yed)-nmeds+1):ncol(datayed))-cntmat.
end if.
compute detcheck=det(t(datayed)*datayed).
compute rk=(detcheck <> 0).
do if (model > 3).
compute datamed = datatm(v,:).
compute detcheck=det(t(datamed)*datamed).
do if (rk=1).
compute rk=(detcheck <> 0).
end if.
end if.
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
compute temp=diag(sigma).
compute bad = bad+(1-rk).
compute false = 1.
end loop if (rk = 1).
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nme
ds)).
do if (model = 4 and nmeds = 1 and ncovs = 0
and cluster = 0).

```

```

compute r2xy =
(sigma(2,4)/(stddevy*stddevx))&**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))&**2.
compute sstot = sigma(2,2)*(n-1).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
end if.
do if (model > 3).
loop im = 1 to nmeds.
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant';
datanmm((2+nmeds):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
do if (model = 6).

```

```

do if (im = 1).
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
end if.
do if (im > 1).
compute xm={cons, datamed(:,3:(im+1)),
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant'; datanmm(2:im,1);
datanmm((mnd):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
compute
mmpaths((im+1),(2:im))=t(coeff(2:im,1)).
end if.
end if.
do if (ws = 1).
compute
coeff(1,1)=csum(datamed(:,(2+im)))/n.
end if.
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
do if (bt = 1).
compute resid=datamed(:,(2+im))-xm*coeff.
compute sse =cssq(resid).
compute mse = sse/(n-ncol(xm)).
compute mnv = csum(data(:,(2+im)))/n.
compute mnv = make(n,1,mnv).
compute sstm = csum((data(:,(2+im))-
mnv)**2).
compute k3 = nrow(coeff).
do if (hc3 = 1).
compute h = xm(:,1).
loop i3=1 to n.
compute h(i3,1)= xm(i3,:)*invXtX*t(xm(i3,:)).
end loop.

```

```

loop i3=1 to k3.
compute xm(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xm(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xm(:,i3) = sqrt(mse)&*xm(:,i3).
end loop.
end if.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute hccov=invXtX*t(xm)*xm*invXtX.
compute mcmats(im,im)=hccov(2,2).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*hccov*lmat)*((t(lmat)
)*coeff))/dfnum.
compute coeff = coeff(1:(nrow(coeff)-
clsdmy),1).
compute mccoeff(im,1)=coeff(2,1).
compute standerr =
sqrt(diag(invXtX*t(xm)*xm*invXtX)).
compute standerr = standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1).
compute standerr(1,1)=stddevm(im,1)/sqrt(n).
end if.
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-
ncol(xm)))).
compute temp=(n-ncol(xm)).
do if (ws=1).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute temp=(n-1).
end if.
compute xd = abs(xp2).

```

```

compute temp = (temp* (exp((temp-
(5/6))*((xd/(temp-(2/3)+(.11/temp)))*(xd/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = coeff-
sqrt(abs(temp))*standerr.
compute temp2 =
coeff+sqrt(abs(temp))*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (ws=1).
compute op=op(1,:).
compute xnmn =
apathnam(((nmeds>1)+1+(im-1)),1).
end if.
compute sobel(im,1) = coeff(2,1).
compute sobel(im,2)=standerr(2,1).
compute temp = mnames(1,im).
compute r2full = 1-(sse/sstm).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute summ = {sqrt(r2full), r2full, mse,
Fratio, dfnum, dfden, pfr}.
do if (detail = 1).
print temp/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels 'R',
'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'/'format = F10.4.
end if.
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
print op/title = 'Model'/rnames = xnmn/clabels
'coeff', 'se', 't', 'p', 'LLCI', 'ULCI'/'format = F10.4.
do if (covcoeff=1 and ws <> 1).
compute
hccovtmp=hccov(1:nrow(op),1:nrow(op)).
compute cnamestp=t(xnmn).

```

```

print hccovtmp/title='Covariance matrix of
regression parameter
estimates'/rnames=xmnm/cnames=cnamestp/f
ormat= F10.4.
end if.
do if (nmods > 0 and nrow(mintkey) > 1).
print mintkey/title = 'Product terms key:.'/format
= A8.
end if.
do if (ws = 1).
compute wsdf=n-1.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
end if.
end if.
compute ymat(1,im) = coeff(2,1).
do if (ws = 1).
compute ymat(1,im)=coeff(1,1).
end if.
do if (wm = 1).
compute ymat(2,im) = coeff(4,1).
do if (zm = 1).
compute ymat(3,im) = coeff(6,1).
do if (wzm = 1).
compute ymat(4, im) = coeff(8,1).
end if.
end if.
end if.
do if (model = 6).
compute mmpaths((im+1),1)=coeff((im+1),1).
end if.
end loop.
end if.
loop totlp = 1 to (1+(toteff*(bt = 1))).
do if (toteff = 1 and totlp =2).
compute xy =
{cons,datayed(:,(3+nmeds):ydatacol)}.
else.
compute xy={cons, datayed(:,3:ydatacol)}.
end if.

```

```

do if (dichy = 1).
compute pt2 =
make(nrow(datayed(:,2)),1,(csum(datayed(:,2))
/n)).
compute LL3 = datayed(:,2)*ln(pt2)+(1-
datayed(:,2))*ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xy),1,0).
compute LL1 = 0.
compute xy22=xy.
loop jjj = 1 to iterate.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.
compute coeff =
bt1+inv(t(xy22)*xy)*t(xy)*(datayed(:,2)-pt1).
compute pt1 = 1/(1+exp(-(xy*coeff))).
compute itprob = csum((pt1 <
.00000000000001) or (pt1 >
.99999999999999)).
do if (itprob = 0).
compute LL = datayed(:,2)*ln(pt1)+(1-
datayed(:,2))*ln(1-pt1).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
break.
end if.
compute bt1 = coeff.
compute LL1 = LL2.
end loop.
do if (jjj >= iterate and iterr = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
computer iterr = 1.
end if.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.

```

```

compute covmat = inv(t(xy22)*xy).
release xy22.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
end if.
do if (dichy = 0).
compute invXtX = inv(t(xy)*xy).
compute coeff =invXtX*t(xy)*datayed(:,2).
do if (effsize=1 and model > 3 and model < 7
and (ncovs > 0 or clsdmy > 0)).
compute xysdy =
{cons,datayed(:,(4+nmeds):ydatacol)}.
compute coeffsd
=inv(t(xysdy)*xysdy)*t(xysdy)*datayed(:,2).
compute resid=datayed(:,2)-(xysdy*coeffsd).
compute sdycov =sqrt(csum(resid&*resid)/(n-
ncol(xysdy))).
compute xvaron=datayed(:,(3+nmeds)).
compute
coeffx2=inv(t(xysdy)*xysdy)*t(xysdy)*xvaron.
compute residx2=xvaron-(xysdy*coeffx2).
compute sdxcov
=sqrt(csum(residx2&*residx2)/(n-ncol(xysdy))).
end if.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4 and bt > 1).
compute resid=datayed(:,2)-xy*coeff.
compute sse = cssq(resid).
compute r2full = 1-(sse/sstot).
end if.
do if (bt =1).
compute resid=data(:,2)-xy*coeff.

```



```

compute k3 = nrow(coeff).
compute sse = cssq(resid).
compute mse = sse/(n-ncol(xy)).
compute
hcnor=coeff&/(sqrt(diag(mse*invXtX))).
do if (hc3 = 1).
compute h = xy(:,1).
loop i3=1 to n.
compute h(i3,1)= xy(i3,:)*invXtX*t(xy(i3,:)).
end loop.
loop i3=1 to k3.
compute xy(:,i3) = (resid(:,ncol(resid))&/(1-
h))*xy(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xy(:,i3) = sqrt(mse)*xy(:,i3).
end loop.
end if.
compute covmat = (invXtX*t(xy)*xy*invXtX).
end if.
end if.
do if (bt = 1).
do if (model = 2).
compute xy2={cons, datayed(:,3:ydatacol)}.
compute temp = ncol(xy2).
do if (temp > 6).
compute xy3=xy2(:,7:temp)}.
end if.
compute xy2={xy2(:,1:3), xy2(:,5)}.
do if (temp > 6).
compute xy2={xy2, xy3}.
release xy3.
end if.
compute invXtX = inv(t(xy2)*xy2).
compute coeff2 =invXtX*t(xy2)*datayed(:,2).
compute ssem2=cssq(datayed(:,2)-xy2*coeff2).
release xy2.
end if.

```

```

compute stderr = sqrt(diag(covmat)).
do if (totlp = 1).
compute
mcmats((nmeds+1):ncol(mcmats),(nmeds+1):n
col(mcmats))=covmat(2:(1+nmeds),2:(1+nmed
s)).
end if.
compute stderr=stderr(1:(nrow(stderr)-
clsdmy),1).
do if (ws=1 and totp=2).
compute stderr(1,1)=stddevy/sqrt(n).
end if.
compute coeffplt = coeff.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*coeff)))/dfnum).
compute coeff=coeff(1:(nrow(coeff)-clsdmy),1).
do if (totlp=1).
compute
mccoeff((nmeds+1):nrow(mccoeff))=coeff(2:(1+
nmeds),1).
end if.
compute bbbb=coeff(2,1).
do if totp = 1.
compute deco(1,1)=2+nmeds.
compute deco = deco(1:decoc,1).
compute covdirt = make((nrow(covmat)-
clsdmy),(ncol(covmat)-clsdmy),0).
compute covdirt = covmat(deco,:).
compute covdir =
make(nrow(covdirt),ncol(covdirt),0).
compute covdir = covdirt(:,t(deco)).
compute deco=coeff(deco,1).
do if (ttt > 0).
compute sedir =
sqrt(diag(directv*covdir*t(directv))).

```

```

compute directv = directv*deco.
end if.
compute sobel(:,3)=coeff(2:(1+nmeds),1).
compute sobel(:,4) = standerr(2:(1+nmeds),1).
compute sobel2 = sobel&*sobel.
do if (varorder <> 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)&*sobel2(:,4)+sobel2(:,3)&*sob
el2(:,2)).
end if.
do if (varorder = 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)&*sobel2(:,4)+sobel2(:,3)&*sob
el2(:,2)+sobel2(:,2)&*sobel2(:,4)).
end if.
compute sobel(:,1) = sobel(:,1)&*sobel(:,3).
compute sobel(:,3) = sobel(:,1)&/sobel(:,2).
compute sobel(:,4) = 2*(1-
cdfnorm(abs(sobel(:,3)))).
end if.
do if (dichy = 0).
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
do if (ws= 1 and totlp=2).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute dfden=n-1.
end if.
compute cnms = {'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute op = {coeff, standerr, tratio, p}.
end if.
do if (dichy = 1).
compute tratio = (coeff&/standerr).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute wald = tratio&*tratio.
compute cnms = {'coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute temp=coeff-abs(xp2)*standerr.
compute op = {coeff, standerr, tratio, p, temp}.
compute temp=coeff+abs(xp2)*standerr.

```

```

compute op = {op, temp}.
end if.
do if (detail = 1).
do if (totlp = 2).
print yname/title = '*****'
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:.'/format = A8.
end if.
do if (totlp <> 2).
print yname/title =
*****
*****'/rlabels = 'Outcome:.'/format
= A8.
end if.
end if.
do if (dichy = 1 and bt = 1 and totp = 1).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:.'/cnames = nmsd/format = F9.2.
end if.
do if (dichy = 0).
compute r2full = 1-(sse/ssty).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute jndf=dfden.
compute wsdf=dfden.
compute xd = abs(xp2).
compute jncrit = (dfden* (exp((dfden-
(5/6))*((xd/(dfden-
(2/3)+(.11/dfden)))*(xd/(dfden-
(2/3)+(.11/dfden)))))-1)).
compute summ = {sqrt(r2full), r2full, mse,
fratio, dfnum, dfden, pfr}.
compute temp1=coeff-sqrt(jncrit)*standerr.
compute temp2=coeff+sqrt(jncrit)*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (detail = 1).
do if (ws <> 1).

```

```

print summ/title = 'Model Summary'/clabels =
'R', 'R-sq', 'MSE' 'F', 'df1', 'df2', 'p'/format =
F10.4.
end if.
end if.
end if.
do if (dichy = 1).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(coeff)-
1)).
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
do if (detail = 1).
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format =
F10.4.
end if.
end if.
do if (totlp = 2).
compute datanmy={'constant';
datanmy((nmeds+2):nrow(datanmy),1)}.
end if.
do if (detail = 1).
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
do if (ws=1).
compute op2=op(1:(1+nmeds),:).
compute datanmy2 = {'c" path'; bpathnam(
(1+(nmeds>1)): ((1+(nmeds>1))+(nmeds-
1)),1)}.
do if (totlp=2).
compute datanmy2={'c path'}.
compute op2=op(1,:).
compute wsdf=n-1.
end if.

```

```

print op2/title = 'Model'/rnames =
datanmy2/cnames = cnms/format = F10.4.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
do if (ws <> 1).
print op/title = 'Model'/rnames =
datanmy/cnames = cnms/format = F10.4.
do if (covcoeff=1).
compute
covmattp=covmat(1:nrow(op),1:nrow(op)).
compute cnamestp=t(datanmy).
print covmattp/title='Covariance matrix of
regression parameter
estimates'/rnames=datanmy/cnames=cnamest
p/format= F10.4.
end if.
end if.
end if.
do if (ttt = 0 and totlp = 1).
compute deco = op((nmeds+2),:).
do if (ws = 1).
compute deco=op(1,:).
end if.
end if.
do if (ttt = 0 and totlp = 2).
compute decotot = op(2,:).
do if (ws = 1).
compute decotot=op(1,:).
end if.
end if.
do if (nmods > 0 and model > 4 and detail = 1
and nrow(yintkey) > 1)).
print yintkey/title = 'Product terms key:'/format
= A8.
end if.
do if (nmods > 0 and model < 4 and detail = 1).
print yintkey2/title = 'Product terms key:'/format
= A8.
do if ((model = 1 or model = 2) and dichy = 0).

```

```

compute temp={((op(4,3)**2)*(1-r2full))/dfden,
op(4,3)**2, 1, dfden, op(4,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(4,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(2,1).
do if (model = 2).
compute temp={temp;((op(6,3)**2)*(1-
r2full))/dfden, op(6,3)**2, 1, dfden, op(6,4)}.
do if (hc3=1).
compute temp(2,1)=((hcnot(6,1)**2)*(1-
r2full))/dfden.
end if.
compute frat2=(dfden*(r2full-(1-
(ssem2/ssty))))/(2*(1-r2full)).
do if (hc3=1).
compute lmat=make(nrow(coeffplt),2,0).
compute lmat(4,1)=1.
compute lmat(6,2)=1.
compute frat2 =
(t(t(lmat)*coeffplt)*inv(t(lmat)*covmat*lmat)*((t(l
mat)*coeffplt)))/2).
end if.
compute temp={temp;(r2full-(1-
(ssem2/ssty))),frat2,2,dfden,1-
fcdf(frat2,2,dfden)}.
compute rnms={rnms; yintkey2(3,1),'Both'}.
end if.
print temp/title = 'R-square increase due to
interaction(s):'/rnames=rnms/clabels = 'R2-
chng', 'F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (model = 3 and dichy = 0).
compute temp={((op(8,3)**2)*(1-r2full))/dfden,
op(8,3)**2, dfden, op(8,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(8,1)**2)*(1-
r2full))/dfden.
end if.

```

```

compute rnms=yintkey2(5,1).
print temp/title = 'R-square increase due to
three-way interaction: '/rnames=rnms/clabels =
'R2-chng', 'F(1,df2)', 'df2', 'p'/format = F10.4.
end if.
end if.
end if.
do if (model = 6 and totlp = 1).
compute
mmpaths(nrow(mmpaths),1)=coeff(nrow(mmpa
ths),1).
compute
mmpaths(nrow(mmpaths),(2:(nmeds+1)))=t(co
eff(2:(nmeds+1),1)).
end if.
do if (totlp = 1).
loop im = 1 to nmeds.
do if (model < 4).
compute ymat(1,im) = coeff(3,1)*(1-yes).
compute ymat(2,im) = coeff(4,1).
compute cmat(1,im) = covmat(3,3).
compute cmat(2,im) = covmat(4,4).
compute cmat(5,im) = covmat(3,4).
compute jnb1=coeff(3,1).
compute jnb3=coeff(4,1).
compute jnsb1=covmat(3,3).
compute jnsb3=covmat(4,4).
compute jnsb1b3=covmat(3,4).
do if (model = 2 or model = 3).
compute ymat(3,im) = coeff(6,1).
compute cmat(3,im) = covmat(6,6).
compute cmat(6,im) = covmat(3,6).
compute cmat(8,im) = covmat(4,6).
end if.
do if (model = 3).
compute ymat(4,im) = coeff(8,1).
compute cmat(4,im) = covmat(8,8).
compute cmat(7,im) = covmat(3,8).
compute cmat(9,im) = covmat(4,8).
compute cmat(10,im) = covmat(6,8).

```



```

compute jnb1=coeff(4,1).
compute jnb3=coeff(8,1).
compute jnsb1=covmat(4,4).
compute jnsb3=covmat(8,8).
compute jnsb1b3=covmat(4,8).
end if.
end if.
do if (model > 3).
compute ymat(5,im) = coeff((1+im),1).
end if.
do if (xmy = 1).
compute ymat(6,im) = coeff((2+nmeds+im),1).
end if.
do if (vy = 1).
compute ymat(6,im) = coeff((3+nmeds+im),1).
end if.
do if (qy = 1 and vy = 1).
compute ymat(6,im) = coeff((5+nmeds+((im-1)*2)),1).
compute ymat(7,im) = coeff((6+nmeds+((im-1)*2)),1).
end if.
do if (vqy = 1).
compute ymat(6,im) = coeff((6+nmeds+((im-1)*3)),1).
compute ymat(7,im) = coeff((7+nmeds+((im-1)*3)),1).
compute ymat(8,im) = coeff((8+nmeds+((im-1)*3)),1).
end if.
do if (wmy = 1).
compute ymat(7,im) = coeff((3+nmeds+im-wy),1).
end if.
do if (wmy = 1 and vy = 1).
compute ymat(7,im) =coeff((4+(nmeds*2)+im-wy),1).
end if.
do if (wmy = 1 and vy = 1 and wvmy = 1).

```

```

compute ymat(7,im) =
coeff((6+(nmeds*2)+((im-1)*2)-wy),1).
compute ymat(8,im) =
coeff((7+(nmeds*2)+((im-1)*2)-wy),1).
end if.
do if (wmy = 1 and zmy = 1).
compute ymat(6,im) = coeff((6-(wzy*3)+(wzm-
1)+(nmeds*2)+((im-1)*2)*wzm)+((im-1)*(1-
wzm))-((zy-wzy)*2),1).
compute ymat(7,im) = coeff((3-
zy+im+nmeds),1).
do if (wzmy = 1).
compute ymat(8,im) = coeff((7-
(wzy*3)+(nmeds*2)+((im-1)*2)),1).
end if.
end if.
do if (nmods > 0 and model <> 5).
loop indlp = 1 to nrow(modvals).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp)).
do if (model > 6).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp))*csum(ym
at(5:8,im)*vmat(5:8,indlp)).
end if.
end loop.
compute indboot((bt+(im-
1)*(boot+1)),:)=t(indeff).
do if (imm = 1 or imm = 3).
compute
indbootp(bt,im)=ymat(2,im)*ymat(5,im).
do if (imm=3).
compute
indbootp(bt,(im+nmeds))=ymat(3,im)*ymat(5,i
m).
do if (zdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxz-cminz).
end if.

```

```

end if.
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
end if.
end if.
do if ((imm= 2 or imm = 4) or model = 74).
compute
indbootp(bt,im)=ymat(1,im)*ymat(6,im).
do if (imm=4).
compute
indbootp(bt,(im+nmeds))=ymat(1,im)*ymat(7,i
m).
do if (qdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxq-cminq).
end if.
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
end if.
end if.
do if (imm=7).
compute
indbootp(bt,im)=ymat(2,im)*ymat(6,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(2,im)*y
mat(6,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(6,im)*onesmv1)+(ymat(2,im)*y
mat(6,im)*t(immmv1)).
do if (wdich=1).

```

```

compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxv-cminv).
end if.
end if.
do if (imm=5).
compute
indbootp(bt,im)=ymat(4,im)*ymat(5,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(4,im)*y
mat(5,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(3,im)*ymat(5,im)*onesmv1)+(ymat(4,im)*y
mat(5,im)*t(immmv1)).
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).

```

```

end if.
do if (zdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxz-
cminz).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxz-cminz).
end if.
end if.
do if (imm=6).
compute
indbootp(bt,im)=ymat(1,im)*ymat(8,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(1,im)*ymat(6,im)*onesmv2)+(ymat(1,im)*y
mat(8,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(7,im)*onesmv1)+(ymat(1,im)*y
mat(8,im)*t(immmv1)).
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxv-cminv).
end if.
do if (qdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxq-
cminq).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-

```

```

1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxq-cminq).
end if.
end if.
do if ((model = 58 or model = 59) and
wvdich=1).
compute
indbootp(bt,im)=(ymat(1,im)*ymat(7,im)*(cmax
w-cminw))+(ymat(2,im)*ymat(5,im)*(cmaxw-
cminw))+(ymat(2,im)*ymat(7,im)*((cmaxw*cma
xw)-(cminw*cminw))).
end if.
end if.
do if (model = 4 or model = 5).
compute
indboot(bt,im)=csum(ymat(1:4,im)*vmat(1:4,1
))*csum(ymat(5:8,im)*vmat(5:8,1)).
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.000000000000001.
end if.
compute sumind=rsum(indboot(bt,:)).
do if (im=nmeds).
compute
pmeff(bt,2:(im+1))=indboot(bt,:)/(sumind+coeff(
(2+nmeds),1)).
end if.
compute
rmeff(bt,(im+1))=indboot(bt,im)/coeff((2+nmeds
),1).
compute
abpseff(bt,(im+1))=indboot(bt,im)/stddevy.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*stddevx.
do if (ncovs > 0 or clsdmy > 0).
compute
abpseff(bt,(im+1))=indboot(bt,im)/sdy cov.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*sdx cov.
end if.

```

```

do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,1) = r2my-(r2full-r2xy).
compute abmmr = 1.
do if (indboot(bt,im) < 0).
compute abmmr = cmin(abmm).
end if.
do if (indboot(bt,im) > 0).
compute abmmr = cmax(abmm).
end if.
compute kappa2(bt,1)=indboot(bt,im)/abmmr.
compute tmp = indboot(bt,im)/abmmr.
end if.
end if.
end if.
do if (model = 6).
do if (nmeds = 2).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(4,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3).
compute
indboot(bt,3)=mmpaths(3,1)*mmpaths(4,3).
end if.
do if (nmeds = 3).
compute indboot(bt,1) =
mmpaths(2,1)*mmpaths(5,2).
compute indboot(bt,2) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3).
compute indboot(bt,3) =
mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4).
compute indboot(bt,4) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4).
compute indboot(bt,5) =
mmpaths(3,1)*mmpaths(5,3).
compute indboot(bt,6) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4).

```

```

compute indboot(bt,7) =
mmpaths(4,1)*mmpaths(5,4).
end if.
do if (nmeds = 4).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(6,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(6,3).
compute
indboot(bt,3)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(6,4).
compute
indboot(bt,4)=mmpaths(2,1)*mmpaths(5,2)*mmpaths(6,5).
compute
indboot(bt,5)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(6,4).
compute
indboot(bt,6)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3)*mmpaths(6,5).
compute
indboot(bt,7)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,8)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,9)=mmpaths(3,1)*mmpaths(6,3).
compute
indboot(bt,10)=mmpaths(3,1)*mmpaths(4,3)*mmpaths(6,4).
compute indboot(bt,11)
=mmpaths(3,1)*mmpaths(5,3)*mmpaths(6,5).
compute indboot(bt,12) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,13) =
mmpaths(4,1)*mmpaths(6,4).

```



```

compute indboot(bt,14) =
mmpaths(4,1)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,15) =
mmpaths(5,1)*mmpaths(6,5).
end if.
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.00000000000001.
end if.
compute
pmeff(bt,:)=indboot(bt,:)/(rsum(indboot(bt,:))+m
mpaths(nrow(mmpaths),1)).
compute rmeff(bt,:) =
indboot(bt,:)/mmpaths(nrow(mmpaths),1).
compute abpseff(bt,:)=indboot(bt,:)/stddevy.
compute
abcseff(bt,:)=(stddevx*indboot(bt,:))/stddevy.
do if (ncovs > 0 or clsdmy > 0).
compute abpseff(bt,:)=indboot(bt,:)/sdy cov.
compute
abcseff(bt,:)=(sdx cov*indboot(bt,:))/sdy cov.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,:) = r2my-(r2full-r2xy).
end if.
end if.
end if.
end loop.
end if.
end loop.
do if (savboot = 1 and bt > 1).
compute bootstrp((bt-1),:)=bootcoef.
end if.
end loop.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.

```

```

release datayed, dat, datamed, data, y, x,
datatm, dataty, mints2, cons, mnv, tmp.
do if (mc > 0).
compute x1 = sqrt(-
2*ln(uniform(mc,nrow(mcmats))))&*cos((2*3.14
159265358979)*uniform(mc,nrow(mcmats))).
compute x1=x1*chol(mcmats).
loop i=1 to nrow(x1).
compute x1(i,:)=x1(i,:)+t(mccoeff).
end loop.
loop i = 1 to nmeds.
compute
indboot(2:nrow(indboot),i)=x1(:,i)&*x1(:,(i+nme
ds)).
end loop.
end if.
do if (ttt = 0 and model > 3).
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
end if.
do if (model <> 74).
do if (dichy = 0).
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 't', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 't', 'p', 'LLCI', 'ULCI'/format =
F10.4.
else.
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 'Z', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.

```

```

print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 'Z', 'p', 'LLCI', 'ULCI' /format =
F10.4.
end if.
end if.
end if.
do if (ttt > 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
compute clbs = {modvnm2, 'Effect', 'SE', 't', 'p',
'LLCI', 'ULCI'}.
compute tratio = (directv&/sedir).
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy))))).
compute outp = {modvalsd, directv, sedir,
tratio, p}.
do if (dichy = 0).
compute temp1=directv-sqrt(jncrit)*sedir.
compute temp2=directv+sqrt(jncrit)*sedir.
compute outp = {outp, temp1, temp2}.
end if.
do if (dichy = 1).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute temp = directv-abs(xp2)*sedir.
compute outp = {outp, temp}.
compute temp = directv+abs(xp2)*sedir.
compute outp = {outp, temp}.
compute clbs = {modvnm2, 'Effect', 'SE', 'Z', 'p',
'LLCI', 'ULCI'}.
end if.
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
print outp/title = 'Conditional direct effect(s) of
X on Y at values of the moderator(s):'/format =
F10.4 /cnames = clbs.
end if.
do if (nmods > 0 and model <> 5).
do if (model < 4).

```

```

print/title =
*****
*****!

compute zmat(1,1) = 1*(1-yes).
compute cfse = make(nrow(modvals),1,0).
loop #m = 1 to nrow(modvals).
do if (model = 1).
compute zmat(2,1)=modvals(#m,1)**2.
compute zmat(5,1)=2*modvals(#m,1).
end if.
do if (model = 2 or model = 3).
compute zmat(2,1)=modvals(#m,2)**2.
compute zmat(3,1)=modvals(#m,1)**2.
compute
zmat(4,1)=(modvals(#m,1)**2)*(modvals(#m,2)
**2)*(1-yes2).
compute zmat(5,1)=2*modvals(#m,2)*(1-yes).
compute zmat(6,1)=2*modvals(#m,1)*(1-yes).
compute
zmat(7,1)=2*modvals(#m,1)*modvals(#m,2)*(1-
yes).
compute
zmat(8,1)=2*modvals(#m,1)*modvals(#m,2).
compute
zmat(9,1)=2*modvals(#m,1)*(modvals(#m,2)**
2)*(1-yes2).
compute
zmat(10,1)=2*(modvals(#m,1)**2)*modvals(#m
,2)*(1-yes2).
end if.
compute cfse(#m,1)=sqrt(csum(zmat&*cmat)).
end loop.
end if.
do if (nmods > 0).
compute clbs = {modvnm, 'Effect'}.
loop im = 1 to nmeds.
compute obs = t(indboot(1+(im-1)*(boot+1),:)).
compute outp = {modvals, obs}.
do if (model < 4).
compute tstat = obs&/cfse.

```

```

do if (dichy = 0).
compute pval = 2*(1-tcdf(abs(tstat), (n-
ncol(xy)))).
compute temp=obs-sqrt(jncrit)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp=obs+sqrt(jncrit)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 't', 'p', 'LLCI', 'ULCI'}.
compute jnclbs=clbs.
end if.
do if (dichy = 1).
compute pval = 2*(1-cdfnorm(abs(tstat))).
compute temp = obs-abs(xp2)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp = obs+abs(xp2)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute jnclbs=clbs.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indboot((1+(im-
1)*(boot+1)),:).
compute indboot2 = indboot((2+(im-
1)*(boot+1)):(1+(im-1)*(boot+1)+boot),:).
compute mnind = t(csum(indboot2)/boot).
compute stdind=t(sqrt((cssq(indboot2)-
((csum(indboot2)**2)/boot))/(boot-1))).
compute llci=make(1,ncol(indboot2),-999).
compute ulci=make(1,ncol(indboot2),-999).
loop #e = 1 to ncol(indboot2).
.
compute temp = indboot2(:,#e).
compute temp(GRADE( indboot2(:,#e) )) =
indboot2(:,#e).
compute badlo = 0.
compute badhi = 0.

```

```

do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.

```

```

do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {modvals, obs, stdind, t(llci),
t(ulci)}.
compute clbs = {modvnm, 'Effect', 'Boot SE',
'BootLLCI', 'BootULCI'}.
end if.
compute mtemp = mnames(1, im).
compute rlbs =
make(nrow(modvals),1,mnames(1,im)).
do if (model < 4).
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
do if (yes=0).
print outp/title = 'Conditional effect of X on Y at
values of the moderator(s):'/cnames =
clbs/format = F10.4.
else.
compute outp=outp(:,3:ncol(outp)).
compute clbs=clbs(:,3:ncol(clbs)).
compute clbs(1,1)='Contrast'.
print outp/title = 'Contrast of conditional effects
of X on Y'/cnames = clbs/format = F10.4.
end if.
end if.
do if (model > 5 and mod74dic <> 1).
do if (im = 1).
print/title = 'Conditional indirect effect(s) of X on
Y at values of the moderator(s):'.
end if.
print outp/title = 'Mediator'/rnames =
rlbs/cnames = clbs/format = F10.4.

```

```

else if (mod74dic = 1 and model = 74).
do if (im = 1).
print/title = 'Indirect effect(s) of X on Y:'.
compute clbs3=clbs(1,2:ncol(clbs)).
end if.
compute outp3=outp(1,2:ncol(outp)).
print outp3/title = 'Mediator'/rnames =
rlbs/cnames = clbs3/format = F10.4.
end if.
end loop.
loop i = notes to 1 by -1.
do if (note(i,1) = 4 and yes=0).
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 5 and yes=0).
print/title = 'Values for quantitative moderators
are the mean and plus/minus one SD from
mean.'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 14 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    below the mean was replaced
with the minimum because one SD below the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.
do if (note(i,1) = 15 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    above the mean was replaced
with the maximum because one SD above the
mean'/space=0.

```



```

print/title = '      is outside of the range of the
data.'/space=0.
end if.
end loop.
do if (model = 3 and yes=0).
compute jnvals=make(nrow(matw),7,0).
compute jnvals(:,1)=matw.
compute jnvals(:,2)=jnb1+jnb3*jnvals(:,1).
compute
jnvals(:,3)=sqrt(jnsb1+2*jnvals(:,1)*jnsb1b3+(jn
vals(:,1)*jnvals(:,1))*jnsb3).
compute jnvals(:,4)=jnvals(:,2)/jnvals(:,3).
do if (dichy = 0).
compute jnvals(:,5)=2*(1-tcdf(abs(jnvals(:,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(:,5)=2*(1-
cdfnorm(abs(jnvals(:,4)))).
end if.
compute jnvals(:,6)=jnvals(:,2)-
sqrt(jncrit)*jnvals(:,3).
compute
jnvals(:,7)=jnvals(:,2)+sqrt(jncrit)*jnvals(:,3).
compute clbs={clbs(:,1),clbs(:,3:ncol(clbs))}.
do if (coeffci = 0).
compute jnvals =jnvals(:,1:(ncol(jnvals)-2)).
end if.
print jnvals/title = 'Conditional effect of X*M
interaction at values of W: '/cnames
=clbs/format = F10.4.
end if.
do if (jn = 1 and (model = 1 or model = 3) and
jndich = 0 and yes=0).
print/title = '***** JOHNSON-
NEYMAN TECHNIQUE
*****'.
compute ajn =(jncrit*jnsb3)-(jnb3*jnb3).
compute bjn = 2*((jncrit*jnsb1b3)-(jnb1*jnb3)).
compute cjn = (jncrit*jnsb1)-(jnb1*jnb1).

```

```

compute radarg = (bjn*bjn)-(4*ajn*cjn).
compute den = 2*ajn.
compute nrts = 0.
do if (radarg >= 0 and den <> 0).
compute x21 = (-bjn+sqrt(radarg))/den.
compute x22 = (-bjn-sqrt(radarg))/den.
compute roots = 0.
do if (x21 >= jnmin and x21 <= jnmax).
compute nrts = 1.
compute roots = {roots; x21}.
end if.
do if (x22 >= jnmin and x22 <= jnmax).
compute nrts = nrts + 1.
compute roots = {roots; x22}.
end if.
compute roots={roots,make(nrow(roots),2,0)}.
compute modtemp=m.
do if (model=3).
compute modtemp=w.
end if.
do if (nrts > 0).
compute roots = roots(2:nrow(roots),1:3).
compute roots(1,2)=(csum(modtemp <
roots(1,1))/n)*100.
compute roots(1,3)=(csum(modtemp >
roots(1,1))/n)*100.
do if (nrow(roots)=2).
compute roots(2,2)=(csum(modtemp <
roots(2,1))/n)*100.
compute roots(2,3)=(csum(modtemp >
roots(2,1))/n)*100.
end if.
print roots/title = 'Moderator value(s) defining
Johnson-Neyman significance
region(s)'/clabels = 'Value', '% below', '%
above'/format = F10.4.
compute jnvals=make((21+nrts),7,0).
loop i= 0 to 20.
compute jnvals((i+1),1)=jnmin+(i*((jnmax-
jnmin)/20)).

```

```

end loop.
loop i = 1 to nrts.
loop j = 2 to nrow(jnvals).
do if ((roots(i,1) > jnvals((j-1),1)) and (roots(i,1)
< jnvals(j,1))).
compute
jnvals((j+1):(21+i),1)=jnvals(j:(20+i),1).
compute jnvals(j,1)=roots(i,1).
end if.
end loop.
end loop.
loop i = 1 to nrow(jnvals).
compute jnvals(i,2)=jnb1+jnb3*jnvals(i,1).
compute
jnvals(i,3)=sqrt(jnsb1+2*jnvals(i,1)*jnsb1b3+(jn
vals(i,1)*jnvals(i,1))*jnsb3).
compute jnvals(i,4)=jnvals(i,2)/jnvals(i,3).
do if (dichy = 0).
compute jnvals(i,5)=2*(1-tcdf(abs(jnvals(i,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(i,5)=2*(1-
cdfnorm(abs(jnvals(i,4)))).
end if.
compute jnvals(i,6)=jnvals(i,2)-
sqrt(jncrit)*jnvals(i,3).
compute
jnvals(i,7)=jnvals(i,2)+sqrt(jncrit)*jnvals(i,3).
end loop.
do if (model = 1).
print jnvals/title = 'Conditional effect of X on Y
at values of the moderator (M)'/cnames
=jnclbs/format = F10.4.
end if.
do if (model = 3).
compute
jnclbs={jnclbs(:,1),jnclbs(:,3:ncol(jnclbs))}.

```

```

print jnvals/title = 'Conditional effect of X*M on
Y at values of the moderator (W)'/cnames
=jnclbs/format = F10.4.
end if.
end if.
do if (nrts = 0).
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
else.
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
end if.
end if.
do if (model < 4 and (plot = 1)).
compute dataplot =
make((nrow(modvals)*nrow(matx)),(ncol(modv
als)+1),0).
compute tmp = 1.
loop i = 1 to nrow(modvals).
loop j = 1 to nrow(matx).
compute dataplot(tmp,.)={matx(j,1),
modvals(i,:)}.
compute tmp=tmp+1.
end loop.
end loop.
compute dataplot = {dataplot,
make(nrow(dataplot),(1+dichy),0)}.
compute dataplo2 = make(nrow(dataplot),1,1).
do if (model = 1).
compute dataplo2 = {dataplo2, dataplot(:,2),
dataplot(:,1), (dataplot(:,1)*dataplot(:,2))}.
end if.
do if (model = 2 or model = 3).
compute dataplo2 = {dataplo2, dataplot(:,3),
dataplot(:,1), (dataplot(:,1)*dataplot(:,3)),
dataplot(:,2), (dataplot(:,1)*dataplot(:,2))}.

```

```

do if (model = 3).
compute dataplo2 = {dataplo2,
(dataplot(:,2)*dataplot(:,3)),
(dataplot(:,1)*dataplot(:,2)*dataplot(:,3))}.
end if.
end if.
loop i = 1 to nrow(dataplot).
compute tmp=dataplo2(i,:).
do if (ncovs > 0).
compute tmp = {tmp, covmeans}.
end if.
do if (cluster > 0).
compute tmp = {tmp, cldmeans}.
end if.
compute dataplot(i,(ncol(dataplot)-
(dichy)))=tmp*coeffplt.
do if (dichy = 1).
compute
dataplot(i,(ncol(dataplot)))=exp(tmp*coeffplt)/(1
+exp(tmp*coeffplt)).
end if.
end loop.
compute clbs = {xname, modvnm, 'yhat'}.
do if (dichy = 1).
compute clbs = {xname, modvnm, 'ln(odds)',
'prob'}.
end if.
print/title =
*****
*****!

```

```

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (model = 1).
do if (dichy = 0).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Eco_Innovation
Financial_Performance.' /space=1.
else if (dichy = 1).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Eco_Innovation
logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH Financial_Performance BY
Eco_Innovation.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Financial_Performance BY
Plan_Orga_Pract.' /space=0.
end if.
end if.

```

```

do if (dichy = 1).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH logodds BY Eco_Innovation.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH prob BY Eco_Innovation.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Plan_Orga_Pract.'
/space=0.
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Plan_Orga_Pract.' /space=0.
end if.
end if.
end if.
do if (model = 2 or model = 3).
do if (dichy = 0).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Eco_Innovation
Financial_Performance.' /space=1.
end if.
do if (dichy = 1).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Eco_Innovation
logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH Financial_Performance BY
Eco_Innovation/PANEL ROWVAR=' /space=0.

```

```

end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Financial_Performance BY
Plan_Orga_Pract/PANEL ROWVAR=.'
/space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH logodds BY Eco_Innovation/PANEL
ROWVAR=.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH prob BY Eco_Innovation/PANEL
ROWVAR=.' /space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Plan_Orga_Pract/PANEL
ROWVAR=.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Plan_Orga_Pract/PANEL
ROWVAR=.' /space=0.
end if.
end if.
end if.
do if (ncovs > 0).
print/title = '* Estimates are based on setting
covariates to their sample means.'.
end if.
end if.
end if.

```



```

do if ((imm > 0) or (model = 74 and
mod74dic=0) or ((model = 58 or model = 59)
and wvdich=1))).
do if (imm > 4 and imm < 8).
compute
indbootp={indbootp(:,1:nmeds),indboop1,indbo
op2}.
end if.
compute obsprod = t(indbootp(1,:)).
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indbootp(1,:).
compute indbootp = indbootp(2:(boot+1),:).
compute mnindp = t(csum(indbootp)/boot).
compute stdindp=t(sqrt((cssq(indbootp)-
((csum(indbootp)**2)/boot))/(boot-1))).
compute llcip=make(1,ncol(indbootp),-999).
compute ulcip=make(1,ncol(indbootp),-999).
loop #e = 1 to ncol(indbootp).

.
compute temp = indbootp(:,#e).
compute temp(GRADE( indbootp(:,#e) )) =
indbootp(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).

```

```

compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcip(1,#e)=llcit.
compute ulcip(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {obsprod, stdindp, t(llcip),
t(ulcip)}.
compute clbs = {'Effect', 'SE(Boot)', 'BootLLCI',
'BootULCI'}.
do if (model = 8 or model = 12).

```

```

print/title='-----'.
compute outpimm=outp(1:nmeds,:).
print/title = 'Indirect effect of highest order
product.'/space=0.
print outpimm/title = 'Mediator'/cnames =
clbs/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
compute clbs3 = {'Index', 'SE(Boot)',
'BootLLCI', 'BootULCI'}.
do if (imm < 5).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator.'/format=A8.
end if.
compute outpimm=outp((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
do if (imm > 4 and imm < 8).

```

```

print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'.

compute outpimm=outp(1:nmeds,:).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
compute
outpimm=outp((nmeds+1):nrow(outp),:).
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.

loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),clbs3}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3(1,1)=modvnm(1,1).
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.

```

```

print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
end if.
do if (boot = 0).
do if (model = 8 or model = 12).
print/title='-----'.
compute obsimm=obsprod(1:nmeds,:).
print/title = 'Indirect effect of highest order
product.'/space=0.
print obsimm/title = 'Mediator'/clabels =
'Effect'/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
do if ((imm > 0 and imm < 5) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=obsprod((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
end loop.
end if.

```

```

do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'
compute outpimm=obsprod(1:nmeds,1).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
compute
outpimm=obsprod((nmeds+1):nrow(obsprod),:)
.
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),'Index'}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3={modvnm(1,1),'Index'}.
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator: '/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
end if.
end if.

```

```

end if.
compute conmake=0.
compute concols = 0.
do if ((model > 3 and model < 7) and (contrast
> 0) and nmods = 0 and nmeds > 1 ).
compute concols =
((ncol(indboot)*(ncol(indboot)-1))/2).
compute indcon=make(nrow(indboot),concols,-
999).
compute conkey = { ' ', ' ', ' ', ' ' }.
compute temp=1.
compute conmake=1.
loop i = 1 to (ncol(indboot)-1).
loop j = (i+1) to (ncol(indboot)).
compute indcon(:,temp)=indboot(:,i)-
indboot(:,j).
do if (contrast = 2).
compute indcon(:,temp)=abs(indboot(:,i))-
abs(indboot(:,j)).
end if.
do if (model <= 6).
compute conkey={conkey; mnames(1,i),
'minus', mnames(1,j)}.
end if.
do if (model = 6).
compute conkey={conkey; indlbl2(i,1), 'minus',
indlbl2(j,1)}.
end if.
compute temp=temp+1.
end loop.
end loop.
end if.
do if (model = 4 or model = 5).
compute clbs = {'Effect'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obs = {csum(obs); obs}.
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.

```

```

compute rlbs = {rlbs;
cntname(1:ncol(indcon),1)}.
end if.
compute outp = obs.
compute outp2=outp.
do if (effsize = 1 and dichy = 0).
compute
pmeff(:,1)=rsum(pmeff(:,2:ncol(pmeff))).
compute
rmeff(:,1)=rsum(rmeff(:,2:ncol(rmeff))).
compute
abpseff(:,1)=rsum(abpseff(:,2:ncol(abpseff))).
compute
abcseff(:,1)=rsum(abcseff(:,2:ncol(abcseff))).
compute eff = {pmeff, rmeff, abpseff, abcseff}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute eff = {eff, r245}.
compute r245obs = {r245(1,1);r245(1,1)}.
compute kappa2ob = kappa2(1,1).
end if.
compute pmobs = t(pmeff(1,1:(nmeds+1))).
compute rmobs = t(rmeff(1,1:(nmeds+1))).
compute psobs = t(abpseff(1,1:(nmeds+1))).
compute csobs = t(abcseff(1,1:(nmeds+1))).
do if (contrast = 0).
compute outp2 = {obs, psobs, csobs, pmobs,
rmobs}.
end if.
do if (contrast > 0).
compute obs2=obs(1:nrow(psobs),:).
compute outp2 = {obs2, psobs, csobs, pmobs,
rmobs}.
end if.
compute clbs = {'ab', 'ab_ps', 'ab_cs', 'ab/c',
'ab/c'}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute outp2 = {outp2, r245obs}.
compute clbs = {clbs, 'R-sq_med'}.

```



```

do if (boot = 0 and mc = 0).
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
do if (boot = 0 and mc = 0).
do if (nmeds = 1).
compute outp2 = outp2(2,:).
compute rlbs = rlbs(2,1).
end if.
do if (model=5).
compute outp2=outp2(:,1:3).
end if.
print outp2/title = 'Indirect effect(s) of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1 and nmeds >
1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0 or mc > 0).
compute indboot = {rsum(indboot), indboot}.
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
compute ones = make(bootsz,1,1).
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(bootsz+1),:).
compute mnind = t(csum(indboot)/bootsz).

```

```

compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/bootSZ))/(bootSZ-1))).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).

.

compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.

```

```

end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llici(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
do if (effsize = 1 and dichy = 0 and boot > 0).
compute estmte=eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).
.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.

```

```

compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).

```

```

compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
end if.
end if.
do if (boot > 0 or mc > 0).
compute outp = {obs, stdind, t(llci), t(ulci)}.
do if (nmeds = 1).
compute outp = outp(2,:).
compute rlbs = rlbs(2,1).
end if.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
do if (mc > 0).
compute clbs = {'Effect', 'MC SE', 'MC LLCI',
'MC ULCI'}.
end if.
print outp/title = 'Indirect effect of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (dichy = 0 and effsize = 1 and boot > 0).
compute outp = {psobs,
stdindf((2*(nmeds+1)+1):(3*(nmeds+1)),1),
t(llcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))),
t(ulcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Partially standardized indirect
effect of X on Y'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute outp = {csobs,
stdindf((3*(nmeds+1)+1):(4*(nmeds+1)),1),
t(llcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))),
t(ulcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.

```

```

print outp/title = 'Completely standardized
indirect effect of X on Y'/rnames = rlbs/cnames
= clbs/format = F10.4.
compute outp = {pmobs,
stdindf(1:(nmeds+1),1),t(llcif(1,1:(nmeds+1))),
t(ulcif(1,1:(nmeds+1))) }.
do if (model = 4).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
compute outp = {rmobs, stdindf(((nmeds+1)+1)
:(2*(nmeds+1)) ,1),
t(llcif(1,((nmeds+1)+1):(2*(nmeds+1)))),
t(ulcif(1,((nmeds+1)+1):(2*(nmeds+1))))}.
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
do if (nmeds = 1 and cluster = 0 and ncovs = 0
and model = 4).
compute r245obs = r245obs(1,1).
compute outp = {r245obs,
stdindf((4*(nmeds+1)+1):(4*(nmeds+1)+1),1),
t(llcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1))),
t(ulcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1)))}.
print outp/title = 'R-squared mediation effect
size (R-sq_med)'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
end if.
do if (normal = 1).

```

```

do if (nmeds = 1).
print sobel/title = 'Normal theory tests for
indirect effect'/clabels = 'Effect', 'se', 'Z',
'p'/format = F10.4.
end if.
do if (nmeds > 1).
compute rlbs2 = rlbs(2:nrow(rlbs),1).
print sobel/title = 'Normal theory tests for
specific indirect effects'/rnames = rlbs2/clabels
= 'Effect', 'se', 'Z', 'p'/format = F10.4.
end if.
end if.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
do if (model = 6).
compute clbs = {'eff'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obsfsum=csum(obs).
compute obs = {obsfsum; obs}.
compute indlbl = indlbl(1:nrow(obs),1).
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute indlbl = {indlbl;
cntname(1:ncol(indcon),1)}.
end if.
compute obs2=obs.
do if (boot = 0).
do if (dichy = 0 and effsize = 1).
compute clbs = {'eff', 'eff_ps', 'eff_cs', 'eff/c',
'eff/c'}.
do if (ncovs = 0).

```

```

compute obs = {obs, obs/stddevy,
(obs*stddevx/stddevy), obs/ctot,
obs/mmpaths(nrow(mmpaths),1)}.
end if.
do if (ncovs > 0 or clsdmy > 0).
compute obs = {obs, obs/sdycov,
(obs*sdxcov/sdycov),
obs/(obsfsum+mmpaths(nrow(mmpaths),1)),
obs/mmpaths(nrow(mmpaths),1)}.
end if.
compute obs2=obs.
do if (contrast > 0).
compute obs2 = obs(1:(nrow(obs)-concols),:).
end if.
end if.
print obs2/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute indboot = {rsum(indboot), indboot}.
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(boot+1),:).
compute mnind = t(csum(indboot)/boot).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/boot))/(boot-1))).
compute temp = nrow(indboot).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).

```



```

loop #e = 1 to ncol(indboot).

.

compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.

```

```

do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute obs = {obs, stdind, t(llci), t(ulci)}.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
print obs/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (effsize = 1 and dichy = 0).
compute indboot=indboot(:,1:(ncol(indboot)-
concols)).
compute eff = {rsum(abpseff), abpseff,
rsum(abcseff), abcseff, rsum(pmeff), pmeff,
rsum(rmeff), rmeff}.
compute effobs = eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).
.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).

```

```

compute badlo = 0.
compute badhi = 0.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(effobs(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.

```

```

compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute temp2 = stdindf(1:ncol(indboot),1).
compute temp3 = effobs(:,1:ncol(indboot)).
compute templo = llcif(1,1:ncol(indboot)).
compute temphi = ulcif(1,1:ncol(indboot)).
compute outp = {t(temp3), temp2,t(templo),
t(temphi)}.
print outp/title = 'Partially standardized indirect
effect of X on Y'/cnames = clbs/rnames =
indlbl/format = F10.4.
compute temp2 =
stdindf((ncol(indboot)+1):(2*ncol(indboot)),1).
compute temp3 =
effobs(:,(ncol(indboot)+1):(2*ncol(indboot))).
compute templo =
llcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute temphi =
ulcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templo),
t(temphi)}.
print outp/title = 'Completely standardized
indirect effect of X on Y'/cnames = clbs/rnames
= indlbl/format = F10.4.
compute temp2 =
stdindf((2*(ncol(indboot))+1):(3*ncol(indboot)),1
).
compute temp3 =
effobs(:,(2*(ncol(indboot))+1):(3*ncol(indboot)))
.

```

```

compute templow =
  llcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute temphi =
  ulcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
  t(temphi)}.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
compute temp =
  eff(:,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temp2 =
  stdindf((3*(ncol(indboot))+1):(4*ncol(indboot)),1
  ).
compute temp3 =
  effobs(:,(3*(ncol(indboot))+1):(4*ncol(indboot)))
  .
compute templow =
  llcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temphi =
  ulcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
  t(temphi)}.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
end if.
end if.
do if (nmeds = 2).
compute effkey = {xname, '->', mnames(1,1), '-
->', yname, '', ''}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, '', ''}.
compute effkey = {indlbl(2:4,1), effkey}.
end if.
do if (nmeds = 3).
compute effkey = {xname, '->' , mnames(1,1), '-
->', yname, '', '', ''}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, '',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, '',' ',' ',' '}.
compute effkey = {indbl(2:8,1), effkey}.
end if.
do if (nmeds = 4).
compute effkey = {xname, '->' , mnames(1,1), '-
>', yname, '',' ',' ',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,4), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname, '',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,4), '->', yname, '',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, '',' '}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', mnames(1,4), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,4), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', mnames(1,4), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,4), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {indl(2:16,1), effkey}.
end if.
print effkey/title = 'Indirect effect key'/format =
A8.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
end if.
DO IF (medte2=1).
compute
name=t(mdlms2(2:nrow(mdlms2),1)).
compute tmpdat=datayed(:,2:ncol(datayed)).
compute dd=tmpdat(:,1).

```

```

compute
dd={dd,tmpdat(:,(2+nmeds)),tmpdat(:,2:(1+nmeds))}.
compute stratify=0.
do if (ncovs > 0).
compute cname = cnames.
compute
dd={dd,tmpdat(:,(3+nmeds):(2+nmeds+ncovs))}.
end if.
compute method = 1.
compute samples = boot.
do if (mc > 0).
compute samples = mc.
compute method=2.
end if.
compute omnibus = 1.
compute booterr = 0.
do if (mcx < 0 or mcx > 4).
compute mcx = 1.
end if.
compute ovals = ncol(design(dd(:,1))).
compute nte = notes.
compute criterr = 0.
compute xskip = 0.
compute dumok = 0.
compute badend = 0.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute priorlo = -9999999.
compute priorhi = 9999999.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute reps = samples.
do if (contrast = 1).
compute note(nte,1) = 26.
compute nte=nte+1.

```



```

end if.
do if (effsize = 1).
compute note(nte,1) = 27.
compute nte=nte+1.
end if.
do if (normal = 1).
compute note(nte,1) = 28.
compute nte=nte+1.
end if.
do if (method = 2).
do if (omnibus = 1).
compute note(nte,1) = 25.
compute nte=nte+1.
end if.
compute samples = 0.
do if (reps < 1000).
compute reps = 1000.
end if.
end if.
do if (method = 1).
compute samples = reps.
end if.
compute temp = ncol(dd).
compute n = nrow(dd).
compute ones = make(n,1,1).
compute temp=dd(:,2).
compute nxd = ncol(temp).
compute nx = nxd.
do if (nx = 1 and mcx > 0).
compute temp = dd.
compute temp(GRADE(dd(:,2)),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,2)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute mnvls = cmin(t(nnvls)).
do if (mnvls < 2).
compute errs = errs + 1.
compute runerrs(errs,1) = 37.
compute criterr = 1.

```

```

end if.
do if (nvls > 9).
compute errs = errs+1.
compute runerrs(errs,1) = 36.
compute criterr = 1.
end if.
do if (criterr = 0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,2).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,2) <> nnvls((temp-1),1)).
compute nnvls(temp,1)=dd(i,2).
compute temp = temp+1.
end if.
end loop.
do if (mcx > 0).
compute x = dummy(:,2:ncol(dummy)).
compute nx = ncol(x).
compute minus1 = make(1,ncol(x),-1).
compute xdes=make((nx+1),3,0).
compute xdes(1,1)=dd(1,2).
compute xdes(1,2)=1.
compute temp = 2.
loop k = 2 to n.
do if (dd(k,2) <> dd((k-1),2)).
compute xdes(temp,2) = k.
compute xdes(temp,1) = dd(k,2).
compute xdes((temp-1),3) = k-1.
compute temp=temp+1.
end if.
end loop.
compute xdes((temp-1),3)=n.
compute xdes = {xdes, (xdes(:,3)-xdes(:,2)+1)}.
do if (stratify <> 0).
compute note(nte,1) = 7.
compute nte=nte+1.
end if.
do if (mcx = 4).

```

```

loop k = 1 to n.
do if (rsum(x(k,:)) = 0).
compute x(k,:) = minus1.
end if.
end loop.
end if.
do if (mcx = 2 or mcx = 3).
compute note(nte,1)=4.
compute nte = nte+1.
loop k = 1 to n.
do if (rsum(x(k,:)) > 0).
loop i = 1 to ncol(x).
do if (x(k,i) = 0).
compute x(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcx = 3).
compute conmat1={-8,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1; 0,0,0,0,0,-
3,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.
loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,:)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute x(k,:)=conmat1((rsum(x(k,:))+1),:).
end loop.
end if.
end if.
compute xdata = x.
compute xskip = 1.
compute xname = ddd(1,1:nx).
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).

```

```

compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcx = 4).
compute dummat(1,:) = minus1.
end if.
do if (mcx = 2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcx = 3).
compute dummat=conmat1.
end if.
compute dummat={nnvls, dummat}.
compute xname2={name(1,2), xname}.
end if.
end if.
compute nm=nmeds.
compute r2b = make(nm,1,-999).
compute indeff = make((nx*nm),1,-999).
compute cov =
make((nx*nm+nm),(nx*nm+nm),0).
compute mns=make(nrow(cov),1,0).
compute ny = ncol(dd)-nm-nxd-ncovs.
compute y = dd(:,1).
compute ydata = y.
compute yname = name(1,1).
do if (xskip = 0).
compute x = dd(:,2:(1+nxd)).
compute xdata = x.
compute xname = name(1,2:(1+nxd)).
compute indlbs = t(xname).
end if.
compute m = dd(:,(2+nxd):(1+nxd+nm)).
compute mdata = m.

```

```

compute mlab = { ' M1 ='; ' M2 ='; ' M3 ='; '
M4 ='; ' M5 ='; ' M6 ='; ' M7 ='; ' M8 ='; '
M9 ='; ' M10 = '}.
compute mlab = {mlab; ' M11 ='; ' M12 ='; '
M13 ='; ' M14 ='; ' M15 = '}.
compute mname =
name(1,(2+nxd):(1+nxd+nm)).
do if (ncovs > 0).
compute c =
dd(:,(2+nxd+nm):(1+nxd+nm+ncovs)).
compute x = {x, c}.
compute xdata = x.
compute cdata = c.
compute xname = {xname, cname}.
end if.
compute nms={yname; t(mname);
t(xname(1,1:(ncol(xname)-ncovs)))}.
do if (dumok = 1).
compute nms={yname; t(mname); name(1,2)}.
end if.
do if (nm < 16).
do if (dumok = 1).
print dummat/title = 'Coding of categorical X
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
compute
boots=make((samples+1),((nx+1)*nm),-999).
do if (errs = 0 and criterr = 0).
loop k = 1 to (samples+1).
do if (k = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef=0.
do if (k > 1).
compute u=trunc(uniform(n,1)*n)+1.
compute xdata = x(u,:).
compute mdata = m(u,:).

```

```

compute ydata = y(u,:).
do if (ncovs > 0).
compute cdata = c(u,:).
end if.
end if.
loop i = 1 to nm.
compute start = 1+(i-1)*nx.
compute xmat = {ones, xdata}.
do if (covmy=2).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = mdata(:,i).
compute amat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*amat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-(((1-r2)*(n-1))/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute r2cha = r2.
compute adjr2cha = adjr2.
do if (ncovs > 0 and covmy <> 2).
compute xmat = {ones, cdata}.
compute atmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*atmat)**2).
compute r2step1 = (sstotal-ssresid2)/sstotal.
compute r2cha = r2-r2step1.
compute adjr2cha = adjr2- (1-(((1-r2step1)*(n-
1))/(n-ncol(xmat)))).
compute xmat = {ones, xdata}.
end if.
compute bootcoef={bootcoef,t(amat)}.
compute r2b(i,1)=adjr2cha.
do if (k = 1).

```

```

compute hcinvtX=inv(t( xmat )' xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,1)*hcinvtX*t( xmat
(i3,1)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtX*t( xmat )' xmat
*hcinvXtX
.
compute tratio = amat&/sqrt(diag(covmat)).
compute rnms = mname(1,i).
print rnms/title =
*****
*****'/rlabels = 'Outcome:'format
= A8.
compute lmat = ident(nrow(amat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*amat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*amat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'format = F10.4
/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'.

```

```

compute op = {amat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*(xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute cov(start:(start+(nx-
1)),start:(start+(nx-
1)))=covmat(2:(ncol(covmat)-(ncovs*(covmy <>
2))),2:(ncol(covmat)-(ncovs*(covmy <> 2)))).
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns(start:(start+(nx-
1)),1)=amat(2:(1+nx)).
end loop.
compute xmat = {ones,mdata,xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (k = 1).
print yname/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.

```



```

do if (ovals = 2).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:'/cnames = nmsd/format = F9.2.
end if.
end if.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(yamat)/n)).
compute LL3 = yamat*&ln(pt2)+(1-yamat)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= yamat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).

```

```

compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*ulp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute bootcoef={bootcoef,t(bmat)}.
compute LL1 = LL2.
compute covmat=varb.
end if.
do if (ovals <> 2).
compute bmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*bmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-(((1-r2)*(n-1))/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute bootcoef={bootcoef,t(bmat)}.
end if.
do if (k = 1).
do if (ovals = 2).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.

```

```

compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).
compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms2/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute
opdirsum=outp((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
end if.
do if (ovals <> 2).
.
compute hcinvXtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtXt*( xmat
(i3,:)).

```

```

end loop.
loop i3=1 to k3.
compute xmat(:,i3) = ( resid(:,ncol( resid
))&/(1-h))&* xmat(:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat(:,i3) = sqrt( mse )&* xmat(:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(bmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/(ncol(xmat)-1).
compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat((2+nmeds):(2+nmeds+(nx-1)),:)=
lmat2.
compute fratiodr =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/nx.
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute df2=(n-ncol(xmat)).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), df2, pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = bmat&/sqrt(diag(covmat)).
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute op = {bmat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat)))) }.

```

```

compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp))))*(xp2/(temp-
(2/3)+(.11/temp)))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms2/format =
F10.4.
compute
opdirsum=op((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
compute xmat = {ones,mdata}.
do if (ncovs > 0).
compute xmat = {xmat, cdata}.
end if.
compute btmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*btmat)**2).
compute r2direct = r2-((sstotal-
ssresid2)/sstotal).
compute pfr = 1-fcdf(fratiodr,(nx),(n-1-nx-nm-
ncovs)).
compute r2direct = {r2direct, fratiodr, nx, (n-1-
nx-nm-ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms2/cnames=rnms2/form
at= F10.4.
end if.
end if.

```

```

do if (k = 1 and toteff = 1).
print yname/title = '*****'
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:.'/format = A8.
compute xmat = {ones, xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(ymat)/n)).
compute LL3 = ymat*&ln(pt2)+(1-ymat)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= ymat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.

```

```

compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute LL1 = LL2.
compute covmat=varb.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(bmat)-
1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms = {'Constant'; t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).

```

```

compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute optotsum=outp(2:(1+nx),:).
end if.
do if (ovals <> 2).
compute cmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*cmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).

.

compute hcinvXtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).

```



```

loop i3=1 to k3.
compute xmat(:,i3) = sqrt( mse )&* xmat(:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(cmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = cmatrix/sqrt(diag(covmat)).
compute op = {cmatrix, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute optotsum=op(2:(1+nx),:).
compute ssresid2 = ssresid.
compute r2total = r2.

```

```

compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat(2:(1+nx),:) = lmat2.
compute fratio3 =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/nx.
do if (ncovs > 0).
compute xmat = {ones, cdata}.
compute ctmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*ctmat)**2).
compute r2total = r2-((sstotal-ssresid2)/sstotal).
end if.
compute pfr = 1-fcdf(fratio3,(nx),(n-1-nx-
ncovs)).
compute r2tot2 = {r2total, fratio3, nx, (n-1-nx-
ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns((nrow(mns)-
nm+1):nrow(mns),1)=bmat(2:(1+nm),1).
compute tmp = 1.
loop i = 1 to (nm*nx).
compute
boots(k,i)=mns(i,1)*mns(((nm*nx)+tmp),1).
do if ((i/nx)=trunc(i/nx)).
compute
boots(k,((nm*nx)+tmp))=r2b(tmp,1)*mns(((nm*
nx)+tmp),1).
compute tmp = tmp+1.
end if.
end loop.
do if (savboot = 1 and k > 1).

```

```

compute bootstrp((k-1),:)=bootcoef.
end if.
end loop.
do if (method = 1).
compute llci=make(1,ncol(boots),-999).
compute ulci=make(1,ncol(boots),-999).
compute indeff = t(boots(1,1:(nm*nx))).
compute indeff22=t(boots(1,:)).
compute omni = t(boots(1,((nm*nx)+1) :
ncol(boots))).
do if (nrow(boots) > 1).
compute boots = boots(2:nrow(boots),:).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
loop i = 1 to ncol(boots).
.
compute temp = boots(:,i).
compute temp(GRADE( boots(:,i) )) = boots(:,i).
compute badlo = 0.
compute badhi = 0.
do if ( indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(indeff22(i,1)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.

```

```

do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,i)=llcit.
compute ulci(1,i)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute llci = t(llci).
compute ulci= t(ulci).
compute llci={llci,ulci}.
end if.
end if.
do if (method = 2).
compute tmp=1.
loop i = 1 to (nm*nx).
compute
indeff(i,1)=mns(i,1)&*mns(((nm*nx)+tmp),1).

```

```

do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = sqrt(-
2*ln(uniform(reps,nrow(cov))))&*cos((2*3.1415
9265358979)*uniform(reps,nrow(cov))).
compute boots=boots*chol(cov).
loop i = 1 to ncol(boots).
compute boots(:,i)=boots(:,i)+mns(i,1).
end loop.
compute tmp=1.
loop i = 1 to (nm*nx).
compute
boots(:,i)=boots(:,i)&*boots(:,((nm*nx)+tmp)).
compute temp = boots(:,i).
compute temp(GRADE(boots(:,i))) = boots(:,i).
compute boots(:,i) = temp.
do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = boots(:,1:(nm*nx)).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
compute llci =
t(({boots(cilow,:)};{boots(cihigh,:)})).
end if.
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
compute rnmssum=rnms(2:(1+nx),1).
do if (ovals <> 2).
print optotsum/title = 'Relative total effects of X
of Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.

```

```

print r2tot2/title = 'Omnibus test of total effect of
X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print optotsum/title = 'Relative total effects of X
on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
end if.
compute
rnmssum=rnms2((2+nmeds):(1+nmeds+nx),1).
do if (ovals <> 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.
print r2direct/title = 'Omnibus test of direct
effect of X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
do if (omnibus = 1 and (method = 1 or method
= 3)).
compute indlbs = {indlbs; 'Omnibus'}.
end if.
loop i = 1 to nm.
compute op = mname(1,i).
print op/title = 'Relative indirect effect(s) of X on
Y through:.'/format = A8/space=0.
do if (method = 2).
compute clbs = {'Effect', 'SE MC', 'MC LLCI',
'MC ULCI'}.
compute op = {indeff, bootse(1:(nm*nx)), lici}.
end if.

```

```

do if (method = 1).
compute clbs = {'Effect', 'SE(boot)', 'LLCI',
'ULCI'}.
compute op = indeff.
do if (samples > 0).
compute op = {indeff, bootse(1:(nm*nx),:),
llci(1:(nm*nx),:)}).
end if.
end if.
compute temp = op((((i-1)*nx)+1):(nx*i),:).
do if ((method = 1) and omnibus = 1).
do if (samples = 0).
compute temp={temp;omni(i,1)}.
end if.
do if (samples > 0).
compute temp = {temp; omni(i,1),
bootse(((nm*nx)+i),:), llci(((nm*nx)+i),:)}).
end if.
end if.
print temp/title = ' '/rnames = indlbs/cnames =
clbs/format = F10.4 /space=0.
print/title = '-----'.
end loop.
end if.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
END IF.
DO IF (mdpbe=1).
compute nm={yname,xname,mnames}.
do if (ncovs > 0).

```

```

compute
nm={yname,cnames,xname,mnames}.
end if.

compute tpx=xname.
compute x = dd(:,2:ncol(dd)).
compute tpy = yname.
compute tempy= dd(:,1).
compute n = nrow(dd).
compute nx=1.
compute criterr=0.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute ddd1 = {'int_1', 'int_2', 'int_3', 'int_4',
'int_5', 'int_6', 'int_7', 'int_8', 'int_9'}.
compute dumok = 0.
compute mcfoc=mcx.
compute mcmmod=mcm.
compute mcloc=(mcfoc>0).
compute con = make(n,1,1).
compute ncovs=ncol(x)-2.
do if (mcfoc > 0 or mcmmod > 0).
compute temp = dd.
compute temp(GRADE(dd(:,(ncol(dd)-
mcloc))),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,ncol(dd)-
mcloc)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute toosmall=rsum(nnvls < 2).
compute mnvls = cmin(t(nnvls)).
do if ((rsum(nnvls < 2)) > 0).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=37.
end if.
do if (nvls > 10).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=36.

```



```

end if.
do if (criterr=0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,ncol(dd)-mcloc).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,ncol(dd)-mcloc) <> nnvls((temp-
1),1)).
compute nnvls(temp,1)=dd(i,ncol(dd)-mcloc).
compute temp = temp+1.
end if.
end loop.
compute dummy = dummy(:,2:ncol(dummy)).
do if (mcfoc=4 or mcmmod=4).
compute minus1=make(1,ncol(dummy),-1).
loop k = 1 to n.
do if (rsum(dummy(k,:)) = 0).
compute dummy(k,:)=minus1.
end if.
end loop.
end if.
do if (mcfoc = 2 or mcfoc = 3) or (mcmmod = 2 or
mcmmod = 3)).
loop k = 1 to n.
do if (rsum(dummy(k,:)) > 0).
loop i = 1 to ncol(dummy).
do if (dummy(k,i) = 0).
compute dummy(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcfoc = 3 or mcmmod=3).
compute conmat1={-8,1,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1,1; 0,0,0,0,0,-
3,1,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.

```

```

loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,:)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute
dummy(k,:)=conmat1((rsum(dummy(k,:))+1),:).
end loop.
end if.
end if.
compute nx = ncol(dummy).
compute xname = ddd(1,1:nx).
compute xdata = dummy.
compute xname = ddd(1,1:nx).
compute xname2={nm(1,(ncol(nm)-mcloc)),
xname}.
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcfoc = 2 or mcmmod=2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcfoc = 3 or mcmmod=3).
compute dummat=conmat1.
end if.
do if (mcfoc = 4 or mcmmod = 4).
compute dummat(1,:)=minus1.
end if.
compute dummat={nnvls, dummat}.
end if.
end if.
do if (criterr=0).
compute y = dd(:,1).
compute ovals = ncol(design(dd(:,1))).

```

```

compute itprob = 0.
do if (ovals = 2).
compute omx = cmax(y(:,1)).
compute omn = cmin(y(:,1)).
compute y(:,1) = (y(:,1) = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute sstotal = csum((y-(csum(y)/n))&**2).
compute outv = t(nm(1,1)).
compute mdtr = nm(1,ncol(dd)).
compute fciv = nm(1,(ncol(dd)-1)).
do if (mcfoc <> 0 or mcmod <> 0).
compute inter=make(n,nx,0).
loop i = 1 to nx.
compute inter(:,i)=dummy(:,i)&*dd(:,ncol(dd)-
(1-mcloc)).
end loop.
do if (ncovs = 0).
compute x={con,dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
do if (ncovs > 0).
compute
x={con,dd(:,2:(1+ncovs)),dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={t(nm(1,2:(1+ncovs)));
nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
end if.
compute xmns = csum(x)/n.
compute focvals = {1,0,0}.
compute highwarn = 0.
compute lowwarn = 0.

```

```

do if (ncol(x) > (2*nx+2)).
compute covmns = {1, xmns(1,2:(ncol(x)-1-
(2*nx)))}.
compute focvals = {covmns,0,0}.
end if.
compute dfres = n-ncol(x).
do if (ovals = 2).
compute pt2 = make(n,1,(csum(y)/n)).
compute LL3 = y*&ln(pt2)+(1-y)&ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(x),1,0).
compute LL1 = 0.

.

compute pt1lp= pt1.
compute xlp= x.
compute ylp= y.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).

```

```

compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
else if (ovals <> 2).
compute invXtX=inv(t(x)*x).
compute b = invXtX*t(x)*y.
compute resid = y-(x*b).
compute ssresid=csum(resid*resid).
compute msresid=ssresid/dfres.
compute varb = msresid*invXtX.
compute
seint=sqrt(varb(nrow(varb),nrow(varb))).
compute k3 = nrow(b).
do if (hc3 = 1).
compute xhc=x.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)= xhc(i3,:)*invXtX*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).
end loop.
compute varb=(invXtX*t(xhc)*xhc*invXtX).
end if.
compute seb = sqrt(diag(varb)).
compute r2 = 1-(ssresid/sstotal).

```

```

do if (mcfoc <> 0 or mcmod <> 0).
compute ytp=y.
compute xtp=x(:,1:(ncol(x)-nx)).
compute btp = inv(t(xtp)*xtp)*t(xtp)*ytp.
compute residt = ytp-(xtp*btp).
compute ssresidt=csum(residt*residt).
compute r2noint = 1-(ssresidt/sstotal).
end if.
compute pr = ncol(x)-1.
compute lmat = ident(nrow(b)).
compute lmat = lmat(:,2:ncol(lmat)).
compute f =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
pr).
compute pf = 1-fcdf(f,pr,dfres).
compute pf = {sqrt(r2),r2,msresid,f,pr,dfres,pf}.
compute xd=abs(xp2).
end if.
do if (itprob = 0).
compute tstat = b&/seb.
do if (ovals <> 2).
compute p = 2*(1-tcdf(abs(tstat), dfres)).
else if (ovals = 2).
compute p = 2*(1-cdfnorm(abs(tstat))).
end if.
compute outp = {b,seb,tstat,p}.
compute nms = {'constant'; nms; 'interact'}.
do if (ovals <> 2).
compute tval = (dfres* (exp((dfres-
(5/6))*((xd/(dfres-(2/3)+(.11/dfres)))*(xd/(dfres-
(2/3)+(.11/dfres)))))-1)).
compute outp={outp,(b-
sqrt(tval)&*seb),(b+sqrt(tval)&*seb)}.
end if.
compute bb = tval.
do if (dumok = 1).
do if (mcfoc > 0).
print dummat/title = 'Coding of categorical X
variable for analysis:'/cnames = xname2/format
= F5.2.

```

```

else if (mcmod > 0).
print dummat/title = 'Coding of categorical M
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
print yname/title =
*****
*****'/rlabels = 'Outcome: '/format
= A8.
do if (ovals = 2).
compute nmsd = {outv, 'Analysis'}.
print rcd/title = 'Coding of binary Y for
analysis: '/cnames = nmsd/format = F10.4.
compute outp = {outp,(outp(:,1)-
sqrt(tval)&*outp(:,2)),(outp(:,1)+sqrt(tval)&*outp
(:,2))}.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
print outp/title = 'Logistic Regression
Model'/rnames = nms/clabels 'Coeff' 'se' 'Z' 'p'
'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
compute varbtmp=varb.
compute btmp=b.

```

```

compute LL2f=LL2.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=x(:,1:(ncol(x)-nx)).
compute bt1 = make(ncol(xlp),1,0).

.

compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(y-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)*&ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.

```



```

compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL2f.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(LLdiff,nx).
compute rcha={LLdiff,nx,pchi}.
end if.
do if (ovals <> 2).
print pf/title = 'Model Summary'/clabels = 'R' 'R-
sq' 'MSE' 'F' 'df1' 'df2' 'p'/format F10.4.
compute lmat=make(nrow(b),1,0).
compute lmat(nrow(lmat),1)=1.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
1).
compute
rcha=((b(nrow(b),1)/seint)*(b(nrow(b),1)/seint))*
(1-r2)/dfres.
compute rcha = {rcha, fcha, 1, dfres,
outp((pr+1),4)}.
print outp/title = 'Model'/rnames = nms/clabels
'coeff' 'se' 't' 'p' 'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
do if (mcfoc > 0 or mcmod > 0).
compute rcha=r2-r2noint.
compute lmat=make((nrow(b)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
nx).
compute pvalr2c=1-fcdf(fcha,nx,dfres).

```

```

compute rcha = {rcha, fcha, nx, dfres,pvalr2c}.
end if.
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute intkey = {'a', 'b', 'c', 'd', 'e'}.
loop i = 1 to nx.
compute intkey={intkey; ddd1(1,i), ' : ', ddd(1,i),
' X ', nm(1,ncol(dd)-(1-mcloc))}.
end loop.
compute intkey=intkey(2:nrow(intkey),:).
print intkey/title='Product terms key:.'/format =
a8.
end if.
do if (ovals <> 2).
print rcha/title = 'R-square increase due to
interaction:.'/clabels 'R2-chng' 'F' 'df1' 'df2'
'p'/format F10.4.
else if (ovals = 2).
print rcha/title = 'Likelihood ratio test for
interaction:.'/clabels 'Chi-sq' 'df' 'p'/format F10.4.
end if.
print/title =
*****
*****'.
compute mdvar = x(:,(ncol(x)-1)).
do if (mcfoc = 0 and mcmmod = 0).
compute g1 = b((ncol(x)-2),1).
compute g3 = b(ncol(x),1).
compute vg1 = varb((ncol(x)-2),(ncol(x)-2)).
compute vg3 = varb(ncol(x),ncol(x)).
compute covg1g3 = varb((ncol(x)-2), ncol(x)).
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute mdvar = x(:,(ncol(x)-(2*nx))).
end if.
compute mdmin = cmin(mdvar).
compute mdmax = cmax(mdvar).
compute fvar = x(:,(ncol(x)-2)).
compute nval = ncol(design(mdvar)).
compute fvmin = cmin(fvar).

```

```

compute fvmax = cmax(fvar).
do if (mmodval = 999).
compute mnmd = csum(mdvar)/n.
compute tmp = make(n,1,mnmd).
compute sdmd = sqrt(csum(((mdvar-
tmp)**2))/(n-1)).
compute probeval = {mnmd-sdmd; mnmd;
mnmd+sdmd}.
do if (probeval(1,1) < mdmin).
compute lowwarn = 1.
compute probeval(1,1)=mdmin.
end if.
do if (probeval(nrow(probeval),1) > mdmax).
compute highwarn = 1.
compute probeval(nrow(probeval),1)=mdmax.
end if.
do if (quantile = 1).
compute tmp = mdvar.
compute tmp(GRADE(mdvar(:,1)),:) = mdvar.
compute mdvar = tmp.
compute
probeval={mdvar(trunc(0.1*n),1);mdvar(trunc(0
.25*n),1);mdvar(trunc(0.5*n),1);mdvar(trunc(0.7
5*n),1);mdvar(trunc(0.9*n),1)}.
end if.
end if.
do if (nval = 2).
compute probeval = make(2,1,0).
compute probeval(1,1)=cmin(mdvar).
loop i = 1 to n.
do if (mdvar(i,1) <> probeval(1,1)).
compute probeval(2,1) = mdvar(i,1).
BREAK.
end if.
end loop.
end if.
do if (mmodval <> 999).
compute probeval = mmodval.
end if.
compute outp = make(nrow(probeval),7,0).

```

```

do if (mcfoc > 0 or mcmmod > 0).
compute focvals=make(1,ncol(x)+1,1).
do if (mcfoc > 0).
print/title = 'Conditional effect of X on Y at
values of the moderator:'.
print/title = ' '/space=0.
compute rnn2=mdtr.
compute matt=make(nx,6,0).
end if.
loop jj=1 to nrow(probeval).
do if (mcfoc > 0).
loop ii=1 to nx.
compute g1=b((ncol(x)-(2*nx)+ii),1).
compute g3=b((ncol(x)-nx+ii),1).
compute vg1=varb((ncol(x)-(2*nx)+ii),(ncol(x)-
(2*nx)+ii)).
compute vg3=varb((ncol(x)-nx+ii),(ncol(x)-
nx+ii)).
compute covg1g3=varb((ncol(x)-
(2*nx)+ii),(ncol(x)-nx+ii)).
compute x2 = probeval(jj,1).
compute w1 = g1+g3*x2.
compute varw1 =
vg1+(2*x2*covg1g3)+((x2*x2)*vg3).
compute sew1 = sqrt(varw1).
compute t1 = w1/sew1.
do if (ovals <> 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-tcdf(abs(t1), dfres)).
compute cnms = {'Coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)= {w1,sew1,t1,p,llci,ulci}.
else if (ovals = 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-cdfnorm(abs(t1))).
compute cnms = {'Coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)= {w1,sew1,t1,p,llci,ulci}.

```

```

end if.
end loop.
compute rnms=t(xname).
compute mdvalpr=probeval(jj,1).
print mdvalpr/title = 'Moderator
value: '/rnames=rnn2/format= F10.4 /space=0.
print matt/title='
'/cnames=cnms/rnames=rnms/format= F10.4
/space=0.
compute xprob=x(:,(ncol(x)-(2*nx)))-mdvalpr.
loop kk = 1 to nx.
compute xprob={xprob,
(xprob(:,1)&*x(:,(ncol(x)-(2*nx)+kk)))}.
end loop.
compute xprob={x(:,1:((ncol(x)-(2*nx))-
1)),xprob}.
do if (ovals <> 2).
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute residc = y-(xprob*bmultc).
compute ssresidc=csum(residc&*residc).
compute r2c = r2-(1-(ssresidc/sstotal)).
compute fcha2=(dfres*r2c)/(nx*(1-r2)).
do if (hc3 = 1).
loop kk = 1 to nx.
compute xprob={xprob, x(:,(ncol(x)-
(2*nx)+kk))}.
end loop.
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute k3 = nrow(bmultc).
compute xhc=xprob.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)=
xhc(i3,)*inv(t(xprob)*xprob)*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).

```

```

end loop.
compute
varbc=(inv(t(xprob)*xprob)*t(xhc)*xhc*inv(t(xprob)*xprob)).
compute lmat=make((nrow(bmultc)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha2 =
(t(t(lmat)*bmultc)*inv(t(lmat)*varbc*lmat)*((t(lmat)*bmultc)))/nx).
end if.
compute pvalr2cc=1-fcdf(fcha2,nx,dfres).
compute rcha2 = {r2c, fcha2, nx,
dfres,pvalr2cc}.
print rcha2/title = 'Test of equality of conditional
means at this value of the moderator'/clabels
'R2-chng' 'F' 'df1' 'df2' 'p'/format F10.4.
else if (ovals = 2).
compute btmp=b.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=xprob.
compute bt1 = make(ncol(xlp),1,0).

.
compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.

```

```

end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL4.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(lldiff,nx).
compute rcha2 = {LLdiff, nx,pchi}.
print rcha2/title = 'Test of equality of log odds
conditioned on this moderator value'/clabels
'Chi-sq' 'df' 'p'/format F10.4.
end if.
end if.
compute
tttt=make(nrow(dummat),1,probeval(jj,1)).
compute tttt={tttt,dummat(:,2:ncol(dummat))}.
loop kkk=1 to nx.
compute tttt={tttt,tttt(:,1)*tttt(:,1+kkk)}.
end loop.
compute ones=make(nrow(dummat),1,1).
do if (ncol(x) > (2*nx+2)).

```

```

compute
covmnmat=make(nrow(tttt),ncol(covmns),0).
loop kkk=1 to nrow(tttt).
compute covmnmat(kkk,:)=covmns.
end loop.
compute tttt={covmnmat,tttt}.
else.
compute tttt={ones,tttt}.
end if.
compute focvals={focvals;dummat(:,1),tttt}.
do if (mcfoc > 0).
compute yhat={dummat(:,1),(tttt*b)}.
compute cmnms={fciv, 'yhat'}.
do if (ovals <> 2).
print yhat/title = 'Estimated conditional means
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (ovals = 2).
print yhat/title = 'Estimated conditional log odds
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (jj <> nrow(probeval)).
print/title='-----'/space=0.
end if.
end if.
end loop.
compute focvals=focvals(2:nrow(focvals),:).
compute yhat=focvals(:,2:ncol(focvals))*b.
compute
focvals={focvals(:,1),focvals(:,(ncol(focvals)-
(2*nx))),yhat}.
compute cnms={fciv,mdtr,'yhat'}.
do if (mcmod > 0).
compute cnms={mdtr,fciv,'yhat'}.
end if.
end if.
do if (mcmod > 0).
compute outp=make((nx+1),7,0).

```



```

compute bcatm={b((ncol(x)-
(2*nx)),1);b((ncol(x)-nx+1):ncol(x),1)}.
compute outp(:,1)=dummat(:,1).
compute bcatcov=varb((ncol(x)-
nx):ncol(x),(ncol(x)-nx):ncol(x)).
compute bcatcov(1,1)=varb((ncol(x)-
(2*nx)),(ncol(x)-(2*nx))).
compute
bcatcov(2:nrow(bcatcov),1)=varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx)).
compute
bcatcov(1,2:nrow(bcatcov))=t(varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx))).
loop i = 1 to nrow(dummat).
compute
catmval={1,dummat(i,2:ncol(dummat))}.
compute condeff=catmval*bcatm.
compute
condse=sqrt(catmval*bcatcov*t(catmval)).
compute outp(i,2:3)={condeff,condse}.
end loop.
compute outp(:,4)=outp(:,2)&/outp(:,3).
compute outp(:,5) = 2*(1-tcdf(abs(outp(:,4)),
dfres)).
compute outp(:,6) = (outp(:,2)-
sqrt(tval)&*outp(:,3)).
compute outp(:,7) =
(outp(:,2)+sqrt(tval)&*outp(:,3)).
do if (ovals <> 2).
compute cnmms = {xname2(1,1), 'coeff', 'se',
't', 'p', 'LLCI', 'ULCI'}.
end if.
do if (ovals = 2).
compute outp(:,5) = 2*(1-
cdfnorm(abs(outp(:,4)))).
compute cnmms = {xname2(1,1), 'coeff', 'se',
'Z', 'p', 'LLCI', 'ULCI'}.
end if.

```

```

print outp/title = 'Conditional Effect of Focal
Predictor in Groups Defined by the Moderator
Variable:.'/cnames=cnmms/format = F10.4.
end if.
do if (nval > 2 and (mmodval = 999) and
mcmmod = 0).
do if (quantile <> 1).
print/title = 'Moderator values are the sample
mean and plus/minus one SD from mean'.
else.
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
end if.
do if (highwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD above the mean was'.
print/title = 'replaced with the maximum
because one SD above the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
do if (lowwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD below the mean was'.
print/title = 'replaced with the minimum
because one SD below the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
end if.
do if (nval = 2 and (mmodval = 999)).
print/title = 'The moderator variable is
dichotomous'.
end if.
do if (mmodval <> 999 and mcmmod = 0 and
((mmodval < mdmin) or (mmodval > mdmax))).
print/title = 'Warning: Moderator value specified
is outside of the range of the data'.
end if.

```

```

do if (plot = 1).
do if (mcfoc = 0 and mcmmod=0).
compute cnms = {t(nms((ncol(x)-2):(ncol(x)-
1),1)), 'yhat'}.
end if.
do if (ovals = 2).
compute prob =
exp(focvals(:,3))/(1+exp(focvals(:,3))).
compute focvals = {focvals, prob}.
compute cnms = {cnms, 'prob'}.
end if.
print/title =
*****
*****!

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (ovals = 2).
do if (mcfoc=0).
print/title = 'DATA LIST FREE/Eco_Innovation
Plan_Orga_Pract logodds prob.' /space=1.
end if.
do if (mcmmod=0).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Eco_Innovation
logodds prob.' /space=1.

```

```

end if.
else if (ovals <> 2).
do if (mcfoc=0).
print/title = 'DATA LIST FREE/Eco_Innovation
Plan_Orga_Pract Financial_Performance.'
/space=1.
end if.
do if (mcmod=0).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Eco_Innovation
Financial_Performance.' /space=1.
end if.
end if.
print/title = 'BEGIN DATA.' /space=0.
print focvals/title = ' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (ovals <> 2 and mcmod=0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Financial_Performance BY
Plan_Orga_Pract.' /space=0.
end if.
do if (ovals <> 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH Financial_Performance BY
Eco_Innovation.' /space=0.
end if.
do if (ovals = 2 and mcmod=0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Plan_Orga_Pract.'
/space=0.
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Plan_Orga_Pract.' /space=0.
end if.
do if (ovals = 2 and mcfoc=0).

```

```

print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH logodds BY Eco_Innovation.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH prob BY Eco_Innovation.' /space=0.
end if.
do if (ncovs > 0).
print/title = 'NOTE: For data above, covariates
are set to their sample means.'.
end if.
end if.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
end if.
END IF.
end if.
do if (bad > 0).
compute note(notes,1) = 9.
compute notes = notes + 1.
end if.
print/title = '***** ANALYSIS
NOTES AND WARNINGS
*****'.
loop i = 1 to errs.
do if (runerrs(i,1) = 1).
print/title = 'ERROR: One of your declared
mediators is dichotomous. This procedure
can"t be used.'.
end if.
do if (runerrs(i,1) = 2).

```

```
print/title = 'ERROR: For model 6, this
procedure limits the number of mediators to
four.'.
end if.
do if (runerrs(i,1) = 3).
print/title = 'ERROR: For models 1, 2, and 3,
only a single variable can be listed in the M
list.'.
end if.
do if (runerrs(i,1) = 4).
print/title = 'ERROR: You requested a model
involving W but did not provide a valid W
variable name.'.
end if.
do if (runerrs(i,1) = 5).
print/title = 'ERROR: You requested a model
involving Z but did not provide a valid Z
variable name.'.
end if.
do if (runerrs(i,1) = 6).
print/title = 'ERROR: You requested a model
involving Q but did not provide a valid Q
variable name.'.
end if.
do if (runerrs(i,1) = 7).
print/title = 'ERROR: You requested a model
involving V but did not provide a valid V
variable name.'.
end if.
do if (runerrs(i,1) = 8).
print/title = 'ERROR: You specified a W
variable for a model that does not need it'.
end if.
do if (runerrs(i,1) = 9).
print/title = 'ERROR: You specified a Z variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 10).
print/title = 'ERROR: You specified a Q variable
for a model that does not need it'.
```

```
end if.  
do if (runerrs(i,1) = 11).  
print/title = 'ERROR: You specified a V variable  
for a model that does not need it.'  
end if.  
do if (runerrs(i,1) = 12).  
print/title = 'ERROR: The variable specified for  
W has already been assigned.'  
end if.  
do if (runerrs(i,1) = 13).  
print/title = 'ERROR: The variable specified for  
Z has already been assigned.'  
end if.  
do if (runerrs(i,1) = 14).  
print/title = 'ERROR: The variable specified for  
Q has already been assigned.'  
end if.  
do if (runerrs(i,1) = 15).  
print/title = 'ERROR: The variable specified for  
V has already been assigned.'  
end if.  
do if (runerrs(i,1) = 16).  
print/title = 'ERROR: You did not provide a  
valid Y variable name.'  
end if.  
do if (runerrs(i,1) = 17).  
print/title = 'ERROR: The variable specified for  
Y has already been assigned.'  
end if.  
do if (runerrs(i,1) = 18).  
print/title = 'ERROR: Model 6 requires more  
than one mediator.'  
end if.  
do if (runerrs(i,1) = 19).  
print/title = 'ERROR: You have not specified a  
valid model number.'  
end if.  
do if (runerrs(i,1) = 20).  
print/title = 'ERROR: At least one and only one  
variable must be listed for X.'
```

```
end if.  
do if (runerrs(i,1) = 21).  
print/title = 'ERROR: At least one and only one  
variable must be listed for Y.'.  
end if.  
do if (runerrs(i,1) = 22).  
print/title = 'ERROR: Iteration didn"t converge  
to a solution. Interpret results with caution.'.  
end if.  
do if (runerrs(i,1) = 23).  
print/title = 'ERROR: You specified a clustering  
variable that does not exist in your variable  
list.'.  
end if.  
do if (runerrs(i,1) = 24).  
print/title = 'ERROR: You specified a clustering  
variable that has already been assigned.'.  
end if.  
do if (runerrs(i,1) = 25).  
print/title = 'ERROR: One or more of your M  
variables is not listed in the variables list.'.  
end if.  
do if (runerrs(i,1) = 26).  
print/title = 'ERROR: A maximum of 20 cluster  
units is allowed. Use multilevel modeling  
instead.'.  
end if.  
do if (runerrs(i,1) = 27).  
print/title = 'ERROR: One of the variables in  
your model is a constant.'.  
end if.  
do if (runerrs(i,1) = 28).  
print/title = 'ERROR: Dichotomous Y is not  
permitted with WS option.'.  
end if.  
do if (runerrs(i,1) = 29).  
print/title = 'ERROR: Insufficient number of  
variables in vars= list when using WS option.'.  
end if.  
do if (runerrs(i,1) = 30).
```



```

print/title = 'ERROR: Too many variables in
vars= list when using WS option. Covariates
not allowed.'.
end if.
do if (runerrs(i,1) = 31).
print/title = 'ERROR: mmodval and wmodval
can"t both be set to zero with contrast option.'.
end if.
do if (runerrs(i,1) = 32).
print/title = 'ERROR: You did not provide a
valid X variable name.'.
end if.
do if (runerrs(i,1) = 33).
print/title = 'ERROR: PROCESS requires all
variable names to be eight characters or
fewer.'.
print/title = '      Please shorten variable names
and reexecute.'/space=0.
end if.
do if (runerrs(i,1)=34).
print/title = 'ERROR: X and M cannot both be
specified as multicategorical.'.
end if.
do if (runerrs(i,1)=35).
print/title = 'ERROR: Multicategorical variable
specification available only for models 1 and
4.'.
end if.
do if (runerrs(i,1)=36).
print/title = 'ERROR: Categorical variables
cannot have more than 10 categories.'.
end if.
do if (runerrs(i,1)=37).
print/title = 'ERROR: Each group must have at
least two cases.'.
end if.
do if (runerrs(i,1)=38).
print/title = 'ERROR: Cluster option not
available with multicategorical variables.'.
end if.

```

```

do if (runerrs(i,1)=39).
print/title = 'ERROR: A maximum of 10
variables is allowed in the M= list.'.
end if.
do if (runerrs(i,1) = 41).
print/title = 'ERROR: MCM option not permitted
in model 4.'.
end if.
end loop.
do if (errs = 0).
do if (boot > 1 or mc > 0).
do if (bconoff = 1 and boot > 1).
print boot/title = 'Number of bootstrap samples
for bias corrected bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (bconoff = 0 and boot > 1).
print boot/title = 'Number of bootstrap samples
for percentile bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (mc > 1).
print mc/title = 'Number of samples for Monte
Carlo confidence intervals:'/format = F8.0.
end if.
do if (booterr = 1).
compute badend = badend(1,2:ncol(badend)).
print badend/title = 'WARNING: Bootstrap CI
endpoints below not trustworthy. Decrease
confidence or increase bootstraps'/format =
F10.4.
end if.
end if.
print conf/title = 'Level of confidence for all
confidence intervals in output:'/format = F8.2.
do if ((center = 1 or ws=1) and (ncol(centvar) >
1)).
compute centvar = centvar(1,2:ncol(centvar)).

```

```

print centvar/title = 'NOTE: The following
variables were mean centered prior to
analysis:.'/format = a8.
end if.
loop i = 1 to notes.
do if (note(i,1) = 1).
print/title = 'NOTE: Confidence level restricted
to between 50 and 99.9999%. 95% confidence
is provided in output'.
end if.
do if (note(i,1) = 2).
print/title = 'NOTE: Effect size measures for
indirect effects are not available for models
with dichotomous outcomes'.
end if.
do if (note(i,1) = 3).
print/title = 'NOTE: All standard errors for
continuous outcome models are based on the
HC3 estimator'.
end if.
do if (note(i,1) = 6).
print/title = 'NOTE: The number of bootstrap
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 7).
print/title = 'NOTE: The Johnson-Neyman
method is available only for Models 1 and 3'.
end if.
do if (note(i,1) = 8).
print/title = 'NOTE: The Johnson-Neyman
method cannot be used with a dichotomous
moderator'.
end if.
do if (note(i,1)=9).
print bad/title = 'NOTE: Some bootstrap
samples had to be replaced. The number of
such replacements was:'.
end if.
do if (note(i,1) = 11).

```

```
print nmiss/title = 'NOTE: Some cases were
deleted due to missing data. The number of
such cases was:'.
end if.
do if (note(i,1) = 12).
print /title = 'NOTE: Monte Carlo method
available only for models 4 and 5.
Bootstrapping was used instead.'.
end if.
do if (note(i,1) = 13).
print/title = 'NOTE: The number of Monte Carlo
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 19).
print/title = 'NOTE: Effect sizes not available for
within-subject analyses.'.
end if.
do if (note(i,1) = 16).
print/title = 'NOTE: Normal theory tests not
available for within-subject analyses.'.
end if.
do if (note(i,1) = 17).
print/title = 'NOTE: Monte Carlo confidence
intervals not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 18 and warnrep=0).
print/title = 'WARNING: You have requested
OLS estimation with a dichotomous criterion.'.
print/title = 'Interpret model coefficients and
inferential statistics with caution.'/space=0.
compute warnrep=1.
end if.
do if (note(i,1) = 20).
print/title = 'NOTE: Saving of bootstrap
estimates not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 22).
```

```
print/title = 'NOTE: Effect size option with
covariates requires covariates in models of M
and Y.'.
end if.
do if (note(i,1) = 23).
print/title = 'NOTE: Johnson-Neyman method
not available with multicategorical X'.
end if.
do if (note(i,1) = 24).
print/title = 'NOTE: MMODVAL option not
available with a multicategorical moderator'.
end if.
do if (note(i,1) = 25).
print/title = 'NOTE: Omnibus test for relative
indirect effects is not available with the MC
option.'.
end if.
do if (note(i,1) = 26).
print/title = 'NOTE: CONTRAST option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 27).
print/title = 'NOTE: EFFSIZE option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 28).
print/title = 'NOTE: NORMAL option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 31).
print/title = 'NOTE: The TOTAL option is not
available when Y is dichotomous.'.
end if.
do if (note(i,1) = 32).
print/title = 'NOTE: Kappa-squared is disabled
from output as of version 2.16.'.
end if.
do if (note(i,1) = 33).
print/title = 'NOTE: COVMY option ignored in
models 1, 2, and 3.'.
```

| | | | |
|-----------|----------------|-------------|-------------|
| | | end if. | |
| | | end loop. | |
| | | end if. | |
| | | end matrix. | |
| Resources | Processor Time | | 00:00:04.55 |
| | Elapsed Time | | 00:00:04.52 |

Run MATRIX procedure:

***** PROCESS Procedure for SPSS Release 2.16.3 *****

Written by Andrew F. Hayes, Ph.D. www.afhayes.com
Documentation available in Hayes (2013). www.guilford.com/p/hayes3

Model = 4
Y = Financia
X = Plan_Org
M = Eco_Inno

Sample size
856

Outcome: Eco_Inno

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|-----------|--------|----------|-------|
| .8299 | .6887 | .3590 | 1889.6042 | 1.0000 | 854.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|-------|-------|---------|-------|-------|-------|
| constant | .3531 | .0810 | 4.3618 | .0000 | .1942 | .5121 |
| Plan_Org | .8827 | .0203 | 43.4696 | .0000 | .8429 | .9226 |

Outcome: Financia

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|-----------|--------|----------|-------|
| .9212 | .8486 | .1836 | 2390.7067 | 2.0000 | 853.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|--------|-------|---------|-------|--------|--------|
| constant | -.1195 | .0585 | -2.0416 | .0415 | -.2344 | -.0046 |
| Eco_Inno | .4803 | .0245 | 19.6288 | .0000 | .4323 | .5283 |
| Plan_Org | .5388 | .0260 | 20.7018 | .0000 | .4877 | .5899 |

***** DIRECT AND INDIRECT EFFECTS *****

Direct effect of X on Y

| Effect | SE | t | p | LLCI | ULCI |
|--------|----|---|---|------|------|
|--------|----|---|---|------|------|

```

        .5388        .0260        20.7018        .0000        .4877        .5899

Indirect effect of X on Y
      Effect      Boot SE      BootLLCI      BootULCI
Eco_Inno      .4240      .0277      .3718      .4813

Normal theory tests for indirect effect
      Effect      se      Z      p
      .4240      .0237      17.8855      .0000

***** ANALYSIS NOTES AND WARNINGS *****

Number of bootstrap samples for bias corrected bootstrap confidence
intervals:
      5000

Level of confidence for all confidence intervals in output:
      95.00

----- END MATRIX -----

restore.

```

Matrix

| Notes | | |
|----------------|--------------------------------|----------------------|
| Output Created | | 20-OCT-2020 16:00:29 |
| Comments | | |
| Input | Active Dataset | DataSet1 |
| | Filter | <none> |
| | Weight | <none> |
| | Split File | <none> |
| | N of Rows in Working Data File | 856 |

Syntax

```
matrix.  
get dat/file = */variables =  
Financial_Performance Opra_Pract  
Eco_Innovation /names = vnames/missing =  
9999.  
compute ninit = nrow(dat).  
get dat/file = */variables =  
Financial_Performance Opra_Pract  
Eco_Innovation /names = vnames/missing =  
omit.  
get tmp/file = */variables =  
Financial_Performance /names =  
yname/missing = omit.  
get tmp2/file = */variables = Opra_Pract  
/names = xname/missing = omit.  
get tmp/file = */variables = Eco_Innovation  
/names = mnames/missing = omit.  
get tmp/file = */variables = w999999t z999999t  
v999999t q999999t.  
compute wname=tmp(1,1).  
do if (wname = ' ').  
compute wname = 'xxx'.  
end if.  
compute zname=tmp(1,2).  
do if (zname = ' ').  
compute zname = 'xxx'.  
end if.  
compute vname=tmp(1,3).  
do if (vname = ' ').  
compute vname = 'xxx'.  
end if.  
compute qname=tmp(1,4).  
do if (qname = ' ').  
compute qname = 'xxx'.  
end if.  
compute n = nrow(dat).  
compute p0=-.322232431088.  
compute p1 = -1.  
compute p2 = -.342242088547.  
compute p3 = -.0204231210245.
```



```

compute p4 = -.0000453642210148.
compute q0 = .0993484626060.
compute q1 = .588581570495.
compute q2 = .531103462366.
compute q3 = .103537752850.
compute q4 = .0038560700634.
compute badend = 0.
compute priorlo = -9999999.
compute priorhi = 9999999.
compute criterr = 0.
compute cluster = 0.
compute clsdmy = 0.
compute jndich = 0.
compute wvdich=0.
compute zqdich=0.
compute mod74dic=0.
compute warnrep=0.
compute contrast = trunc( 0 ).
do if (contrast > 2 or contrast < 0).
compute contrast = 0.
end if.
compute booterr = 0.
compute effsize = ( 0 = 1).
compute note = make(10,1,0).
compute notes = 1.
compute iterr = 0.
compute clsmtn = 0.
compute quantile = ( 0 = 1).
compute mmodval= 999.
compute jn = ( 0 = 1).
compute covcoeff=( 0 =1).
compute center = ( 0 = 1).
compute detail = ( 1 = 1).
compute coeffci = ( 1 = 1).
compute conf = 95.
compute longname = ( 1 =1).
compute bconoff=( 0 <> 1).
compute covmy = trunc( 0 ).
do if (covmy < 0 or covmy > 2).
compute covmy = 0.

```

```

end if.
do if (trunc(conf) >= 100 or (trunc(conf) <=
50)).
compute conf = 95.
compute note(notes,1) = 1.
compute notes = notes + 1.
end if.
do if (n < ninit).
compute nmiss = ninit-n.
compute note(notes,1) = 11.
compute notes = notes + 1.
end if.
compute errs = 0.
compute quantd = {0,0,0,0,0,0}.
compute quantc = {0,0,0,0,0,0}.
compute mcheck = 0.
compute ttt = 0.
compute plot = trunc( 0 ).
compute runerrs = make(50,1,0).
compute model = trunc( 4 ).
compute normal = 1.
compute ws= ( 0 =1).
compute olsdichm= ( 0 =1).
compute olsdichy= ( 0 =1).
compute mcx=trunc( 0 ).
compute mcm=trunc( 0 ).
do if (ws = 1).
do if (effsize = 1).
compute note(notes,1) = 19.
compute notes = notes + 1.
compute effsize=0.
end if.
do if (normal=1).
compute note(notes,1) = 16.
compute notes = notes + 1.
compute normal=0.
end if.
end if.
do if (model <> 4).
compute ws = 0.

```

```

end if.
do if (model > 0 and model < 4 and covmy <>
0).
compute covmy=0.
compute note(notes,1) = 33.
compute notes = notes + 1.
end if.
print/space=0.
do if (jn = 1 and model <> 1 and model <> 3).
compute note(notes,1) = 7.
compute notes = notes + 1.
end if.
do if ((model > 76) or (model < 1)).
compute model = 77.
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 19.
end if.
compute itprob=0.
do if (mcx < 0).
compute mcx=0.
end if.
do if (mcx > 4).
compute mcx=0.
end if.
do if (mcm < 0).
compute mcm=0.
end if.
do if (mcm > 4).
compute mcm=0.
end if.
do if (mcx > 0 and mcm > 0).
compute errs=errs+1.
compute runerrs(errs,1)=34.
compute criterr=1.
end if.
do if ((model <> 1 and model <> 4) and (mcx >
0 or mcm > 0)).
compute errs=errs+1.
compute runerrs(errs,1)=35.

```

```
compute criterr=1.  
end if.  
do if ((model = 4) and (mcm > 0)).  
compute errs=errs+1.  
compute runerrs(errs,1)=41.  
compute criterr=1.  
end if.  
do if ((mcx > 0) and jn <> 0).  
compute jn=0.  
compute note(notes,1) = 23.  
compute notes = notes + 1.  
end if.  
do if (mmodval <> 999 and mcm > 0).  
compute note(notes,1) = 24.  
compute notes = notes + 1.  
end if.  
compute toteff = 0.  
compute toteff = (( 0 = 1)*( 4 = 4 or 4 = 6)).  
compute varorder = 2.  
compute hc3 = ( 0 <> 0).  
compute cname = 'xxx'.  
compute centvar = {'xxx'}.  
compute nmeds = ncol(mnames).  
do if (longname=0).
```

```
compute criterr = 1.
```

```
compute errs = errs+1.
```

```
compute runerrs(errs,1) = 33.
```

1

2

3

end if.

compute modelm =

```
{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,2;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,3;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,5;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,6;  
1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,7;  
1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,8;  
1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,9;  
1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,3,10;  
1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,5,11;  
1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,0,5,12;  
1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5,13;  
0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,2,14;  
0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,2,15;  
0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,4,16;  
0,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,4,17;  
0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,6,18;  
0,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,6,19;  
0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,6,20;  
1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,7,21;  
1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,7,22;  
1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,23;  
1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,24; 1,1,1,1
```

```
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,25;  
1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,26;  
1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,27;  
1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,7,28;  
1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,7,29;  
1,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,30;
```

1,1,0,1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,31;
1,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,32;
1,1,1,1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,33;
1,1,1,1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,34;
1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,35;
1,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,36;
1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,37;
1,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,38;
1,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,39;
1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,40;
1,0,0,1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,0,41;
1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,42;
1,0,0,1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,0,43;
1,0,0,1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,0,44;
1,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,45;
1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,46;
1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,47;
1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,48;
1,1,0,1,1,0,1,1,0,0,0,

0,0,0,0,0,0,0,0,0,49;
1,1,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,50;
1,1,1,1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,51;
1,1,1,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,52;
1,1,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,53;
1,1,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,54;
1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,55;
1,1,1,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,56;
1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,57;
1,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,58;
1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,59;
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,60;
1,1,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,61;
1,1,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,62;
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,0,0,0,0,63;
1,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,64;
1,0,0,1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,65;
1,0,0,1,0,0,0,0,0,0,1,0,0,1,0,0,0,0,0,66;
1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,67;
1,1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,68;


```

1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,69;
1,0,0,1,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,70;
1,0,0,1,0,0,1,0,0,1,0,0,1,1,1,0,0,0,0,71;
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,1,1,0,0,72;
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,1,1,0,0

,73; 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,74;
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,75;
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,1,0,0,0,76;
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,77}.
compute wm = modelm(model, 1).
compute zm = modelm(model, 2).
compute wzm = modelm(model, 3).
compute vy = modelm(model, 4).
compute qy = modelm(model, 5).
compute vqy = modelm(model, 6).
compute wy = modelm(model, 7).
compute zy = modelm(model, 8).
compute wzy = modelm(model, 9).
compute vxy = modelm(model, 10).
compute qxy = modelm(model, 11).
compute vqxy = modelm(model, 12).
compute wmy = modelm(model, 13).
compute wvmy = modelm(model, 14).
compute wvxy = modelm(model, 15).
compute zmy = modelm(model, 16).
compute wzmy = modelm(model, 17).
compute xmy = modelm(model, 18).
compute imm=modelm(model, 19).
do if (model = 74).
compute vdich=0.
end if.
do if (ncol(xname) <> 1).
compute errs = errs+1.
compute runerrs(errs, 1) = 20.
compute criterr = 1.
end if.
do if (ncol(mnames) > 10).
compute errs = errs+1.
compute runerrs(errs, 1) = 39.

```

```

compute criterr = 1.
end if.
do if (ncol(yname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 21.
compute criterr = 1.
end if.
do if (( 999 =0) and ( 999 =0) and (model=3 or
model=2) and (contrast=1)).
compute errs = errs+1.
compute runerrs(errs,1) = 31.
compute criterr = 1.
end if.
compute xlist = (wm or zm or wzm or wy or zy
or wzy or vxy or qxy or vqxy or wvxy or xmy).
compute mlist = (vy or qy or vqy or zmy or wmy
or wzmy or xmy or (model < 4)).
compute bad=0.
do if (criterr = 0).
compute werr = 0.
compute verr = 0.
compute qerr = 0.
compute zerr = 0.
compute yerr = 1.
compute xerr = 1.
compute wlist = (wm or wzm or wy or wzy or
wm or wvmy or wvmy or wvxy or wzmy).
do if (wlist = 1 and wname = 'xxx').
compute werr = 1.
compute wlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (wlist = 1 and ((wname = qname) or
(wname = vname) or (wname = zname) or
(wname = xname) or (wname = yname))).
compute werr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 12).
end if.

```

```

compute zlist = (zm or wzm or zy or wzy or
zmy or wzmy).
do if (zlist = 1 and zname = 'xxx').
compute zerr = 1.
compute zlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (zlist = 1 and ((zname = qname) or
(zname = vname) or (zname = wname) or
(zname = xname) or (zname = yname))).
compute zerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 13.
end if.
compute qlist = (qy or vqy or qxy or vqxy).
do if (qlist = 1 and qname = 'xxx').
compute qerr = 1.
compute qlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (qlist = 1 and ((qname = zname) or
(qname = vname) or (qname = wname) or
(qname = xname) or (qname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 14.
end if.
compute vlist = (vy or vqy or vxy or vqxy or
wvmy or wvxy).
do if (vlist = 1 and vname = 'xxx').
compute verr = 1.
compute vlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (vlist = 1 and ((vname = zname) or
(vname = qname) or (vname = wname) or
(vname = xname) or (vname = yname))).

```

```

compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 15.
end if.
do if (wlist = 0 and wname <> 'xxx').
compute werr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 8.
end if.
do if (zlist = 0 and zname <> 'xxx').
compute zerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 9.
end if.
do if (qlist = 0 and qname <> 'xxx').
compute qerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 10.
end if.
do if (vlist = 0 and vname <> 'xxx').
compute verr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 11.
end if.
do if (hc3 = 1).
compute note(notes,1) = 3.
compute notes = notes+1.
end if.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute cons = make(n,1,1).
compute temp = (n*sscp(dat))-
(t(csum(dat))*(csum(dat))).
compute temp = temp/(n*(n-1)).
compute temp = csum(((diag(temp))=0)).
compute temp2=1.
do if (ws = 1).

```

```

compute temp2 = (n*sscp(tmp2))-
(t(csum(tmp2))*(csum(tmp2))).
compute temp2 = temp2/(n*(n-1)).
end if.
do if ((temp > 0 and ws = 0) or (temp > 0 and
ws = 1 and temp2 > 0)).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 27.
end if.
compute nmeds = ncol(mnames).
compute mcmats=ident(nmeds*2).
compute mccoef=make(nmeds*2,1,0).
compute sobel = make(nmeds,4,-999).
do if (model = 6 and nmeds > 4).
compute errs = errs+1.
compute runerrs(errs,1)=2.
end if.
do if (model < 4 and nmeds > 1).
compute errs = errs+1.
compute runerrs(errs,1) = 3.
end if.
compute nmods = (model = 74).
compute bad = 0.
compute intcnt = 1.
compute modvals = 0.
compute modvalsd = 0.
compute yintemp = {'int_1', 'int_2', 'int_3',
'int_4', 'int_5', 'int_6', 'int_7', 'int_8', 'int_9',
'int_10', 'int_11', 'int_12', 'int_13', 'int_14',
'int_15'}.
compute yintemp = {yintemp, 'int_16', 'int_17',
'int_18', 'int_19', 'int_20', 'int_21', 'int_22',
'int_23', 'int_24', 'int_25', 'int_26', 'int_27',
'int_28'}.
compute yintemp = {yintemp, 'int_29', 'int_30',
'int_31', 'int_32', 'int_33', 'int_34', 'int_35',
'int_36', 'int_37', 'int_38', 'int_39', 'int_40'}.

```

```

compute yintemp = {yintemp, 'int_41', 'int_42',
'int_43', 'int_44', 'int_45', 'int_46', 'int_47',
'int_48', 'int_49', 'int_50', 'int_51', 'int_52'}.
compute yintemp = {yintemp, 'int_53', 'int_54',
'int_55', 'int_56', 'int_57', 'int_58', 'int_59',
'int_60', 'int_61', 'int_62', 'int_63', 'int_64'}.
compute
cntname={('C1'),('C2'),('C3'),('C4'),('C5'),('C6'),('
C7'),('C8'),('C9'),('C10'),('C11'),('C12'),('C13'),('
C14'),('C15'),('C16'),('C17')}.
compute
cntname={cntname;('C18'),('C19'),('C20'),('C21
'),('C22'),('C23'),('C24'),('C25'),('C26'),('C27'),('
C28'),('C29'),('C30'),('C31')}.
compute
cntname={cntname;('C32'),('C33'),('C34'),('C35
'),('C36'),('C37'),('C38'),('C39'),('C40'),('C41'),('
C42'),('C43'),('C44'),('C45')}.
compute
cntname={cntname;('C46'),('C47'),('C48'),('C49
'),('C50'),('C51'),('C52'),('C53'),('C54'),('C55'),('
C56'),('C57'),('C58'),('C59')}.
compute
cntname={cntname;('C60'),('C61'),('C62'),('C63
'),('C64'),('C65'),('C66'),('C67'),('C68'),('C69'),('
C70'),('C71'),('C72'),('C73')}.
compute
cntname={cntname;('C74'),('C75'),('C76'),('C77
'),('C78'),('C79'),('C80'),('C81'),('C82'),('C83'),('
C84'),('C85'),('C86'),('C87')}.
compute
cntname={cntname;('C88'),('C89'),('C90'),('C91
'),('C92'),('C93'),('C94'),('C95'),('C96'),('C97'),('
C98'),('C99'),('C100'),('C101')}.
compute
cntname={cntname;('C102'),('C103'),('C104'),('
C105')}.
compute apathnam={'a path', 'a1 path', 'a2
path', 'a3 path', 'a4 path', 'a5 path', 'a6 path',
'a7 path', 'a8 path', 'a9 path', 'a10 path'}.

```

```

compute bpathnam={'b path'; 'b1 path'; 'b2
path'; 'b3 path'; 'b4path'; 'b5path'; 'b6 path'; 'b7
path'; 'b8 path'; 'b9 path'; 'b10 path'}.
compute modvnm = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute modvnm2 = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute mlab = { ' M1 ='; ' M2 ='; ' M3 ='; '
M4 ='; ' M5 ='; ' M6 ='; ' M7 ='; ' M8 ='; '
M9 ='; ' M10 ='}.
compute m = make(n,nmeds,1).
compute mmat = make(16,nmeds,0).
compute ymat = make(8,nmeds,0).
compute deco = make(10,1,0).
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
compute iterate = abs(trunc( 10000 )).
compute converge = abs( 0.00000001 ).
compute boot = abs(trunc( 5000 )).
do if (ws=1 and 0 > 0).
compute note(notes,1)=17.
compute notes=notes+1.
end if.
compute mc=(abs(trunc( 0 ))*(1-ws)).
do if (mc > 0 and model > 5).
do if (boot = 0).
compute boot=mc.
end if.
compute mc=0.
compute note(notes,1)=12.
compute notes = notes+1.
end if.
compute adjust = 0.
do if (mc > 0 and (model > 3 and model < 6)).
compute boot=0.
compute bconoff=0.
end if.
do if (boot > 0 and mc > 0).

```

```

compute mc = 0.
end if.
compute savboot = 0.
compute saveboot = ( 0 = 1).
do if (ws=1 and saveboot=1).
compute saveboot=0.
compute note(notes,1) = 20.
compute notes = notes + 1.
compute normal=0.
end if.
do if (saveboot =1 and boot > 0 and model >
3).
compute savboot = 1.
end if.
do if (boot <> 0 or mc <> 0).
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
loop.
compute cilow = rnd(bootsz*(1-(conf/100))/2).
compute cihigh =
trunc((bootsz*(conf/100)+(bootsz*(1-
(conf/100))/2)))+1.
do if (cilow < 1 or cihigh > bootsz).
compute
bootsz=trunc((bootsz+1000)/1000)*1000.
compute adjust = 1.
end if.
end loop if (cilow gt 0 and cihigh le bootsz).
compute boot=bootsz.
do if (mc > 0).
compute mc = bootsz.
end if.
do if (boot > 0 and mc > 0 and (model > 3 and
model < 6)).
compute boot=0.
end if.
do if (adjust = 1 and boot > 0).
compute note(notes,1)=6.

```



```

compute notes = notes+1.
end if.
do if (adjust = 1 and mc > 0).
compute note(notes,1)=13.
compute notes = notes+1.
end if.
end if.
do if (model = 6 and nmeds > 1).
compute mmpaths =
make((nmeds+2),(nmeds+2),0).
do if (nmeds = 2).
compute indboot = make(boot+1, 3, 999).
else if (nmeds = 3).
compute indboot = make(boot+1, 7, 999).
else if (nmeds = 4).
compute indboot = make(boot+1, 15, 999).
end if.
compute indlbl = {'Total:','Ind1 :','Ind2 :','Ind3
:','Ind4 :','Ind5 :','Ind6 :','Ind7 :','Ind8 :','Ind9
:','Ind10:','Ind11:','Ind12:','Ind13:','Ind14:','
Ind15:'}.
compute indlbl2 = {'Ind1','Ind2','Ind3','Ind4';
'Ind5','Ind6','Ind7','Ind8','Ind9','Ind10';
'Ind11','Ind12','Ind13','Ind14','Ind15'}.
compute indces = make(boot+1, 4, 999).
end if.
do if (model < 4).
compute boot = 0.
compute cmat = make(10,1,0).
compute zmat = make(10,1,0).
end if.
compute nvarch = make(1,ncol(dat),0).
compute wmatch = 0.
compute zmatch = 0.
compute vmatch = 0.
compute qmatch = 0.
compute mmatch=0.
compute minprobe=0.
compute maxprobe=0.
loop i = 1 to ncol(vnames).

```

```

do if (vnames(:,i)=yname).
compute y = dat(:,i).
compute nvarch(1,i)=1.
compute yerr = 0.
do if ((yname = xname) or (yname = wname) or
(yname = zname) or (yname = vname) or
(yname = qname)).
compute errs = errs+1.
compute runerrs(errs,1)=17.
end if.
end if.
do if (vnames(:,i)=xname).
compute x = dat(:,i).
compute nvarch(1,i)=1.
compute xdich = 1.
compute xerr=0.
loop jj = 1 to n.
do if ((x(jj,1) <> cmin(x)) and (x(jj,1) <>
cmax(x))).
compute xdich = 0.
break.
end if.
end loop.
compute xmean = csum(x)/n.
do if (center = 1 and (model < 4 or xlist > 0)
and mcx = 0).
compute meanvec = make(n,1, xmean).
compute x = x-meanvec.
compute centvar = {centvar, xname}.
end if.
compute xmean = csum(x)/n.
compute tmp = x-(cons*xmean).
compute xsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
do if (xdich = 0).
compute quantc(1,6) = 1.
compute matx = {xmean-xsd; xmean;
xmean+xsd}.
do if ( 999 = 999 and quantile = 0).
do if ((matx(1,1) < cmin(x)) and model = 74).
compute matx(1,1)=cmin(x).

```

```

compute minprobe=1.
end if.
do if ((matx(3,1) > cmax(x)) and model = 74).
compute matx(3,1)=cmax(x).
compute maxprobe=1.
end if.
end if.
do if (quantile = 1).
compute quantd(1,6) = 1.
compute quantc(1,6) = 0.
compute tmp = x.
compute tmp(GRADE(x)) = x.
compute matx =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (xdich = 1).
compute matx = {cmin(x); cmax(x)}.
do if (model=74).
compute matx=cmin(x).
compute mod74dic=1.
end if.
end if.
do if ( 999 <> 999).
compute matx = 999.
compute quantd(1,6) = 0.
compute quantc(1,6) = 0.
end if.
end if.
do if (werr = 0 and wlist = 1).
do if (vnames(:,i)=wname).
compute werr = 0.
compute wmatch = 1.
compute w = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(w)/n).
compute w = w-meanvec.
compute centvar = {centvar, wname}.

```

```

end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute wmean = csum(w)/n.
compute tmp = w-(cons*wmean).
compute wsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute wdich = 1.
loop jj = 1 to n.
do if ((w(jj,1) <> cmin(w)) and (w(jj,1) <>
cmax(w))).
compute wdich = 0.
break.
end if.
end loop.
do if (model = 3).
compute jndich=wdich.
compute jnmin=cmin(w).
compute jnmax=cmax(w).
end if.
do if (wdich = 0).
compute matw = {wmean-wsd; wmean;
wmean+wsd}.
do if ( 999 = 999 and quantile = 0).
do if (matw(1,1) < cmin(w)).
compute matw(1,1)=cmin(w).
compute minprobe=1.
end if.
do if (matw(3,1) > cmax(w)).
compute matw(3,1)=cmax(w).
compute maxprobe=1.
end if.
end if.
compute quantc(1,1) = 1.
do if (quantile = 1).
compute quantd(1,1) = 1.
compute quantc(1,1) = 0.
compute tmp = w.
compute tmp(GRADE(w)) = w.
compute matw =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);

```

```

tmp(trunc(n*.50),1); tmp(trunc(n*0.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (wdich = 1).
compute matw = {cmin(w); cmax(w)}.
compute wvdich=1.
compute cmaxw=cmax(w).
compute cminw=cmin(w).
end if.
do if ( 999 <> 999).
compute matw = 999.
compute quantd(1,1) = 0.
compute quantc(1,1) = 0.
end if.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm(1,1)=wname.
compute modmatp(1,1) = 1.
end if.
end if.
do if (zerr = 0 and zlist = 1).
do if (vnames(:,i)=zname).
compute zerr = 0.
compute zmatch = 1.
compute z = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(z)/n).
compute z = z-meanvec.
compute centvar = {centvar, zname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute zmean = csum(z)/n.
compute tmp = z-(cons*zmean).
compute zsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute zdich = 1.
loop jj = 1 to n.
do if ((z(jj,1) <> cmin(z)) and (z(jj,1) <>
cmax(z))).

```

```

compute zdich = 0.
break.
end if.
end loop.
do if (zdich = 0).
compute matz = {zmean-zsd; zmean;
zmean+zsd}.
do if ( 999 = 999 and quantile = 0).
do if (matz(1,1) < cmin(z)).
compute matz(1,1)=cmin(z).
compute minprobe=1.
end if.
do if (matz(3,1) > cmax(z)).
compute matz(3,1)=cmax(z).
compute maxprobe=1.
end if.
end if.
compute quantc(1,2) = 1.
do if (quantile = 1).
compute quantd(1,2) = 1.
compute quantc(1,2) = 0.
compute tmp = z.
compute tmp(GRADE(z)) = z.
compute matz = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (zdich = 1).
compute matz = {cmin(z); cmax(z)}.
compute cmaxz=cmax(z).
compute cminz=cmin(z).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matz = 999.
compute quantd(1,2) = 0.
compute quantc(1,2) = 0.
end if.
compute modmatv(1,2)=nrow(matz).

```

```

compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm(1,2)=zname.
compute modmatp(1,2) = 1.
end if.
end if.
do if (verr = 0 and vlist = 1).
do if (vnames(:,i)=vname).
compute verr = 0.
compute vmatch = 1.
compute v = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(v)/n).
compute v = v-meanvec.
compute centvar = {centvar, vname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute vmean = csum(v)/n.
compute tmp = v-(cons*vmean).
compute vsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute vdich = 1.
loop jj = 1 to n.
do if ((v(jj,1) <> cmin(v)) and (v(jj,1) <>
cmax(v))).
compute vdich = 0.
break.
end if.
end loop.
do if (vdich = 0).
compute matv = {vmean-vsd; vmean;
vmean+vsd}.
do if ( 999 = 999 and quantile = 0).
do if (matv(1,1) < cmin(v)).
compute matv(1,1)=cmin(v).
compute minprobe=1.
end if.
do if (matv(3,1) > cmax(v)).
compute matv(3,1)=cmax(v).
compute maxprobe=1.
end if.

```

```

end if.
compute quantc(1,3) = 1.
do if (quantile = 1).
compute quantd(1,3) = 1.
compute quantc(1,3) = 0.
compute tmp = v.
compute tmp(GRADE(v)) = v.
compute matv = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (vdich = 1).
compute matv = {cmin(v); cmax(v)}.
compute wvdich=1.
compute cmaxv=cmax(v).
compute cminv=cmin(v).
end if.
do if ( 999 <> 999).
compute matv = 999.
compute quantd(1,3) = 0.
compute quantc(1,3) = 0.
end if.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm(1,3)=vname.
compute modmatp(1,3) = 1.
end if.
end if.
do if (qerr = 0 and qlist = 1).
do if (vnames(:,i)=qname).
compute qerr = 0.
compute qmatch = 1.
compute q = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(q)/n).
compute q = q-meanvec.
compute centvar = {centvar, qname}.
end if.
compute nvarch(1,i)=1.

```



```

compute nmods = nmods + 1.
compute qmean = csum(q)/n.
compute tmp = q-(cons*qmean).
compute qsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute qdich = 1.
loop jj = 1 to n.
do if ((q(jj,1) <> cmin(q)) and (q(jj,1) <>
cmax(q))).
compute qdich = 0.
break.
end if.
end loop.
do if (qdich = 0).
compute matq = {qmean-qsd; qmean;
qmean+qsd}.
do if ( 999 = 999 and quantile = 0).
do if (matq(1,1) < cmin(q)).
compute matq(1,1)=cmin(q).
compute minprobe=1.
end if.
do if (matq(3,1) > cmax(q)).
compute matq(3,1)=cmax(q).
compute maxprobe=1.
end if.
end if.
compute quantc(1,4) = 1.
do if (quantile = 1).
compute quantd(1,4) = 1.
compute quantc(1,4) = 0.
compute tmp = q.
compute tmp(GRADE(q)) = q.
compute matq = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (qdich = 1).
compute matq = {cmin(q); cmax(q)}.
compute cmaxq=cmax(q).
compute cminq=cmin(q).

```

```

compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matq = 999.
compute quantd(1,4) = 0.
compute quantc(1,4) = 0.
end if.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm(1,4)=qname.
compute modmatp(1,4) = 1.
end if.
end if.
do if (vnames(:,i)=cname).
compute cld = dat(:,i).
compute cvname = vnames(:,i).
compute nvarch(1,i)=1.
compute clsmatch = 1.
end if.
loop j = 1 to ncol(mnames).
do if (vnames(:,i)=mnames(1,j)).
compute mmatch = mmatch + 1.
compute m(:,j)=dat(:,i).
do if (center = 1 and nvarch(1,i) = 0 and mlist >
0 and mcm = 0).
compute meanvec = make(n,1,csum(m(:,j))/n).
compute m(:,j) = m(:,j)-meanvec.
compute centvar = {centvar, mnames(1,j)}.
end if.
compute nvarch(1,i)=1.
compute dichm = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute dichm = 0.
break.
end if.
end loop.
do if ((dichm = 1) and (olstdichm = 1)).
compute note(notes,1) = 18.

```

```

compute notes = notes + 1.
end if.
do if (dichm = 1 and model > 3 and mcheck = 0
and oldsdichm = 0).
compute errs= errs+1.
compute runerrs(errs,1) = 1.
compute mcheck = 1.
end if.
do if ((model <= 3) and (ncol(mnames)=1)).
compute nmods = nmods + 1.
compute mmean = csum(m(:,j))/n.
compute tmp = m(:,j)-(cons*mmean).
compute msd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute mdich = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute mdich = 0.
break.
end if.
end loop.
do if (model = 1).
compute jndich=mdich.
compute jnmin=cmin(m(:,j)).
compute jnmax=cmax(m(:,j)).
end if.
do if (mdich = 0).
compute matm = {mmean-msd; mmean;
mmean+msd}.
do if ( 999 = 999 and quantile = 0).
do if (matm(1,1) < cmin(m(:,j))).
compute matm(1,1)=cmin(m(:,j)).
compute minprobe=1.
end if.
do if (matm(3,1) > cmax(m(:,j))).
compute matm(3,1)=cmax(m(:,j)).
compute maxprobe=1.
end if.
end if.
compute quantc(1,5) = 1.

```

```

do if (quantile = 1).
compute quantd(1,5) = 1.
compute quantc(1,5) = 0.
compute tmp = m(:,j).
compute tmp(GRADE(m(:,j))) = m(:,j).
compute matm = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (mdich = 1).
compute matm = {cmin(m); cmax(m)}.
end if.
do if ( 999 <> 999).
compute matm = 999.
compute quantd(1,5) = 0.
compute quantc(1,5) = 0.
end if.
compute modmatv(1,5)=nrow(matm).
compute modmat((1:nrow(matm)), 5) = matm.
compute modvnm(1,5)=mnames(1,j).
compute modmatp(1,5) = 1.
end if.
end if.
end loop.
end loop.
do if (minprobe=1).
compute note(notes,1) = 14.
compute notes = notes + 1.
end if.
do if (maxprobe=1).
compute note(notes,1) = 15.
compute notes = notes + 1.
end if.
do if (cname <> 'xxx' and clsmatch = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 23.
end if.
do if (cname <> 'xxx').

```

```

do if ((cname = zname) or (cname = vname)
or (cname = wname) or (cname = xname) or
(cname = yname) or (cname = qname)).
compute errs = errs+1.
compute runerrs(errs,1) = 24.
end if.
end if.
do if (wlist = 1 and werr = 0 and wmatch = 0).
compute werr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (zlist = 1 and zerr = 0 and zmatch = 0).
compute zerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (qlist = 1 and qerr = 0 and qmatch = 0).
compute qerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (vlist = 1 and verr = 0 and vmatch = 0).
compute verr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (yerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 16.
end if.
do if (xerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 32.
end if.
do if (model = 6 and nmeds < 2).
compute errs = errs+1.
compute runerrs(errs,1) = 18.
end if.
do if (mmatch < ncol(mnames)).

```

```

compute errs=errs+1.
compute runerrs(errs,1) = 25.
end if.
do if (cname <> 'xxx').
compute cld = design(cld).
compute cluster = ncol(cld).
compute cld = cld(:,2:ncol(cld)).
compute clsdmy = ncol(cld).
do if (clsdmy > 19).
compute errs = errs+1.
compute runerrs(errs,1) = 26.
end if.
do if (mcx > 0 or mcm > 0).
compute errs = errs+1.
compute runerrs(errs,1) = 38.
end if.
end if.
compute dichy = 1.
loop jj = 1 to n.
do if ((y(jj,1) <> cmin(y)) and (y(jj,1) <>
cmax(y))).
compute dichy = 0.
break.
end if.
end loop.
do if ((dichy = 1) and (olsdichy = 1)).
compute dichy=0.
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichy = 1).
do if (toteff=1).
compute toteff=0.
compute note(notes,1) = 31.
compute notes = notes + 1.
end if.
compute jncrit=xp2*xp2.
do if (ws = 1).
compute criterr = 1.
compute errs = errs+1.

```

```

compute runerrs(errs,1) = 28.
end if.
end if.
compute ncovs = ncol(dat)-rsum(nvarch).
do if (effsize = 1 and covmy <> 0 and ncovs >
0 and model > 3 and model < 7).
compute note(notes,1) = 22.
compute notes = notes + 1.
compute effsize = 0.
end if.
do if (ws = 1 and ncovs < nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 29.
end if.
do if (ws = 1 and ncovs > nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 30.
end if.
end if.
do if (errs = 0).
do if (rsum(quantd) > 0).
compute note(notes,1) = 4.
compute notes = notes+1.
end if.
do if (rsum(quantc) > 0).
compute note(notes,1) = 5.
compute notes = notes+1.
end if.
do if (ncovs > 0).
compute c = make(n,ncovs,0).
compute cnames = {'x'}.
compute j = 1.
loop i = 1 to ncol(vnames).
do if (nvarch(1,i)) = 0.
compute c(:,j) = dat(:,i).
compute nvarch(1,i)=1.
compute j=j+1.
compute cnames = {cnames, vnames(:,i)}.

```

```

end if.
end loop.
compute cnames = cnames(1,2:ncol(cnames)).
do if (ws = 1).
compute covmean=csum(c)/n.
loop i = 1 to ncovs.
compute meanvec = make(n,1, covmean(1,i)).
compute c(:,i) = c(:,i)-meanvec.
end loop.
compute centvar = {centvar, cnames}.
compute x=c(:,1:ncovs).
do if (ncovs=nmeds).
compute ncovs=0.
end if.
end if.
end if.
compute names = {yname, xname, mnames,
wname, zname, vname, qname}.
do if (ncovs > 0).
compute names = {names, cnames}.
end if.
do if (dichy = 1 and effsize = 1).
compute note(notes,1) = 2.
compute notes = notes+1.
end if.
do if (model > 3 and model < 6).
compute indeff=make(nmeds,1,0).
compute indboot=make(boot+1,nmeds,999).
do if (mc > 0).
compute indboot=make(mc+1,nmeds,999).
end if.
do if (effsize = 1 and dichy = 0).
compute rmeff=make(boot+1,nmeds+1,999).
compute abpseff=make(boot+1,nmeds+1,999).
compute abcseff=make(boot+1,nmeds+1,999).
compute pmeff=make(boot+1,nmeds+1,999).
compute r245 = make(boot+1,1,999).
compute kappa2 = make(boot+1,1,999).
end if.
end if.

```



```

do if (model = 6 and effsize = 1 and dichy = 0).
compute
rmeff=make(boot+1,ncol(indboot),999).
compute
abpseff=make(boot+1,ncol(indboot),999).
compute
abcseff=make(boot+1,ncol(indboot),999).
compute
pmeff=make(boot+1,ncol(indboot),999).
end if.
do if (nmods > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm(1,tmp) = modvnm(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:nmods).
compute modvnm=modvnm(:,1:nmods).
compute modmatv=modmatv(:,1:nmods).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvals =
make((modmatv(1,1)*modprod(1,1)),
nmods,0).
loop i = 1 to nmods.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvals)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).

```

```

compute fnsh = fnsh+nrow(tmp).
compute modvals(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
do if (model = 74).
compute modvals=matx.
compute modvnm=xname.
end if.
compute vmat = make(8,nrow(modvals),0).
compute vmat(1,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute vmat(5,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute indeff=make(nrow(modvals),1,0).
do if (model <> 5).
compute indboot = make(((boot+1)*nmeds),
nrow(modvals), -99999999).
compute indbootp =
make(boot+1,(nmeds*nmods), -99999999).
do if (imm > 4 and imm < 8).
do if (imm = 7).
compute immmv1=matw.
compute immmv2=matv.
end if.
do if (imm = 5).
compute immmv1=matw.
compute immmv2=matz.
end if.
do if (imm = 6).
compute immmv1=matv.
compute immmv2=matq.
end if.
compute
indboop1=make((boot+1),(nmeds*(nrow(immm
v2))),-99999999).
compute
indboop2=make((boot+1),(nmeds*(nrow(immm
v1))),-99999999).

```

```

compute onesmv1=make(1,nrow(immmv1),1).
compute onesmv2=make(1,nrow(immmv2),1).
compute
nimvs=nrow(immmv1)+nrow(immmv2).
compute cmmval=0.
compute cmmlbs1={'xx'}.
compute cmmlbs2={'xx'}.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv2}.
compute
cmmlbt=make(nrow(immmv2),1,mnames(1,i)).
compute cmmlbs1={cmmlbs1;cmmlbt}.
end loop.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv1}.
compute
cmmlbt=make(nrow(immmv1),1,mnames(1,i)).
compute cmmlbs2={cmmlbs2;cmmlbt}.
end loop.
compute cmmval=cmmval(2:nrow(cmmval),:).
end if.
end if.
end if.
do if (nmods> 0).
loop i = 1 to ncol(modvals).
do if (modvnm(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm(1,i)=qname).
compute qcol = i.
end if.
end loop.
end if.
do if (dichy = 1).

```

```

compute omx = cmax(y).
compute omn = cmin(y).
compute y = (y = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute data = {cons,y,m,x}.
compute datamed = data.
compute datayed = data.
compute datanm = {'constant'; yname;
t(mnames); xname}.
compute datanmm = {'constant'; yname;
t(mnames); xname}.
compute datanmy = {'constant'; yname;
t(mnames); xname}.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
do if (model < 4 and errs = 0).
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' ', ' '}.
compute datayed = {datayed, x&*m}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(1,i) = 1.
compute vmat(2,i) = modvals(i,1).
end loop.
compute mmat = make(16,nmeds,1).
end if.
do if (model = 2 or model = 3).
compute int1 = x&*w.
compute datayed = {datayed, w, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,2).
compute vmat(3,i) = modvals(i,1).
compute vmat(4,i) = modvals(i,1)*modvals(i,2).

```

```

end loop.
end if.
do if (model = 3).
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames, ' X', wname, '', ''}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*m.
compute int2 = x&*w&*m.
compute datayed = {datayed, int1, int2}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' X', wname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (model = 4 or model = 5).
compute vmat = make(8,1,1).
end if.
compute yintkey2 = yintkey.
do if (wm = 1).
compute int1 = x&*w.
compute datamed = {datamed, w,int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, '', ''}.
compute datanmm = {datanmm; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,wcol).
end loop.
do if (zm = 1).
compute int1 = x&*z.
compute datamed = {datamed, z, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, '', ''}.
compute datanmm = {datanmm; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.

```

```

loop i = 1 to nrow(modvals).
compute vmat(3,i) = modvals(i,zcol).
end loop.
end if.
do if (wzm = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*z.
compute int2 = x&*w&*z.
compute datamed = {datamed, int1, int2}.
loop i = 1 to nrow(modvals).
compute vmat(4,i) =
modvals(i,wcol)*modvals(i,zcol).
end loop.
end if.
end if.
compute mdatacol=ncol(datamed).
compute mintkey = yintkey.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
compute medints = intcnt-1.
do if (vy = 1 or xmy = 1).
compute mp = 1.
do if (xmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,1).
end loop.
end if.
do if (vy = 1).
compute datayed = {datayed, v}.
compute datanmy = {datanmy; vname}.
compute mmodes = 1.
loop i = 1 to nrow(modvals).

```

```

compute vmat(6,i) = modvals(i,vcol).
end loop.
do if (qy = 1).
compute mp = 2.
compute datayed = {datayed,q}.
compute datanmy = {datanmy; qname}.
compute mmods = 2.
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,qcol).
end loop.
end if.
do if (vqy = 1).
compute mp = 3.
compute datayed = {datayed, v&*q}.
compute mmods = 3.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,vcol)*modvals(i,qcol).
end loop.
end if.
end if.
compute mints = make(n,(nmeds*mp),0).
loop i = 0 to (nmeds-1).
do if (i = 0 and vqy = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
vname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (xmy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))*x.

```

```

compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', xname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (qy = 1).
compute mints(:,((i*mp)+2))=m(:,(i+1))&*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqy = 1).
compute mints(:,((i*mp)+3))=m(:,(i+1))&*v&*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end loop.
compute datayed = {datayed, mints}.
end if.
compute mp = 1.
do if (wvmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,wcol)*modvals(i,vcol).
end loop.
end if.
compute mints2 = make(n,(nmeds*mp),0).
do if (wmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,wcol).
end loop.
do if (wy = 0 and model > 3).
compute datayed = {datayed, w}.

```



```

compute datanmy = {datanmy; wname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wvmy = 1).
compute datayed = {datayed,w&*v}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints2(:,((i*mp)+1))= m(:,(i+1))&*w.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wvmy = 1).
compute mints2(:,((i*mp)+2))=
m(:,(i+1))&*w&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints2}.
end if.
compute mp = 1.
do if (zmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,zcol).
end loop.
do if (wzmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,zcol)&*modvals(i,wcol).
end loop.

```

```

end if.
end if.
do if (zmy = 1).
compute mints3 = make(n,(nmeds*mp),0).
do if (zy = 0).
compute datayed = {datayed, z}.
compute datanmy = {datanmy; zname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wzmy = 1 and wzy = 0).
compute datayed = {datayed,w&*z}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints3(:,((i*mp)+1))= m(:,(i+1))&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wzmy = 1).
compute mints3(:,((i*mp)+2))=
m(:,(i+1))&*w&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', zname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints3}.
end if.
compute decoc = 1.
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.

```

```

do if (wy = 1 and model > 3).
compute datayed = {datayed, w, x&*w}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm2(1,1)=wname.
compute modmatp(1,1) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (zy = 1).
compute datayed = {datayed,z,x&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm2(1,2)=zname.
compute modmatp(1,2) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (wzy = 1).
compute datayed = {datayed,w&*z,x&*w&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.

```

```

compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vxy = 1).
compute datayed = {datayed, x&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm2(1,3)=vname.
compute modmatp(1,3) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, ',', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (qxy = 1).
compute datayed = {datayed, x&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm2(1,4)=qname.
compute modmatp(1,4) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', qname, ',', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqxy = 1).
compute datayed = {datayed, x&*v&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, 'X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.

```

```

end if.
end if.
do if (wvxy = 1).
compute datayed = {datayed, x&*w&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, 'X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute modvalsd = 0.
compute ttt = rsum(modmatp).
do if (rsum(modmatp) > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm2(1,tmp) = modvnm2(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:ttt).
compute modvnm2=modvnm2(:,1:ttt).
compute modmatv=modmatv(:,1:ttt).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvalsd =
make((modmatv(1,1)*modprod(1,1)), ttt,0).
loop i = 1 to ttt.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvalsd)).

```

```

loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvalsd(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
end if.
do if (ttt > 0).
loop i = 1 to ncol(modvalsd).
do if (modvnm2(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm2(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm2(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm2(1,i)=qname).
compute qcol = i.
end if.
end loop.
compute directv = make(nrow(modvalsd),1,1).
do if (wy = 1).
compute directv ={directv, modvalsd(:,wcol)}.
end if.
do if (zy = 1).
compute directv ={directv, modvalsd(:,zcol)}.
end if.
do if (wzy = 1).
compute directv ={directv,
(modvalsd(:,wcol)&*modvalsd(:,zcol))}.
end if.
do if (vxy = 1).
compute directv ={directv, modvalsd(:,vcol)}.
end if.
do if (qxy = 1).

```

```

compute directv = {directv, modvalsd(:,qcol)}.
end if.
do if (vqxy = 1).
compute directv = {directv,
(modvalsd(:,vcol)*modvalsd(:,qcol))}.
end if.
do if (wvxy = 1).
compute directv = {directv,
(modvalsd(:,vcol)*modvalsd(:,wcol))}.
end if.
end if.
compute ydatacol = ncol(datayed).
do if (ncovs > 0).
do if (covmy <> 2).
compute datamed = {datamed,c}.
end if.
do if (covmy <> 1).
compute datayed = {datayed,c}.
end if.
do if (covmy = 1 and model = 4 and mcx > 0).
compute datayed = {datayed,c}.
end if.
compute covmeans = csum(c)/n.
end if.
do if (cluster > 0).
compute datamed = {datamed,cld}.
compute datayed = {datayed, cld}.
compute cldmeans = csum(cld)/n.
end if.
compute mst = 3.
compute mnd = mst+nmeds-1.
compute ydatacol = ncol(datayed).
compute mdatacol = ncol(datamed).
do if (ncovs > 0).
compute datanmy = {datanmy; t(cnames)}.
do if (model > 3).
compute datanmm = {datanmm; t(cnames)}.
end if.
end if.

```

```

compute datanmy = {'constant';
datanmy(3:nrow(datanmy),1)}.
do if (model > 3).
compute datanmm = {'constant';
datanmm(3:nrow(datanmm),1)}.
end if.
compute amm = make(2,1,0).
compute abmm = make(2,1,0).
compute mnv = csum(datayed(:,2)/n).
compute mnv = make(n,1,mnv).
compute ssty = csum((datayed(:,2)-mnv)&**2).
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
do if (ws=1).
compute stddevm=diag(sigma).
compute
stddevm=sqrt(stddevm(3:(nmeds+2),1)).
end if.
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute r2xy =
(sigma(2,(3+nmeds))/(stddevy*stddevx))&**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))&**2.
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).
do if (model = 4 and nmeds = 1 and cluster = 0
and ncovs = 0).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).

```



```

compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
compute datatm = datamed.
compute dataty = datayed.
compute mdlrms2 = { '4' ; yname; xname}.
compute mdlrms = {'Model ='; ' Y ='; ' X ='
}.
do if (ws = 1).
compute mdlrms2 = { '4' ; yname}.
compute mdlrms = {'Model ='; ' Y ='}.
end if.
loop i = 1 to ncol(mnames).
compute mdlrms2 = {mdlrms2; mnames(1,i)}.
do if (i = 1 and ncol(mnames) = 1).
compute mdlrms = {mdlrms; ' M ='}.
else.
compute mdlrms = {mdlrms; mlab(i,1)}.
end if.
end loop.
do if (wname <> 'xxx').
compute mdlrms2 = {mdlrms2; wname}.
compute mdlrms = {mdlrms; ' W ='}.
end if.
do if (zname <> 'xxx').
compute mdlrms2 = {mdlrms2; zname}.
compute mdlrms = {mdlrms; ' Z ='}.

```

```

end if.
do if (vname <> 'xxx').
compute mdlms2 = {mdlms2; vname}.
compute mdlms = {mdlms; '  V = '}.
end if.
do if (qname <> 'xxx').
compute mdlms2 = {mdlms2; qname}.
compute mdlms = {mdlms; '  Q = '}.
end if.
do if (jn = 1 and model = 1 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
do if (jn = 1 and model = 3 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
compute mdpbe=0.
compute medte2=0.
do if ( 4 =4 and mcx > 0).
compute medte2=1.
end if.
do if (((mcx > 0) or (mcm > 0)) AND model =
1).
compute mdpbe=1.
compute dd={datayed(:,4),datayed(:,3)}.
do if (ncovs > 0).
compute dd={datayed(:,6:(5+ncovs)),dd}.
end if.
compute dd={datayed(:,2),dd}.
end if.
compute yes = ((nrow(modvals)=1) and
(contrast = 1) and (model = 3 or model = 2)).
compute yes2 = ((nrow(modvals)=1) and
(contrast = 1) and (model = 2)).
print/title = '***** PROCESS Procedure
for SPSS Release 2.16.3 *****'.
print/title = '      Written by Andrew F. Hayes,
Ph.D.      www.afhayes.com'.

```

```

print/title = ' Documentation available in
Hayes (2013).
www.guilford.com/p/hayes3'/space=0.
print mdlrms2/title =
*****
*****'/rnames = mdlrms/format =
a8.
do if (ncovs > 0).
print cnames/title = 'Statistical Controls: '/rlabels
= 'CONTROL=' /format a8.
end if.
print n/title = 'Sample size' /format F10.0.
do if (cluster > 0).
print cluster/rnames = cvname/title =
'Clustering Variable and Number of Clusters'.
end if.
do if (model > 3 and ( 'random' <> 'random')).
compute seedt= 'random'.
print seedt/title='Custom' /format=A12/rlabel =
'Seed:'.
end if.
do if (mdpbe=0 and medte2 = 0).
loop bt = 1 to (boot+1).
do if (bt = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef = 0.
do if (bt > 1).
loop.
compute v=trunc(uniform(n,1)*n)+1.
compute datayed = dataty(v,:).
do if (ws=1).
compute
bmeans=csum(datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed)))/n.
compute cntmat=make(n,nmeds,1).
compute bmeans2=bmeans.
do if (nmeds > 1).
compute bmeans2=mdiag(bmeans).

```

```

end if.
compute cntmat=cntmat*bmeans2.
compute datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed))=datayed(:,(ncol(data
yed)-nmeds+1):ncol(datayed))-cntmat.
end if.
compute detcheck=det(t(datayed)*datayed).
compute rk=(detcheck <> 0).
do if (model > 3).
compute datamed = datatm(v,:).
compute detcheck=det(t(datamed)*datamed).
do if (rk=1).
compute rk=(detcheck <> 0).
end if.
end if.
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
compute temp=diag(sigma).
compute bad = bad+(1-rk).
compute false = 1.
end loop if (rk = 1).
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nme
ds)).
do if (model = 4 and nmeds = 1 and ncovs = 0
and cluster = 0).
compute r2xy =
(sigma(2,4)/(stddevy*stddevx))&**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))&**2.
compute sstot = sigma(2,2)*(n-1).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).

```

```

compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
end if.
do if (model > 3).
loop im = 1 to nmeds.
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant';
datanmm((2+nmeds):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
do if (model = 6).
do if (im = 1).
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
end if.
do if (im > 1).

```

```

compute xm={cons, datamed(:,3:(im+1)),
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant'; datanmm(2:im,1);
datanmm((mnd):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
compute
mmpaths((im+1),(2:im))=t(coeff(2:im,1)).
end if.
end if.
do if (ws = 1).
compute
coeff(1,1)=csum(datamed(:,(2+im)))/n.
end if.
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
do if (bt = 1).
compute resid=datamed(:,(2+im))-xm*coeff.
compute sse =cssq(resid).
compute mse = sse/(n-ncol(xm)).
compute mnv = csum(data(:,(2+im)))/n.
compute mnv = make(n,1,mnv).
compute sstm = csum((data(:,(2+im))-
mnv)&^2).
compute k3 = nrow(coeff).
do if (hc3 = 1).
compute h = xm(:,1).
loop i3=1 to n.
compute h(i3,1)= xm(i3,:)*invXtX*t(xm(i3,:)).
end loop.
loop i3=1 to k3.
compute xm(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xm(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xm(:,i3) = sqrt(mse)&*xm(:,i3).

```

```

end loop.
end if.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute hccov=invXtX*t(xm)*xm*invXtX.
compute mcmats(im,im)=hccov(2,2).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*hccov*lmat)*((t(lmat)
)*coeff))/dfnum.
compute coeff = coeff(1:(nrow(coeff)-
clsdmy),1).
compute mcoeff(im,1)=coeff(2,1).
compute standerr =
sqrt(diag(invXtX*t(xm)*xm*invXtX)).
compute standerr = standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1).
compute standerr(1,1)=stddevm(im,1)/sqrt(n).
end if.
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-
ncol(xm)))).
compute temp=(n-ncol(xm)).
do if (ws=1).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute temp=(n-1).
end if.
compute xd = abs(xp2).
compute temp = (temp* (exp((temp-
(5/6))*((xd/(temp-(2/3)+(.11/temp)))*(xd/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = coeff-
sqrt(abs(temp))*standerr.
compute temp2 =
coeff+sqrt(abs(temp))*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (ws=1).

```

```

compute op=op(1,:).
compute xnmn =
apathnam(((nmeds>1)+1+(im-1)),1).
end if.
compute sobel(im,1) = coeff(2,1).
compute sobel(im,2)=standerr(2,1).
compute temp = mnames(1,im).
compute r2full = 1-(sse/sstm).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute summ = {sqrt(r2full), r2full, mse,
Fratio, dfnum, dfden, pfr}.
do if (detail = 1).
print temp/title =
*****
*****'/rlabels = 'Outcome:'/format
= A8.
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels 'R',
'R-sq', 'MSE','F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
print op/title = 'Model'/rnames = xnmn/clabels
'coeff', 'se', 't', 'p', 'LLCI', 'ULCI'/format = F10.4.
do if (covcoeff=1 and ws <> 1).
compute
hccovtmp=hccov(1:nrow(op),1:nrow(op)).
compute cnamestp=t(xnmn).
print hccovtmp/title='Covariance matrix of
regression parameter
estimates'/rnames=xnmn/cnames=cnamestp/f
ormat= F10.4.
end if.
do if (nmods > 0 and nrow(mintkey) > 1).
print mintkey/title = 'Product terms key:'/format
= A8.
end if.
do if (ws = 1).
compute wsdf=n-1.

```



```

print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
end if.
end if.
compute ymat(1,im) = coeff(2,1).
do if (ws = 1).
compute ymat(1,im)=coeff(1,1).
end if.
do if (wm = 1).
compute ymat(2,im) = coeff(4,1).
do if (zm = 1).
compute ymat(3,im) = coeff(6,1).
do if (wzm = 1).
compute ymat(4, im) = coeff(8,1).
end if.
end if.
end if.
do if (model = 6).
compute mmpaths((im+1),1)=coeff((im+1),1).
end if.
end loop.
end if.
loop totlp = 1 to (1+(toteff*(bt = 1))).
do if (toteff = 1 and totlp =2).
compute xy =
{cons,datayed(:,(3+nmeds):ydatacol)}.
else.
compute xy={cons, datayed(:,3:ydatacol)}.
end if.
do if (dichy = 1).
compute pt2 =
make(nrow(datayed(:,2)),1,(csum(datayed(:,2))
/n)).
compute LL3 = datayed(:,2)*ln(pt2)+(1-
datayed(:,2))*ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xy),1,0).
compute LL1 = 0.
compute xy22=xy.

```

```

loop jjj = 1 to iterate.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.
compute coeff =
bt1+inv(t(xy22)*xy)*t(xy)*(datayed(:,2)-pt1).
compute pt1 = 1/(1+exp(-(xy*coeff))).
compute itprob = csum((pt1 <
.00000000000001) or (pt1 >
.99999999999999)).
do if (itprob = 0).
compute LL = datayed(:,2)*ln(pt1)+(1-
datayed(:,2))*ln(1-pt1).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
break.
end if.
compute bt1 = coeff.
compute LL1 = LL2.
end loop.
do if (jjj >= iterate and iterr = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
compute iterr = 1.
end if.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.
compute covmat = inv(t(xy22)*xy).
release xy22.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
end if.
do if (dichy = 0).
compute invXtX = inv(t(xy)*xy).
compute coeff =invXtX*t(xy)*datayed(:,2).

```

```

do if (effsize=1 and model > 3 and model < 7
and (ncovs > 0 or clsdmy > 0)).
compute xysdy =
{cons,datayed(:,(4+nmeds):ydatacol)}.
compute coeffsd
=inv(t(xysdy)*xysdy)*t(xysdy)*datayed(:,2).
compute resid=datayed(:,2)-(xysdy*coeffsd).
compute sdy cov =sqrt(csum(resid*&resid)/(n-
ncol(xysdy))).
compute xvaron=datayed(:,(3+nmeds)).
compute
coeffx2=inv(t(xysdy)*xysdy)*t(xysdy)*xvaron.
compute residx2=xvaron-(xysdy*coeffx2).
compute sdx cov
=sqrt(csum(residx2*&residx2)/(n-ncol(xysdy))).
end if.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4 and bt > 1).
compute resid=datayed(:,2)-xy*coeff.
compute sse = cssq(resid).
compute r2full = 1-(sse/sstot).
end if.
do if (bt =1).
compute resid=data(:,2)-xy*coeff.
compute k3 = nrow(coeff).
compute sse = cssq(resid).
compute mse = sse/(n-ncol(xy)).
compute
hcnot=coeff&/(sqrt(diag(mse*invXtX))).
do if (hc3 = 1).
compute h = xy(:,1).
loop i3=1 to n.
compute h(i3,1)= xy(i3,:)*invXtX*t(xy(i3,:)).
end loop.
loop i3=1 to k3.

```

```

compute xy(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xy(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xy(:,i3) = sqrt(mse)&*xy(:,i3).
end loop.
end if.
compute covmat = (invXtX*t(xy)*xy*invXtX).
end if.
end if.
do if (bt = 1).
do if (model = 2).
compute xy2={cons, datayed(:,3:ydatacol)}.
compute temp = ncol(xy2).
do if (temp > 6).
compute xy3=xy2(:,7:temp)}.
end if.
compute xy2={xy2(:,1:3), xy2(:,5)}.
do if (temp > 6).
compute xy2={xy2, xy3}.
release xy3.
end if.
compute invXtX = inv(t(xy2)*xy2).
compute coeff2 =invXtX*t(xy2)*datayed(:,2).
compute ssem2=cssq(datayed(:,2)-xy2*coeff2).
release xy2.
end if.
compute standerr = sqrt(diag(covmat)).
do if (totlp =1).
compute
mcmats((nmeds+1):ncol(mcmats),(nmeds+1):n
col(mcmats))=covmat(2:(1+nmeds),2:(1+nmed
s)).
end if.
compute standerr=standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1 and totp=2).
compute standerr(1,1)=stddevy/sqrt(n).

```

```

end if.
compute coeffplt = coeff.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*covmat*lmat)*((t(lmat)*coeff))/dfnum).
compute coeff=coeff(1:(nrow(coeff)-clsdmy),1).
do if (totlp=1).
compute
mccoeff((nmeds+1):nrow(mccoeff))=coeff(2:(1+nmeds),1).
end if.
compute bbbb=coeff(2,1).
do if totp = 1.
compute deco(1,1)=2+nmeds.
compute deco = deco(1:decoc,1).
compute covdirt = make((nrow(covmat)-clsdmy),(ncol(covmat)-clsdmy),0).
compute covdirt = covmat(deco,:).
compute covdir =
make(nrow(covdirt),ncol(covdirt),0).
compute covdir = covdirt(:,t(deco)).
compute deco=coeff(deco,1).
do if (ttt > 0).
compute sedir =
sqrt(diag(directv*covdir*t(directv))).
compute directv = directv*deco.
end if.
compute sobel(:,3)=coeff(2:(1+nmeds),1).
compute sobel(:,4) = standerr(2:(1+nmeds),1).
compute sobel2 = sobel*sobel.
do if (varorder <> 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)*sobel2(:,4)+sobel2(:,3)*sobel2(:,2)).
end if.
do if (varorder = 2).

```

```

compute sobel(:,2) =
sqrt(sobel2(:,1)*sobel2(:,4)+sobel2(:,3)*sobel2(:,2)+sobel2(:,2)*sobel2(:,4)).
end if.
compute sobel(:,1) = sobel(:,1)*sobel(:,3).
compute sobel(:,3) = sobel(:,1)/sobel(:,2).
compute sobel(:,4) = 2*(1-
cdfnorm(abs(sobel(:,3)))).
end if.
do if (dichy = 0).
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
do if (ws= 1 and totlp=2).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute dfden=n-1.
end if.
compute cnms = {'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute op = {coeff, standerr, tratio, p}.
end if.
do if (dichy = 1).
compute tratio = (coeff&/standerr).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute wald = tratio*tratio.
compute cnms = {'coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute temp=coeff-abs(xp2)*standerr.
compute op = {coeff, standerr, tratio, p, temp}.
compute temp=coeff+abs(xp2)*standerr.
compute op = {op, temp}.
end if.
do if (detail = 1).
do if (totlp = 2).
print yname/title = '*****
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome: '/format = A8.
end if.
do if (totlp <> 2).

```

```

print yname/title =
*****
*****'/rlabels = 'Outcome:.'/format
= A8.
end if.
end if.
do if (dichy = 1 and bt = 1 and totlp = 1).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:.'/cnames = nmsd/format = F9.2.
end if.
do if (dichy = 0).
compute r2full = 1-(sse/ssty).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute jndf=dfden.
compute wsdf=dfden.
compute xd = abs(xp2).
compute jncrit = (dfden* (exp((dfden-
(5/6))*((xd/(dfden-
(2/3)+(.11/dfden))))*(xd/(dfden-
(2/3)+(.11/dfden)))))-1)).
compute summ = {sqrt(r2full), r2full, mse,
fratio, dfnum, dfden, pfr}.
compute temp1=coeff-sqrt(jncrit)*standerr.
compute temp2=coeff+sqrt(jncrit)*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (detail = 1).
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels =
'R', 'R-sq', 'MSE' 'F', 'df1', 'df2', 'p'/format =
F10.4.
end if.
end if.
end if.
do if (dichy = 1).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(coeff)-
1)).
compute mcF = LLdiff/LL3.

```

```

compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
do if (detail = 1).
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format =
F10.4.
end if.
end if.
do if (totlp = 2).
compute datanmy={constant';
datanmy((nmeds+2):nrow(datanmy),1)}.
end if.
do if (detail = 1).
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
do if (ws=1).
compute op2=op(1:(1+nmeds),:).
compute datanmy2 = {'c" path'; bpathnam(
(1+(nmeds>1)): ((1+(nmeds>1))+(nmeds-
1)),1)}.
do if (totlp=2).
compute datanmy2={'c path'}.
compute op2=op(1,:).
compute wsdf=n-1.
end if.
print op2/title = 'Model'/rnames =
datanmy2/cnames = cnms/format = F10.4.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
do if (ws <> 1).
print op/title = 'Model'/rnames =
datanmy/cnames = cnms/format = F10.4.
do if (covcoeff=1).
compute
covmattp=covmat(1:nrow(op),1:nrow(op)).
compute cnamestp=t(datanmy).

```



```

print covmattp/title='Covariance matrix of
regression parameter
estimates'/rnames=datanmy/cnames=cnamest
p/format= F10.4.
end if.
end if.
end if.
do if (ttt = 0 and totlp = 1).
compute deco = op((nmeds+2),:).
do if (ws = 1).
compute deco=op(1,:).
end if.
end if.
do if (ttt = 0 and totlp = 2).
compute decotot = op(2,:).
do if (ws = 1).
compute decotot=op(1,:).
end if.
end if.
do if (nmods > 0 and model > 4 and detail = 1
and nrow(yintkey) > 1)).
print yintkey/title = 'Product terms key:'/format
= A8.
end if.
do if (nmods > 0 and model < 4 and detail = 1).
print yintkey2/title = 'Product terms key:'/format
= A8.
do if ((model = 1 or model = 2) and dichy = 0).
compute temp={((op(4,3)**2)*(1-r2full))/dfden,
op(4,3)**2, 1, dfden, op(4,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(4,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(2,1).
do if (model = 2).
compute temp={temp;((op(6,3)**2)*(1-
r2full))/dfden, op(6,3)**2, 1, dfden, op(6,4)}.
do if (hc3=1).

```

```

compute temp(2,1)=((hcnot(6,1)**2)*(1-
r2full))/dfden.
end if.
compute frat2=(dfden*(r2full-(1-
(ssem2/ssty))))/(2*(1-r2full)).
do if (hc3=1).
compute lmat=make(nrow(coeffplt),2,0).
compute lmat(4,1)=1.
compute lmat(6,2)=1.
compute frat2 =
(t(t(lmat)*coeffplt)*inv(t(lmat)*covmat*lmat)*((t(l
mat)*coeffplt))/2).
end if.
compute temp={temp;(r2full-(1-
(ssem2/ssty))),frat2,2,dfden,1-
fcd(frat2,2,dfden)}.
compute rnms={rnms; yintkey2(3,1);'Both'}.
end if.
print temp/title = 'R-square increase due to
interaction(s):'/rnames=rnms/clabels = 'R2-
chng', 'F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (model = 3 and dichy = 0).
compute temp={((op(8,3)**2)*(1-r2full))/dfden,
op(8,3)**2, dfden, op(8,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(8,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(5,1).
print temp/title = 'R-square increase due to
three-way interaction:.'/rnames=rnms/clabels =
'R2-chng', 'F(1,df2)', 'df2', 'p'/format = F10.4.
end if.
end if.
end if.
do if (model = 6 and totlp = 1).
compute
mmpaths(nrow(mmpaths),1)=coeff(nrow(mmpa
ths),1).

```

```

compute
mmpaths(nrow(mmpaths),(2:(nmeds+1)))=t(co
eff(2:(nmeds+1),1)).
end if.
do if (totlp = 1).
loop im = 1 to nmeds.
do if (model < 4).
compute ymat(1,im) = coeff(3,1)*(1-yes).
compute ymat(2,im) = coeff(4,1).
compute cmat(1,im) = covmat(3,3).
compute cmat(2,im) = covmat(4,4).
compute cmat(5,im) = covmat(3,4).
compute jnb1=coeff(3,1).
compute jnb3=coeff(4,1).
compute jnsb1=covmat(3,3).
compute jnsb3=covmat(4,4).
compute jnsb1b3=covmat(3,4).
do if (model = 2 or model = 3).
compute ymat(3,im) = coeff(6,1).
compute cmat(3,im) = covmat(6,6).
compute cmat(6,im) = covmat(3,6).
compute cmat(8,im) = covmat(4,6).
end if.
do if (model = 3).
compute ymat(4,im) = coeff(8,1).
compute cmat(4,im) = covmat(8,8).
compute cmat(7,im) = covmat(3,8).
compute cmat(9,im) = covmat(4,8).
compute cmat(10,im) = covmat(6,8).
compute jnb1=coeff(4,1).
compute jnb3=coeff(8,1).
compute jnsb1=covmat(4,4).
compute jnsb3=covmat(8,8).
compute jnsb1b3=covmat(4,8).
end if.
end if.
do if (model > 3).
compute ymat(5,im) = coeff((1+im),1).
end if.
do if (xmy = 1).

```

```

compute ymat(6,im) = coeff((2+nmeds+im),1).
end if.
do if (vy = 1).
compute ymat(6,im) = coeff((3+nmeds+im),1).
end if.
do if (qy = 1 and vy = 1).
compute ymat(6,im) = coeff((5+nmeds+((im-1)*2)),1).
compute ymat(7,im) = coeff((6+nmeds+((im-1)*2)),1).
end if.
do if (vqy = 1).
compute ymat(6,im) = coeff((6+nmeds+((im-1)*3)),1).
compute ymat(7,im) = coeff((7+nmeds+((im-1)*3)),1).
compute ymat(8,im) = coeff((8+nmeds+((im-1)*3)),1).
end if.
do if (wmy = 1).
compute ymat(7,im) = coeff((3+nmeds+im-wy),1).
end if.
do if (wmy = 1 and vy = 1).
compute ymat(7,im) =coeff((4+(nmeds*2)+im-wy),1).
end if.
do if (wmy = 1 and vy = 1 and wvmy = 1).
compute ymat(7,im) =
coeff((6+(nmeds*2)+((im-1)*2)-wy),1).
compute ymat(8,im) =
coeff((7+(nmeds*2)+((im-1)*2)-wy),1).
end if.
do if (wmy = 1 and zmy = 1).
compute ymat(6,im) = coeff((6-(wzy*3)+(wzm-1)+(nmeds*2)+((im-1)*2)*wzm)+((im-1)*(1-wzm))-((zy-wzy)*2),1).
compute ymat(7,im) = coeff((3-zy+im+nmeds),1).
do if (wzmy = 1).

```

```

compute ymat(8,im) = coeff((7-
(wzy*3)+(nmeds*2)+((im-1)*2)),1).
end if.
end if.
do if (nmods > 0 and model <> 5).
loop indlp = 1 to nrow(modvals).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp)).
do if (model > 6).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp))*csum(ym
at(5:8,im)*vmat(5:8,indlp)).
end if.
end loop.
compute indboot((bt+(im-
1)*(boot+1)),:)=t(indeff).
do if (imm =1 or imm = 3).
compute
indbootp(bt,im)=ymat(2,im)*ymat(5,im).
do if (imm=3).
compute
indbootp(bt,(im+nmeds))=ymat(3,im)*ymat(5,i
m).
do if (zdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxz-cminz).
end if.
end if.
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
end if.
end if.
do if ((imm= 2 or imm = 4) or model = 74).
compute
indbootp(bt,im)=ymat(1,im)*ymat(6,im).
do if (imm=4).

```

```

compute
indbootp(bt,(im+nmeds))=ymat(1,im)*ymat(7,i
m).
do if (qdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxq-cminq).
end if.
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
end if.
end if.
do if (imm=7).
compute
indbootp(bt,im)=ymat(2,im)*ymat(6,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(2,im)*y
mat(6,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(6,im)*onesmv1)+(ymat(2,im)*y
mat(6,im)*t(immmv1)).
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).

```

```

compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1))))*(c
maxv-cminv).
end if.
end if.
do if (imm=5).
compute
indbootp(bt,im)=ymat(4,im)*ymat(5,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(4,im)*y
mat(5,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(3,im)*ymat(5,im)*onesmv1)+(ymat(4,im)*y
mat(5,im)*t(immmv1)).
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2))))*(c
maxw-cminw).
end if.
do if (zdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxz-
cminz).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1))))*(c
maxz-cminz).
end if.
end if.
do if (imm=6).

```

```

compute
indbootp(bt,im)=ymat(1,im)*ymat(8,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(1,im)*ymat(6,im)*onesmv2)+(ymat(1,im)*y
mat(8,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(7,im)*onesmv1)+(ymat(1,im)*y
mat(8,im)*t(immmv1)).
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxv-cminv).
end if.
do if (qdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxq-
cminq).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxq-cminq).
end if.
end if.
do if ((model = 58 or model = 59) and
wvdich=1).
compute
indbootp(bt,im)=(ymat(1,im)*ymat(7,im)*(cmax
w-cminw)+(ymat(2,im)*ymat(5,im)*(cmaxw-
cminw)+(ymat(2,im)*ymat(7,im)*((cmaxw*cma
xw)-(cminw*cminw))).
end if.
end if.

```



```

do if (model = 4 or model = 5).
compute
indboot(bt,im)=csum(yamat(1:4,im)*vmat(1:4,1
))*csum(yamat(5:8,im)*vmat(5:8,1)).
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.00000000000001.
end if.
compute sumind=rsum(indboot(bt,:)).
do if (im=nmeds).
compute
pmeff(bt,2:(im+1))=indboot(bt,:)/(sumind+coeff(
(2+nmeds),1)).
end if.
compute
rmeff(bt,(im+1))=indboot(bt,im)/coeff((2+nmeds
),1).
compute
abpseff(bt,(im+1))=indboot(bt,im)/stddevy.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*stddevx.
do if (ncovs > 0 or clsdmy > 0).
compute
abpseff(bt,(im+1))=indboot(bt,im)/sdy cov.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*sdx cov.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,1) = r2my-(r2full-r2xy).
compute abmmr = 1.
do if (indboot(bt,im) < 0).
compute abmmr = cmin(abmm).
end if.
do if (indboot(bt,im) > 0).
compute abmmr = cmax(abmm).
end if.
compute kappa2(bt,1)=indboot(bt,im)/abmmr.
compute tmp = indboot(bt,im)/abmmr.
end if.

```

```

end if.
end if.
do if (model = 6).
do if (nmeds = 2).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(4,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3).
compute
indboot(bt,3)=mmpaths(3,1)*mmpaths(4,3).
end if.
do if (nmeds = 3).
compute indboot(bt,1) =
mmpaths(2,1)*mmpaths(5,2).
compute indboot(bt,2) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3).
compute indboot(bt,3) =
mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4).
compute indboot(bt,4) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4).
compute indboot(bt,5) =
mmpaths(3,1)*mmpaths(5,3).
compute indboot(bt,6) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4).
compute indboot(bt,7) =
mmpaths(4,1)*mmpaths(5,4).
end if.
do if (nmeds = 4).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(6,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(6,3).
compute
indboot(bt,3)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(6,4).

```

```

compute
indboot(bt,4)=mmpaths(2,1)*mmpaths(5,2)*mmpaths(6,5).
compute
indboot(bt,5)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(6,4).
compute
indboot(bt,6)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3)*mmpaths(6,5).
compute
indboot(bt,7)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,8)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,9)=mmpaths(3,1)*mmpaths(6,3).
compute
indboot(bt,10)=mmpaths(3,1)*mmpaths(4,3)*mmpaths(6,4).
compute indboot(bt,11)
=mmpaths(3,1)*mmpaths(5,3)*mmpaths(6,5).
compute indboot(bt,12) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,13) =
mmpaths(4,1)*mmpaths(6,4).
compute indboot(bt,14) =
mmpaths(4,1)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,15) =
mmpaths(5,1)*mmpaths(6,5).
end if.
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.00000000000001.
end if.
compute
pmeff(bt,:)=indboot(bt,:)/(rsum(indboot(bt,:))+mmpaths(nrow(mmpaths),1)).

```

```

compute rmeff(bt,:) =
indboot(bt,:)/mmpaths(nrow(mmpaths),1).
compute abpseff(bt,:)=indboot(bt,:)/stddevy.
compute
abcseff(bt,:)=(stddevx*indboot(bt,:))/stddevy.
do if (ncovs > 0 or clsdmy > 0).
compute abpseff(bt,:)=indboot(bt,:)/sdycov.
compute
abcseff(bt,:)=(sdxcov*indboot(bt,:))/sdycov.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,:) = r2my-(r2full-r2xy).
end if.
end if.
end if.
end loop.
end if.
end loop.
do if (savboot = 1 and bt > 1).
compute bootstrp((bt-1),:)=bootcoef.
end if.
end loop.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
release datayed, dat, datamed, data, y, x,
datatm, dataty, mints2, cons, mnv, tmp.
do if (mc > 0).
compute x1 = sqrt(-
2*ln(uniform(mc,nrow(mcmats))))&*cos((2*3.14
159265358979)*uniform(mc,nrow(mcmats))).
compute x1=x1*chol(mcmats).
loop i=1 to nrow(x1).
compute x1(i,:)=x1(i,:)+t(mccoeff).
end loop.
loop i = 1 to nmeds.

```

```

compute
indboot(2:nrow(indboot),i)=x1(:,i)*x1(:,(i+nme
ds)).
end loop.
end if.
do if (ttt = 0 and model > 3).
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
end if.
do if (model <> 74).
do if (dichy = 0).
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 't', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 't', 'p', 'LLCI', 'ULCI'/format =
F10.4.
else.
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 'Z', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 'Z', 'p', 'LLCI', 'ULCI' /format =
F10.4.
end if.
end if.
end if.
do if (ttt > 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
compute clbs = {modvnm2, 'Effect', 'SE', 't', 'p',
'LLCI', 'ULCI'}.

```

```

compute tratio = (directv&/sedir).
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
compute outp = {modvalsd, directv, sedir,
tratio, p}.
do if (dichy = 0).
compute temp1=directv-sqrt(jncrit)*sedir.
compute temp2=directv+sqrt(jncrit)*sedir.
compute outp = {outp, temp1, temp2}.
end if.
do if (dichy = 1).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute temp = directv-abs(xp2)*sedir.
compute outp = {outp, temp}.
compute temp = directv+abs(xp2)*sedir.
compute outp = {outp, temp}.
compute clbs = {modvnm2, 'Effect', 'SE', 'Z', 'p',
'LLCI', 'ULCI'}.
end if.
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
print outp/title = 'Conditional direct effect(s) of
X on Y at values of the moderator(s):'/format =
F10.4 /cnames = clbs.
end if.
do if (nmods > 0 and model <> 5).
do if (model < 4).
print/title =
'*****'
'*****'.
compute zmat(1,1) = 1*(1-yes).
compute cfse = make(nrow(modvals),1,0).
loop #m = 1 to nrow(modvals).
do if (model = 1).
compute zmat(2,1)=modvals(#m,1)**2.
compute zmat(5,1)=2*modvals(#m,1).
end if.
do if (model = 2 or model = 3).
compute zmat(2,1)=modvals(#m,2)**2.
compute zmat(3,1)=modvals(#m,1)**2.

```

```

compute
zmat(4,1)=(modvals(#m,1)**2)*(modvals(#m,2)
**2)*(1-yes2).
compute zmat(5,1)=2*modvals(#m,2)*(1-yes).
compute zmat(6,1)=2*modvals(#m,1)*(1-yes).
compute
zmat(7,1)=2*modvals(#m,1)*modvals(#m,2)*(1-
yes).
compute
zmat(8,1)=2*modvals(#m,1)*modvals(#m,2).
compute
zmat(9,1)=2*modvals(#m,1)*(modvals(#m,2)**
2)*(1-yes2).
compute
zmat(10,1)=2*(modvals(#m,1)**2)*modvals(#m
,2)*(1-yes2).
end if.
compute cfse(#m,1)=sqrt(csum(zmat&*cmat)).
end loop.
end if.
do if (nmods > 0).
compute clbs = {modvnm, 'Effect'}.
loop im = 1 to nmeds.
compute obs = t(indboot(1+(im-1)*(boot+1),:)).
compute outp = {modvals, obs}.
do if (model < 4).
compute tstat = obs&/cfse.
do if (dichy = 0).
compute pval = 2*(1-tcdf(abs(tstat), (n-
ncol(xy)))).
compute temp=obs-sqrt(jncrit)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp=obs+sqrt(jncrit)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 't', 'p', 'LLCI', 'ULCI'}.
compute jnclbs=clbs.
end if.
do if (dichy = 1).
compute pval = 2*(1-cdfnorm(abs(tstat))).
compute temp = obs-abs(xp2)*cfse.

```

```

compute outp = {outp, cfse, tstat, pval, temp}.
compute temp = obs+abs(xp2)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute jnclbs=clbs.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indboot((1+(im-
1)*(boot+1)),:).
compute indboot2 = indboot((2+(im-
1)*(boot+1)):(1+(im-1)*(boot+1)+boot),:).
compute mnind = t(csum(indboot2)/boot).
compute stdind=t(sqrt((cssq(indboot2)-
((csum(indboot2)**2)/boot))/(boot-1))).
compute llci=make(1,ncol(indboot2),-999).
compute ulci=make(1,ncol(indboot2),-999).
loop #e = 1 to ncol(indboot2).

.
compute temp = indboot2(:,#e).
compute temp(GRADE( indboot2(:,#e) )) =
indboot2(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).

```



```

do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {modvals, obs, stdind, t(llci),
t(ulci)}.
compute clbs = {modvnm, 'Effect', 'Boot SE',
'BootLLCI', 'BootULCI'}.

```

```

end if.
compute mtemp = mnames(1, im).
compute rlbs =
make(nrow(modvals),1,mnames(1,im)).
do if (model < 4).
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
do if (yes=0).
print outp/title = 'Conditional effect of X on Y at
values of the moderator(s):'/cnames =
clbs/format = F10.4.
else.
compute outp=outp(:,3:ncol(outp)).
compute clbs=clbs(:,3:ncol(clbs)).
compute clbs(1,1)='Contrast'.
print outp/title = 'Contrast of conditional effects
of X on Y'/cnames = clbs/format = F10.4.
end if.
end if.
do if (model > 5 and mod74dic <> 1).
do if (im = 1).
print/title = 'Conditional indirect effect(s) of X on
Y at values of the moderator(s):'.
end if.
print outp/title = 'Mediator'/rnames =
rlbs/cnames = clbs/format = F10.4.
else if (mod74dic = 1 and model = 74).
do if (im = 1).
print/title = 'Indirect effect(s) of X on Y:'.
compute clbs3=clbs(1,2:ncol(clbs)).
end if.
compute outp3=outp(1,2:ncol(outp)).
print outp3/title = 'Mediator'/rnames =
rlbs/cnames = clbs3/format = F10.4.
end if.
end loop.
loop i = notes to 1 by -1.
do if (note(i,1) = 4 and yes=0).

```

```

print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 5 and yes=0).
print/title = 'Values for quantitative moderators
are the mean and plus/minus one SD from
mean.'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 14 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    below the mean was replaced
with the minimum because one SD below the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.
do if (note(i,1) = 15 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    above the mean was replaced
with the maximum because one SD above the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.
end loop.
do if (model = 3 and yes=0).
compute jnvals=make(nrow(matw),7,0).
compute jnvals(:,1)=matw.
compute jnvals(:,2)=jnb1+jnb3*jnvals(:,1).
compute
jnvals(:,3)=sqrt(jnsb1+2*jnvals(:,1)*jnsb1b3+(jn
vals(:,1)&*jnvals(:,1))*jnsb3).
compute jnvals(:,4)=jnvals(:,2)&/jnvals(:,3).

```

```

do if (dichy = 0).
compute jnvals(:,5)=2*(1-tcdf(abs(jnvals(:,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(:,5)=2*(1-
cdfnorm(abs(jnvals(:,4)))).
end if.
compute jnvals(:,6)=jnvals(:,2)-
sqrt(jncrit)*jnvals(:,3).
compute
jnvals(:,7)=jnvals(:,2)+sqrt(jncrit)*jnvals(:,3).
compute clbs={clbs(:,1),clbs(:,3:ncol(clbs))}.
do if (coeffci = 0).
compute jnvals =jnvals(:,1:(ncol(jnvals)-2)).
end if.
print jnvals/title = 'Conditional effect of X*M
interaction at values of W:'/cnames
=clbs/format = F10.4.
end if.
do if (jn = 1 and (model = 1 or model = 3) and
jndich = 0 and yes=0).
print/title = '***** JOHNSON-
NEYMAN TECHNIQUE
*****'.
compute ajn =(jncrit*jnsb3)-(jnb3*jnb3).
compute bjn = 2*((jncrit*jnsb1b3)-(jnb1*jnb3)).
compute cjn = (jncrit*jnsb1)-(jnb1*jnb1).
compute radarg = (bjn*bjn)-(4*ajn*cjn).
compute den = 2*ajn.
compute nrts = 0.
do if (radarg >= 0 and den <> 0).
compute x21 = (-bjn+sqrt(radarg))/den.
compute x22 = (-bjn-sqrt(radarg))/den.
compute roots = 0.
do if (x21 >= jnmin and x21 <= jnmax).
compute nrts = 1.
compute roots = {roots; x21}.
end if.
do if (x22 >= jnmin and x22 <= jnmax).

```

```

compute nrts = nrts + 1.
compute roots = {roots; x22}.
end if.
compute roots={roots,make(nrow(roots),2,0)}.
compute modtemp=m.
do if (model=3).
compute modtemp=w.
end if.
do if (nrts > 0).
compute roots = roots(2:nrow(roots),1:3).
compute roots(1,2)=(csum(modtemp <
roots(1,1))/n)*100.
compute roots(1,3)=(csum(modtemp >
roots(1,1))/n)*100.
do if (nrow(roots)=2).
compute roots(2,2)=(csum(modtemp <
roots(2,1))/n)*100.
compute roots(2,3)=(csum(modtemp >
roots(2,1))/n)*100.
end if.
print roots/title = 'Moderator value(s) defining
Johnson-Neyman significance
region(s)'/clabels = 'Value', '% below', '%
above'/format = F10.4.
compute jnvals=make((21+nrts),7,0).
loop i= 0 to 20.
compute jnvals((i+1),1)=jnmin+(i*((jnmax-
jnmin)/20)).
end loop.
loop i = 1 to nrts.
loop j = 2 to nrow(jnvals).
do if ((roots(i,1) > jnvals((j-1),1)) and (roots(i,1)
< jnvals(j,1))).
compute
jnvals((j+1):(21+i),1)=jnvals(j:(20+i),1).
compute jnvals(j,1)=roots(i,1).
end if.
end loop.
end loop.
loop i = 1 to nrow(jnvals).

```

```

compute jnvals(i,2)=jnb1+jnb3*jnvals(i,1).
compute
jnvals(i,3)=sqrt(jnsb1+2*jnvals(i,1)*jnsb1b3+(jn
vals(i,1)*jnvals(i,1))*jnsb3).
compute jnvals(i,4)=jnvals(i,2)/jnvals(i,3).
do if (dichy = 0).
compute jnvals(i,5)=2*(1-tcdf(abs(jnvals(i,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(i,5)=2*(1-
cdfnorm(abs(jnvals(i,4)))).
end if.
compute jnvals(i,6)=jnvals(i,2)-
sqrt(jncrit)*jnvals(i,3).
compute
jnvals(i,7)=jnvals(i,2)+sqrt(jncrit)*jnvals(i,3).
end loop.
do if (model = 1).
print jnvals/title = 'Conditional effect of X on Y
at values of the moderator (M)'/cnames
=jnclbs/format = F10.4.
end if.
do if (model = 3).
compute
jnclbs={jnclbs(:,1),jnclbs(:,3:ncol(jnclbs))}.
print jnvals/title = 'Conditional effect of X*M on
Y at values of the moderator (W)'/cnames
=jnclbs/format = F10.4.
end if.
end if.
do if (nrts = 0).
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
else.
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.

```

```

end if.
end if.
end if.
do if (model < 4 and (plot = 1)).
compute dataplot =
make((nrow(modvals)*nrow(matx)),(ncol(modv
als)+1),0).
compute tmp = 1.
loop i = 1 to nrow(modvals).
loop j = 1 to nrow(matx).
compute dataplot(tmp,:)= {matx(j,1),
modvals(i,:)}.
compute tmp=tmp+1.
end loop.
end loop.
compute dataplot = {dataplot,
make(nrow(dataplot),(1+dichy),0)}.
compute dataplo2 = make(nrow(dataplot),1,1).
do if (model = 1).
compute dataplo2 = {dataplo2, dataplot(:,2),
dataplot(:,1), (dataplot(:,1)*dataplot(:,2))}.
end if.
do if (model = 2 or model = 3).
compute dataplo2 = {dataplo2, dataplot(:,3),
dataplot(:,1), (dataplot(:,1)*dataplot(:,3)),
dataplot(:,2), (dataplot(:,1)*dataplot(:,2))}.
do if (model = 3).
compute dataplo2 = {dataplo2,
(dataplot(:,2)*dataplot(:,3)),
(dataplot(:,1)*dataplot(:,2)*dataplot(:,3))}.
end if.
end if.
loop i = 1 to nrow(dataplot).
compute tmp=dataplo2(i,:).
do if (ncovs > 0).
compute tmp = {tmp, covmeans}.
end if.
do if (cluster > 0).
compute tmp = {tmp, cldmeans}.
end if.

```

```

compute dataplot(i,(ncol(dataplot)-
(dichy)))=tmp*coeffplt.
do if (dichy = 1).
compute
dataplot(i,(ncol(dataplot)))=exp(tmp*coeffplt)/(1
+exp(tmp*coeffplt)).
end if.
end loop.
compute clbs = {xname, modvnm, 'yhat'}.
do if (dichy = 1).
compute clbs = {xname, modvnm, 'ln(odds)',
'prob'}.
end if.
print/title =
*****
*****!

```

```

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (model = 1).

```



```

do if (dichy = 0).
print/title = 'DATA LIST FREE/Opra_Pract
Eco_Innovation Financial_Performance.'
/space=1.
else if (dichy = 1).
print/title = 'DATA LIST FREE/Opra_Pract
Eco_Innovation logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
Financial_Performance BY Eco_Innovation.'
/space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Financial_Performance BY Opra_Pract.'
/space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
logodds BY Eco_Innovation.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
prob BY Eco_Innovation.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Opra_Pract.' /space=0.

```

```

print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Opra_Pract.' /space=0.
end if.
end if.
end if.
do if (model = 2 or model = 3).
do if (dichy = 0).
print/title = 'DATA LIST FREE/Opra_Pract
Eco_Innovation Financial_Performance.'
/space=1.
end if.
do if (dichy = 1).
print/title = 'DATA LIST FREE/Opra_Pract
Eco_Innovation logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
Financial_Performance BY
Eco_Innovation/PANEL ROWVAR=.' /space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Financial_Performance BY
Opra_Pract/PANEL ROWVAR=.' /space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
logodds BY Eco_Innovation/PANEL
ROWVAR=.' /space=0.

```

```

print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
prob BY Eco_Innovation/PANEL ROWVAR='
/space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Opra_Pract/PANEL
ROWVAR=' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Opra_Pract/PANEL
ROWVAR=' /space=0.
end if.
end if.
end if.
do if (ncovs > 0).
print/title = '* Estimates are based on setting
covariates to their sample means.'.
end if.
end if.
end if.
do if ((imm > 0) or (model = 74 and
mod74dic=0) or ((model = 58 or model = 59)
and wvdich=1))).
do if (imm > 4 and imm < 8).
compute
indbootp={indbootp(:,1:nmeds),indboop1,indbo
op2}.
end if.
compute obsprod = t(indbootp(1,:)).
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indbootp(1,:).
compute indbootp = indbootp(2:(boot+1),:).
compute mnindp = t(csum(indbootp)/boot).
compute stdindp=t(sqrt((cssq(indbootp)-
((csum(indbootp)**2)/boot))/(boot-1))).
compute llcip=make(1,ncol(indbootp),-999).

```

```

compute ulcip=make(1,ncol(indbootp),-999).
loop #e = 1 to ncol(indbootp).

.

compute temp = indbootp(:,#e).
compute temp(GRADE( indbootp(:,#e) )) =
indbootp(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).

```

```

end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcip(1,#e)=llcit.
compute ulcip(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {obsprod, stdindp, t(llcip),
t(ulcip)}.
compute clbs = {'Effect', 'SE(Boot)', 'BootLLCI',
'BootULCI'}.
do if (model = 8 or model = 12).
print/title='-----'.
compute outpimm=outp(1:nmeds,:).
print/title = 'Indirect effect of highest order
product:'/space=0.
print outpimm/title = 'Mediator'/cnames =
clbs/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).

```

```

print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'

end if.
compute clbs3 = {'Index', 'SE(Boot)',
'BootLLCI', 'BootULCI'}.
do if (imm < 5).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=outp((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'

compute outpimm=outp(1:nmeds,:).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
compute
outpimm=outp((nmeds+1):nrow(outp),:).
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).

```

```

compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),clbs3}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3(1,1)=modvnm(1,1).
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
end if.
do if (boot = 0).
do if (model = 8 or model = 12).
print/title='-----'.
compute obsimm=obsprod(1:nmeds,:).
print/title = 'Indirect effect of highest order
product:.'/space=0.
print obsimm/title = 'Mediator'/clabels =
'Effect'/rnames = mnames/format = F10.4.
end if.

```

```

do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
do if ((imm > 0 and imm < 5) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator: '/format=A8.
end if.
compute outpimm=obsprod((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
end loop.
end if.
do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'.
compute outpimm=obsprod(1:nmeds,1).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
compute
outpimm=obsprod((nmeds+1):nrow(obsprod),:)
.
compute outpimm={cmmval,outpimm}.

```



```

print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),'Index'}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3={modvnm(1,1),'Index'}.
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
end if.
end if.
end if.
compute conmake=0.
compute concols = 0.
do if ((model > 3 and model < 7) and (contrast
> 0) and nmods = 0 and nmeds > 1 ).
compute concols =
((ncol(indboot)*(ncol(indboot)-1))/2).
compute indcon=make(nrow(indboot),concols,-
999).
compute conkey = { ' ', ' ', ' ' }.
compute temp=1.
compute conmake=1.
loop i = 1 to (ncol(indboot)-1).

```

```

loop j = (i+1) to (ncol(indboot)).
compute indcon(:,temp)=indboot(:,i)-
indboot(:,j).
do if (contrast = 2).
compute indcon(:,temp)=abs(indboot(:,i))-
abs(indboot(:,j)).
end if.
do if (model <> 6).
compute conkey={conkey; mnames(1,i),
'minus', mnames(1,j)}.
end if.
do if (model = 6).
compute conkey={conkey; indlbl2(i,1), 'minus',
indlbl2(j,1)}.
end if.
compute temp=temp+1.
end loop.
end loop.
end if.
do if (model = 4 or model = 5).
compute clbs = {'Effect'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obs = {csum(obs); obs}.
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute rlbs = {rlbs;
cntname(1:ncol(indcon),1)}.
end if.
compute outp = obs.
compute outp2=outp.
do if (effsize = 1 and dichy = 0).
compute
pmeff(:,1)=rsum(pmeff(:,2:ncol(pmeff))).
compute
rmeff(:,1)=rsum(rmeff(:,2:ncol(rmeff))).
compute
abpseff(:,1)=rsum(abpseff(:,2:ncol(abpseff))).
compute
abcseff(:,1)=rsum(abcseff(:,2:ncol(abcseff))).

```

```

compute eff = {pmeff, rmeff, abpseff, abcseff}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute eff = {eff, r245}.
compute r245obs = {r245(1,1);r245(1,1)}.
compute kappa2ob = kappa2(1,1).
end if.
compute pmobs = t(pmeff(1,1:(nmeds+1))).
compute rmobs = t(rmeff(1,1:(nmeds+1))).
compute psobs = t(abpseff(1,1:(nmeds+1))).
compute csobs = t(abcseff(1,1:(nmeds+1))).
do if (contrast = 0).
compute outp2 = {obs, psobs, csobs, pmobs,
rmobs}.
end if.
do if (contrast > 0).
compute obs2=obs(1:nrow(psobs),:).
compute outp2 = {obs2, psobs, csobs, pmobs,
rmobs}.
end if.
compute clbs = {'ab', 'ab_ps', 'ab_cs', 'ab/c',
'ab/c'}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute outp2 = {outp2, r245obs}.
compute clbs = {clbs, 'R-sq_med'}.
do if (boot = 0 and mc = 0).
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
do if (boot = 0 and mc = 0).
do if (nmeds = 1).
compute outp2 = outp2(2,:).
compute rlbs = rlbs(2,1).
end if.
do if (model=5).
compute outp2=outp2(:,1:3).
end if.

```

```

print outp2/title = 'Indirect effect(s) of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1 and nmeds >
1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0 or mc > 0).
compute indboot = {rsum(indboot), indboot}.
compute bootSZ=boot.
do if (mc > 0).
compute bootSZ=mc.
end if.
compute ones = make(bootSZ,1,1).
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(bootSZ+1),:).
compute mnind = t(csum(indboot)/bootSZ).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/bootSZ))/(bootSZ-1))).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).
.
compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).

```

```

compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.

```

```

compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
do if (effsize = 1 and dichy = 0 and boot > 0).
compute estmte=eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).

.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2)))+1.

```

```

do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
end if.
end if.
do if (boot > 0 or mc > 0).
compute outp = {obs, stdind, t(llci), t(ulci)}.
do if (nmeds = 1).
compute outp = outp(2,:).
compute rlbs = rlbs(2,1).
end if.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.

```

```

do if (mc > 0).
compute clbs = {'Effect', 'MC SE', 'MC LLCI',
'MC ULCI'}.
end if.
print outp/title = 'Indirect effect of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (dichy = 0 and effsize = 1 and boot > 0).
compute outp = {psobs,
stdindf((2*(nmeds+1)+1):(3*(nmeds+1)),1),
t(llcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))),
t(ulcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Partially standardized indirect
effect of X on Y'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute outp = {csobs,
stdindf((3*(nmeds+1)+1):(4*(nmeds+1)),1),
t(llcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))),
t(ulcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Completely standardized
indirect effect of X on Y'/rnames = rlbs/cnames
= clbs/format = F10.4.
compute outp = {pmobs,
stdindf(1:(nmeds+1),1),t(llcif(1,1:(nmeds+1))),
t(ulcif(1,1:(nmeds+1))) }.
do if (model = 4).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
compute outp = {rmobs, stdindf(((nmeds+1)+1)
:(2*(nmeds+1)) ,1),

```



```

t(llcif(1,((nmeds+1)+1):(2*(nmeds+1))))),
t(ulcif(1,((nmeds+1)+1):(2*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
do if (nmeds = 1 and cluster = 0 and ncovs = 0
and model = 4).
compute r245obs = r245obs(1,1).
compute outp = {r245obs,
stdindf((4*(nmeds+1)+1):(4*(nmeds+1)+1),1),
t(llcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1))),
t(ulcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1)))}.
print outp/title = 'R-squared mediation effect
size (R-sq_med)'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
end if.
do if (normal = 1).
do if (nmeds = 1).
print sobel/title = 'Normal theory tests for
indirect effect'/clabels = 'Effect', 'se', 'Z',
'p'/format = F10.4.
end if.
do if (nmeds > 1).
compute rlbs2 = rlbs(2:nrow(rlbs),1).
print sobel/title = 'Normal theory tests for
specific indirect effects'/rnames = rlbs2/clabels
= 'Effect', 'se', 'Z', 'p'/format = F10.4.
end if.
end if.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).

```

```

print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
do if (model = 6).
compute clbs = {'eff'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obsfsum=csum(obs).
compute obs = {obsfsum; obs}.
compute indlbl = indlbl(1:nrow(obs),1).
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute indlbl = {indlbl;
cntname(1:ncol(indcon),1)}.
end if.
compute obs2=obs.
do if (boot = 0).
do if (dichy = 0 and effsize = 1).
compute clbs = {'eff', 'eff_ps', 'eff_cs', 'eff/c',
'eff/c'}.
do if (ncovs = 0).
compute obs = {obs, obs/stddevy,
(obs*stddevx/stddevy), obs/ctot,
obs/mmpaths(nrow(mmpaths),1)}.
end if.
do if (ncovs > 0 or clsdmy > 0).
compute obs = {obs, obs/sdycov,
(obs*sdxcov/sdycov),
obs/(obsfsum+mmpaths(nrow(mmpaths),1)),
obs/mmpaths(nrow(mmpaths),1)}.
end if.
compute obs2=obs.
do if (contrast > 0).
compute obs2 = obs(1:(nrow(obs)-concols),:).
end if.
end if.

```

```

print obs2/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute indboot = {rsum(indboot), indboot}.
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(boot+1),:).
compute mnind = t(csum(indboot)/boot).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/boot))/(boot-1))).
compute temp = nrow(indboot).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).
.
compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).

```

```

compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llici(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.

```

```

end if.
end loop.
compute obs = {obs, stdind, t(llci), t(ulci)}.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
print obs/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (effsize = 1 and dichy = 0).
compute indboot=indboot(:,1:(ncol(indboot)-
concols)).
compute eff = {rsum(abpseff), abpseff,
rsum(abcseff), abcseff, rsum(pmeff), pmeff,
rsum(rmeff), rmeff}.
compute effobs = eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).

.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(effobs(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).

```

```

do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( ( effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute temp2 = stdindf(1:ncol(indboot),1).
compute temp3 = effobs(:,1:ncol(indboot)).
compute templow = llcif(1,1:ncol(indboot)).
compute temphi = ulcif(1,1:ncol(indboot)).

```

```

compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Partially standardized indirect
effect of X on Y'/cnames = clbs/rnames =
indlbl/format = F10.4.
compute temp2 =
stdindf((ncol(indboot)+1):(2*ncol(indboot)),1).
compute temp3 =
effobs(:,(ncol(indboot)+1):(2*ncol(indboot))).
compute templow =
llcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute temphi =
ulcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Completely standardized
indirect effect of X on Y'/cnames = clbs/rnames
= indlbl/format = F10.4.
compute temp2 =
stdindf((2*(ncol(indboot))+1):(3*ncol(indboot)),1
).
compute temp3 =
effobs(:,(2*(ncol(indboot))+1):(3*ncol(indboot)))
.
compute templow =
llcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute temphi =
ulcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
compute temp =
eff(:,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temp2 =
stdindf((3*(ncol(indboot))+1):(4*ncol(indboot)),1
).

```

```

compute temp3 =
effobs(:,(3*(ncol(indboot))+1):(4*ncol(indboot)))
.
compute templow =
llcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temphi =
ulcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
end if.
end if.
do if (nmeds = 2).
compute effkey = {xname, '->', mnames(1,1), '->', yname, ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, ' ', ' '}.
compute effkey = {indlbl(2:4,1), effkey}.
end if.
do if (nmeds = 3).
compute effkey = {xname, '->', mnames(1,1), '->', yname, ' ', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname, ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,3), '->', yname, ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, ' ', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', mnames(1,3), '->', yname, ' ', ' '}.

```



```

compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, '', '', '', ''}.
compute effkey = {indl(2:8,1), effkey}.
end if.
do if (nmeds = 4).
compute effkey = {xname, '->' , mnames(1,1), '->', yname, '', '', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->', yname, '
', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->', yname, '
', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,4), '->', yname, '
', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname, '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,4), '->', yname, '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', mnames(1,4), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, '', '', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,4), '->', yname, '
', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, '', ''}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, '', '', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', mnames(1,4), '->', yname, '
', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,4), '->', yname, '', '', '', '', ''}.
compute effkey = {indlbl(2:16,1), effkey}.
end if.
print effkey/title = 'Indirect effect key'/format =
A8.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
end if.
DO IF (medte2=1).
compute
name=t(mdlnms2(2:nrow(mdlnms2),1)).
compute tmpdat=datayed(:,2:ncol(datayed)).
compute dd=tmpdat(:,1).
compute
dd={dd,tmpdat(:,(2+nmeds)),tmpdat(:,2:(1+nmeds
ds)))}.
compute stratify=0.
do if (ncovs > 0).
compute cname = cnames.
compute
dd={dd,tmpdat(:,(3+nmeds):(2+nmeds+ncovs))
}.
end if.
compute method = 1.
compute samples = boot.
do if (mc > 0).
compute samples = mc.
compute method=2.

```

```

end if.
compute omnibus = 1.
compute booterr = 0.
do if (mcx < 0 or mcx > 4).
compute mcx = 1.
end if.
compute ovals = ncol(design(dd(:,1))).
compute nte = notes.
compute criterr = 0.
compute xskip = 0.
compute dumok = 0.
compute badend = 0.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute priorlo = -9999999.
compute priorhi = 9999999.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute reps = samples.
do if (contrast = 1).
compute note(nte,1) = 26.
compute nte=nte+1.
end if.
do if (effsize = 1).
compute note(nte,1) = 27.
compute nte=nte+1.
end if.
do if (normal = 1).
compute note(nte,1) = 28.
compute nte=nte+1.
end if.
do if (method = 2).
do if (omnibus = 1).
compute note(nte,1) = 25.
compute nte=nte+1.
end if.
compute samples = 0.

```

```

do if (reps < 1000).
compute reps = 1000.
end if.
end if.
do if (method = 1).
compute samples = reps.
end if.
compute temp = ncol(dd).
compute n = nrow(dd).
compute ones = make(n,1,1).
compute temp=dd(:,2).
compute nxd = ncol(temp).
compute nx = nxd.
do if (nx = 1 and mcx > 0).
compute temp = dd.
compute temp(GRADE(dd(:,2)),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,2)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute mnvls = cmin(t(nnvls)).
do if (mnvls < 2).
compute errs = errs + 1.
compute runerrs(errs,1) = 37.
compute criterr = 1.
end if.
do if (nvls > 9).
compute errs = errs+1.
compute runerrs(errs,1) = 36.
compute criterr = 1.
end if.
do if (criterr = 0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,2).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,2) <> nnvls((temp-1),1)).
compute nnvls(temp,1)=dd(i,2).
compute temp = temp+1.

```

```

end if.
end loop.
do if (mcx > 0).
compute x = dummy(:,2:ncol(dummy)).
compute nx = ncol(x).
compute minus1 = make(1,ncol(x),-1).
compute xdes=make((nx+1),3,0).
compute xdes(1,1)=dd(1,2).
compute xdes(1,2)=1.
compute temp = 2.
loop k = 2 to n.
do if (dd(k,2) <> dd((k-1),2)).
compute xdes(temp,2) = k.
compute xdes(temp,1) = dd(k,2).
compute xdes((temp-1),3) = k-1.
compute temp=temp+1.
end if.
end loop.
compute xdes((temp-1),3)=n.
compute xdes = {xdes, (xdes(:,3)-xdes(:,2)+1)}.
do if (stratify <> 0).
compute note(nte,1) = 7.
compute nte=nte+1.
end if.
do if (mcx = 4).
loop k = 1 to n.
do if (rsum(x(k,:)) = 0).
compute x(k,:) = minus1.
end if.
end loop.
end if.
do if (mcx = 2 or mcx = 3).
compute note(nte,1)=4.
compute nte = nte+1.
loop k = 1 to n.
do if (rsum(x(k,:)) > 0).
loop i = 1 to ncol(x).
do if (x(k,i) = 0).
compute x(k,i) = 1.
else.

```

```

break.
end if.
end loop.
end if.
end loop.
do if (mcx = 3).
compute conmat1={-8,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1; 0,0,0,0,0,-
3,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.
loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute x(k,:)=conmat1((rsum(x(k,:))+1),:).
end loop.
end if.
end if.
compute xdata = x.
compute xskip = 1.
compute xname = ddd(1,1:nx).
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcx = 4).
compute dummat(1,:) = minus1.
end if.
do if (mcx = 2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcx = 3).
compute dummat=conmat1.
end if.

```

```

compute dummat={nnvls, dummat}.
compute xname2={name(1,2), xname}.
end if.
end if.
end if.
compute nm=nmeds.
compute r2b = make(nm,1,-999).
compute indeff = make((nx*nm),1,-999).
compute cov =
make((nx*nm+nm),(nx*nm+nm),0).
compute mns=make(nrow(cov),1,0).
compute ny = ncol(dd)-nm-nxd-ncovs.
compute y = dd(:,1).
compute ydata = y.
compute yname = name(1,1).
do if (xskip = 0).
compute x = dd(:,2:(1+nxd)).
compute xdata = x.
compute xname = name(1,2:(1+nxd)).
compute indlbs = t(xname).
end if.
compute m = dd(:,(2+nxd):(1+nxd+nm)).
compute mdata = m.
compute mlab = { ' M1 = ' ; ' M2 = ' ; ' M3 = ' ; '
M4 = ' ; ' M5 = ' ; ' M6 = ' ; ' M7 = ' ; ' M8 = ' ; '
M9 = ' ; ' M10 = ' }.
compute mlab = {mlab ; ' M11 = ' ; ' M12 = ' ; '
M13 = ' ; ' M14 = ' ; ' M15 = ' }.
compute mname =
name(1,(2+nxd):(1+nxd+nm)).
do if (ncovs > 0).
compute c =
dd(:,(2+nxd+nm):(1+nxd+nm+ncovs)).
compute x = {x, c}.
compute xdata = x.
compute cdata = c.
compute xname = {xname, cname}.
end if.
compute nms={yname; t(mname);
t(xname(1,1:(ncol(xname)-ncovs)))}.

```

```

do if (dumok = 1).
compute nms={yname; t(mname); name(1,2)}.
end if.
do if (nm < 16).
do if (dumok = 1).
print dummat/title = 'Coding of categorical X
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
compute
boots=make((samples+1),((nx+1)*nm),-999).
do if (errs = 0 and criterr = 0).
loop k = 1 to (samples+1).
do if (k = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef=0.
do if (k > 1).
compute u=trunc(uniform(n,1)*n)+1.
compute xdata = x(u,:).
compute mdata = m(u,:).
compute ydata = y(u,:).
do if (ncovs > 0).
compute cdata = c(u,:).
end if.
end if.
loop i = 1 to nm.
compute start = 1+(i-1)*nx.
compute xmat = {ones, xdata}.
do if (covmy=2).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = mdata(:,i).
compute amat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*amat.

```



```

compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute r2cha = r2.
compute adjr2cha = adjr2.
do if (ncovs > 0 and covmy <> 2).
compute xmat = {ones, cdata}.
compute atmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*atmat)**2).
compute r2step1 = (sstotal-ssresid2)/sstotal.
compute r2cha = r2-r2step1.
compute adjr2cha = adjr2- (1-((1-r2step1)*(n-
1)/(n-ncol(xmat)))).
compute xmat = {ones, xdata}.
end if.
compute bootcoef={bootcoef,t(amat)}.
compute r2b(i,1)=adjr2cha.
do if (k = 1).
.
compute hcinvtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.

```

```

end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute tratio = amat&/sqrt(diag(covmat)).
compute rnms = mname(1,i).
print rnms/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.
compute lmat = ident(nrow(amat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*amat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*amat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/'format = F10.4
/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'.
compute op = {amat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/'clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/'rnames = rnms/'format =
F10.4.
compute cov(start:(start+(nx-
1)),start:(start+(nx-

```

```

1)))=covmat(2:(ncol(covmat)-(ncovs*(covmy <>
2))),2:(ncol(covmat)-(ncovs*(covmy <> 2)))).
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
=F10.4.
end if.
end if.
compute mns(start:(start+(nx-
1)),1)=amat(2:(1+nx)).
end loop.
compute xmat = {ones,mdata,xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (k = 1).
print yname/title =
*****
*****'/rlabels = 'Outcome:'/format
=A8.
do if (ovals = 2).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:'/cnames = nmsd/format = F9.2.
end if.
end if.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(ymat)/n)).
compute LL3 = ymat*&ln(pt2)+(1-ymat)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.
.
compute pt1lp= pt1.
compute xlp= xmat.

```

```

compute ylp= ymat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp&*ln(pt1lp)+(1-ylp)&*ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.

```

```

compute bootcoef={bootcoef,t(bmat)}.
compute LL1 = LL2.
compute covmat=varb.
end if.
do if (ovals <> 2).
compute bmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*bmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute bootcoef={bootcoef,t(bmat)}.
end if.
do if (k = 1).
do if (ovals = 2).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).
compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)*sqrt(diag(covmat)).
compute outp = {outp, temp}.

```

```

compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms2/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute
opdirsum=outp((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
end if.
do if (ovals <> 2).

.

compute hcinvtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(bmat)).
compute lmat = lmat(:,2:ncol(lmat)).

```

```

compute fratio =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/(ncol(xmat)-1).
compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat((2+nmeds):(2+nmeds+(nx-1)),:)
= lmat2.
compute fratiodr =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/nx.
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute df2=(n-ncol(xmat)).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), df2, pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = bmat&/sqrt(diag(covmat)).
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute op = {bmat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat)))) }.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*(xp2/(temp-
(2/3)+(.11/temp)))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms2/format =
F10.4.
compute
opdirsum=op((2+nmeds):(1+nmeds+nx),:).

```

```

compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
compute xmat = {ones,mdata}.
do if (ncovs > 0).
compute xmat = {xmat, cdata}.
end if.
compute btmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*btmat)**2).
compute r2direct = r2-((sstotal-
ssresid2)/sstotal).
compute pfr = 1-fcdf(fratiodr,(nx),(n-1-nx-nm-
ncovs)).
compute r2direct = {r2direct, fratiodr, nx, (n-1-
nx-nm-ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms2/cnames=rnms2/form
at= F10.4.
end if.
end if.
do if (k = 1 and toteff = 1).
print yname/title = '*****
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome: '/format = A8.
compute xmat = {ones, xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(ymat)/n)).
compute LL3 = ymat*ln(pt2)+(1-ymat)*ln(1-
pt2).
compute LL3 = -2*csum(LL3).

```



```

compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.

compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= ymat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)*&ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.

```

```

end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute LL1 = LL2.
compute covmat=varb.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(bmat)-
1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms = {'Constant'; t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).
compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute optotsum=outp(2:(1+nx),:).
end if.
do if (ovals <> 2).

```

```

compute cmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*cmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).

.

compute hcinvtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvtX*t( xmat )* xmat
*hcinvtX
.
compute lmat = ident(nrow(cmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).

```

```

compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = cmat&/sqrt(diag(covmat)).
compute op = {cmat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*(xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute optotsum=op(2:(1+nx),:).
compute ssresid2 = ssresid.
compute r2total = r2.
compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat(2:(1+nx),:) = lmat2.
compute fratio3 =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/nx.
do if (ncovs > 0).
compute xmat = {ones, cdata}.
compute ctmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*ctmat)**2).
compute r2total = r2-((sstotal-ssresid2)/sstotal).
end if.

```

```

compute pfr = 1-fcdf(fratio3,(nx),(n-1-nx-
ncovs)).
compute r2tot2 = {r2total, fratio3, nx, (n-1-nx-
ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns((nrow(mns)-
nm+1):nrow(mns),1)=bmat(2:(1+nm),1).
compute tmp = 1.
loop i = 1 to (nm*nx).
compute
boots(k,i)=mns(i,1)*mns(((nm*nx)+tmp),1).
do if ((i/nx)=trunc(i/nx)).
compute
boots(k,((nm*nx)+tmp))=r2b(tmp,1)*mns(((nm*
nx)+tmp),1).
compute tmp = tmp+1.
end if.
end loop.
do if (savboot = 1 and k > 1).
compute bootstrp((k-1),:)=bootcoef.
end if.
end loop.
do if (method = 1).
compute llci=make(1,ncol(boots),-999).
compute ulci=make(1,ncol(boots),-999).
compute indeff = t(boots(1,1:(nm*nx))).
compute indeff22=t(boots(1,:)).
compute omni = t(boots(1,((nm*nx)+1) :
ncol(boots))).
do if (nrow(boots) > 1).
compute boots = boots(2:nrow(boots),:).
compute bootse = t(sqrt((((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).

```

```

loop i = 1 to ncol(boots).

.

compute temp = boots(:,i).
compute temp(GRADE( boots(:,i) )) = boots(:,i).
compute badlo = 0.
compute badhi = 0.
do if ( (indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(indeff22(i,1)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.

```

```

do if ( (indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,i)=llcit.
compute ulci(1,i)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute llci = t(llci).
compute ulci= t(ulci).
compute llci={llci,ulci}.
end if.
end if.
do if (method = 2).
compute tmp=1.
loop i = 1 to (nm*nx).
compute
indeff(i,1)=mns(i,1)*mns(((nm*nx)+tmp),1).
do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = sqrt(-
2*ln(uniform(reps,nrow(cov))))&*cos((2*3.1415
9265358979)*uniform(reps,nrow(cov))).
compute boots=boots*chol(cov).
loop i = 1 to ncol(boots).
compute boots(:,i)=boots(:,i)+mns(i,1).
end loop.
compute tmp=1.
loop i = 1 to (nm*nx).

```

```

compute
boots(:,i)=boots(:,i)*boots(:,((nm*nx)+tmp)).
compute temp = boots(:,i).
compute temp(GRADE(boots(:,i))) = boots(:,i).
compute boots(:,i) = temp.
do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = boots(:,1:(nm*nx)).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
compute llci =
t(((boots(cilow,:));(boots(cihigh,:)))).
end if.
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
compute rnmssum=rnms(2:(1+nx),1).
do if (ovals <> 2).
print optotsum/title = 'Relative total effects of X
of Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.
print r2tot2/title = 'Omnibus test of total effect of
X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print optotsum/title = 'Relative total effects of X
on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
end if.
compute
rnmssum=rnms2((2+nmeds):(1+nmeds+nx),1).
do if (ovals <> 2).

```



```

print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.
print r2direct/title = 'Omnibus test of direct
effect of X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/'space=0.
do if (omnibus = 1 and (method = 1 or method
= 3)).
compute indlbs = {indlbs; 'Omnibus'}.
end if.
loop i = 1 to nm.
compute op = mname(1,i).
print op/title = 'Relative indirect effect(s) of X on
Y through:.'/format = A8/space=0.
do if (method = 2).
compute clbs = {'Effect', 'SE MC', 'MC LLCI',
'MC ULCI'}.
compute op = {indeff, bootse(1:(nm*nx)), llci}.
end if.
do if (method = 1).
compute clbs = {'Effect', 'SE(boot)', 'LLCI',
'ULCI'}.
compute op = indeff.
do if (samples > 0).
compute op = {indeff, bootse(1:(nm*nx),:),
llci(1:(nm*nx),:)};.
end if.
end if.
compute temp = op((((i-1)*nx)+1):(nx*i),:).
do if ((method = 1) and omnibus = 1).
do if (samples = 0).
compute temp={temp;omni(i,1)}.
end if.

```

```

do if (samples > 0).
compute temp = {temp; omni(i,1),
bootse(((nm*nx)+i,:), llci(((nm*nx)+i,:),:))}.
end if.
end if.
print temp/title = ' '/rnames = indlbs/cnames =
clbs/format = F10.4 /space=0.
print/title = '-----'.
end loop.
end if.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
END IF.
DO IF (mdpbe=1).
compute nm={yname,xname,mnames}.
do if (ncovs > 0).
compute
nm={yname,cnames,xname,mnames}.
end if.
compute tpx=xname.
compute x = dd(:,2:ncol(dd)).
compute tpy = yname.
compute tempy= dd(:,1).
compute n = nrow(dd).
compute nx=1.
compute criterr=0.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute ddd1 = {'int_1', 'int_2', 'int_3', 'int_4',
'int_5', 'int_6', 'int_7', 'int_8', 'int_9'}.

```

```

compute dumok = 0.
compute mcfoc=mcx.
compute mcmmod=mcm.
compute mcloc=(mcfoc>0).
compute con = make(n,1,1).
compute ncovs=ncol(x)-2.
do if (mcfoc > 0 or mcmmod > 0).
compute temp = dd.
compute temp(GRADE(dd(:,(ncol(dd)-
mcloc))),:)= dd.
compute dd = temp.
compute dummy = design(dd(:,ncol(dd)-
mcloc)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute toosmall=rsum(nnvls < 2).
compute mnvls = cmin(t(nnvls)).
do if ((rsum(nnvls < 2)) > 0).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=37.
end if.
do if (nvls > 10).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=36.
end if.
do if (criterr=0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,ncol(dd)-mcloc).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,ncol(dd)-mcloc) <> nnvls((temp-
1),1)).
compute nnvls(temp,1)=dd(i,ncol(dd)-mcloc).
compute temp = temp+1.
end if.
end loop.
compute dummy = dummy(:,2:ncol(dummy)).

```

```

do if (mcfoc=4 or mcmmod=4).
compute minus1=make(1,ncol(dummy),-1).
loop k = 1 to n.
do if (rsum(dummy(k,:)) = 0).
compute dummy(k,:)=minus1.
end if.
end loop.
end if.
do if (mcfoc = 2 or mcfoc = 3) or (mcmmod = 2 or
mcmmod = 3)).
loop k = 1 to n.
do if (rsum(dummy(k,:)) > 0).
loop i = 1 to ncol(dummy).
do if (dummy(k,i) = 0).
compute dummy(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcfoc = 3 or mcmmod=3).
compute conmat1={-8,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1; 0,0,0,0,0,-
3,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.
loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute
dummy(k,:)=conmat1((rsum(dummy(k,:))+1),:).
end loop.
end if.
end if.
compute nx = ncol(dummy).
compute xname = ddd(1,1:nx).
compute xdata = dummy.

```

```

compute xname = ddd(1,1:nx).
compute xname2={nm(1,(ncol(nm)-mcloc)),
xname}.
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcfoc = 2 or mcmmod=2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcfoc = 3 or mcmmod=3).
compute dummat=conmat1.
end if.
do if (mcfoc = 4 or mcmmod = 4).
compute dummat(1,:)=minus1.
end if.
compute dummat={nnvls, dummat}.
end if.
end if.
do if (criterr=0).
compute y = dd(:,1).
compute ovals = ncol(design(dd(:,1))).
compute itprob = 0.
do if (ovals = 2).
compute omx = cmax(y(:,1)).
compute omn = cmin(y(:,1)).
compute y(:,1) = (y(:,1) = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute sstotal = csum((y-(csum(y)/n))&**2).
compute outv = t(nm(1,1)).
compute mdtr = nm(1,ncol(dd)).
compute fciv = nm(1,(ncol(dd)-1)).
do if (mcfoc <> 0 or mcmmod <> 0).
compute inter=make(n,nx,0).
loop i = 1 to nx.

```

```

compute inter(:,i)=dummy(:,i)*dd(:,ncol(dd)-
(1-mcloc)).
end loop.
do if (ncovs = 0).
compute x={con,dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y)))}.
end if.
do if (ncovs > 0).
compute
x={con,dd(:,2:(1+ncovs)),dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={t(nm(1,2:(1+ncovs)));
nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y)))}.
end if.
end if.
compute xmns = csum(x)/n.
compute focvals = {1,0,0}.
compute highwarn = 0.
compute lowwarn = 0.
do if (ncol(x) > (2*nx+2)).
compute covmns = {1, xmns(1,2:(ncol(x)-1-
(2*nx)))}.
compute focvals = {covmns,0,0}.
end if.
compute dfres = n-ncol(x).
do if (ovals = 2).
compute pt2 = make(n,1,(csum(y)/n)).
compute LL3 = y*&ln(pt2)+(1-y)*&ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(x),1,0).
compute LL1 = 0.

```

```

.
compute pt1lp= pt1.
compute xlp= x.
compute ylp= y.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)*&ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.

```

```

end if.
compute tval=xp2*xp2.
else if (ovals <> 2).
compute invXtX=inv(t(x)*x).
compute b = invXtX*t(x)*y.
compute resid = y-(x*b).
compute ssresid=csum(resid&*resid).
compute msresid=ssresid/dfres.
compute varb = msresid*invXtX.
compute
seint=sqrt(varb(nrow(varb),nrow(varb))).
compute k3 = nrow(b).
do if (hc3 = 1).
compute xhc=x.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)= xhc(i3,:)*invXtX*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).
end loop.
compute varb=(invXtX*t(xhc)*xhc*invXtX).
end if.
compute seb = sqrt(diag(varb)).
compute r2 = 1-(ssresid/sstotal).
do if (mcfoc <> 0 or mcmmod <> 0).
compute ytp=y.
compute xtp=x(:,1:(ncol(x)-nx)).
compute btp = inv(t(xtp)*xtp)*t(xtp)*ytp.
compute residt = ytp-(xtp*btp).
compute ssresidt=csum(residt&*residt).
compute r2noint = 1-(ssresidt/sstotal).
end if.
compute pr = ncol(x)-1.
compute lmat = ident(nrow(b)).
compute lmat = lmat(:,2:ncol(lmat)).
compute f =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
pr).

```



```

compute pf = 1-fcdf(f,pr,dfres).
compute pf = {sqrt(r2),r2,msresid,f,pr,dfres,pf}.
compute xd=abs(xp2).
end if.
do if (itprob = 0).
compute tstat = b&/seb.
do if (ovals <> 2).
compute p = 2*(1-tcdf(abs(tstat), dfres)).
else if (ovals = 2).
compute p = 2*(1-cdfnorm(abs(tstat))).
end if.
compute outp = {b,seb,tstat,p}.
compute nms = {'constant'; nms; 'interact'}.
do if (ovals <> 2).
compute tval = (dfres* (exp((dfres-
(5/6))*((xd/(dfres-(2/3)+(.11/dfres)))*(xd/(dfres-
(2/3)+(.11/dfres)))))-1)).
compute outp={outp,(b-
sqrt(tval)&*seb),(b+sqrt(tval)&*seb)}.
end if.
compute bb = tval.
do if (dumok = 1).
do if (mcfoc > 0).
print dummat/title = 'Coding of categorical X
variable for analysis:.'/cnames = xname2/format
= F5.2.
else if (mcmod > 0).
print dummat/title = 'Coding of categorical M
variable for analysis:.'/cnames = xname2/format
= F5.2.
end if.
end if.
print yname/title =
*****
*****'/rlabels = 'Outcome:.'/format
= A8.
do if (ovals = 2).
compute nmsd = {outv, 'Analysis'}.
print rcd/title = 'Coding of binary Y for
analysis:.'/cnames = nmsd/format = F10.4.

```

```

compute outp = {outp,(outp(:,1)-
sqrt(tval)&*outp(:,2)),(outp(:,1)+sqrt(tval)&*outp
(:,2))}.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
print outp/title = 'Logistic Regression
Model'/rnames = nms/clabels 'Coeff' 'se' 'Z' 'p'
'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
compute varbtmp=varb.
compute btmp=b.
compute LL2f=LL2.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=x(:,1:(ncol(x)-nx)).
compute bt1 = make(ncol(xlp),1,0).

.

compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).

```

```

compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&*ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL2f.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(LLdiff,nx).
compute rcha={LLdiff,nx,pchi}.
end if.
do if (ovals <> 2).
print pf/title = 'Model Summary'/clabels = 'R' 'R-
sq' 'MSE' 'F' 'df1' 'df2' 'p'/format F10.4.
compute lmat=make(nrow(b),1,0).

```

```

compute lmat(nrow(lmat),1)=1.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
1).
compute
rcha=((b(nrow(b),1)/seint)*(b(nrow(b),1)/seint))*
(1-r2)/dfres.
compute rcha = {rcha, fcha, 1, dfres,
outp((pr+1),4)}.
print outp/title = 'Model'/rnames = nms/clabels
'coeff' 'se' 't' 'p' 'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute rcha=r2-r2noint.
compute lmat=make((nrow(b)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
nx).
compute pvalr2c=1-fcdf(fcha,nx,dfres).
compute rcha = {rcha, fcha, nx, dfres,pvalr2c}.
end if.
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute intkey = {'a', 'b', 'c', 'd', 'e'}.
loop i = 1 to nx.
compute intkey={intkey; ddd1(1,i), ' : ', ddd1(1,i),
' X ', nm(1,ncol(dd)-(1-mcloc))}.
end loop.
compute intkey=intkey(2:nrow(intkey),:).
print intkey/title='Product terms key:.'/format =
a8.
end if.
do if (ovals <> 2).

```

```

print rcha/title = 'R-square increase due to
interaction:.'/clabels 'R2-chng' 'F' 'df1' 'df2'
'p'/format F10.4.
else if (ovals = 2).
print rcha/title = 'Likelihood ratio test for
interaction:.'/clabels 'Chi-sq' 'df' 'p'/format F10.4.
end if.
print/title =
*****
*****!
compute mdvar = x(:,(ncol(x)-1)).
do if (mcfoc = 0 and mcmod = 0).
compute g1 = b((ncol(x)-2),1).
compute g3 = b(ncol(x),1).
compute vg1 = varb((ncol(x)-2),(ncol(x)-2)).
compute vg3 = varb(ncol(x),ncol(x)).
compute covg1g3 = varb((ncol(x)-2), ncol(x)).
end if.
do if (mcfoc > 0 or mcmod > 0).
compute mdvar = x(:,(ncol(x)-(2*nx))).
end if.
compute mdmin = cmin(mdvar).
compute mdmax = cmax(mdvar).
compute fvar = x(:,(ncol(x)-2)).
compute nval = ncol(design(mdvar)).
compute fvmin = cmin(fvar).
compute fvmax = cmax(fvar).
do if (mmodval = 999).
compute mnmd = csum(mdvar)/n.
compute tmp = make(n,1,mnmd).
compute sdmd = sqrt(csum(((mdvar-
tmp)**2))/(n-1)).
compute probeval = {mnmd-sdmd; mnmd;
mnmd+sdmd}.
do if (probeval(1,1) < mdmin).
compute lowwarn = 1.
compute probeval(1,1)=mdmin.
end if.
do if (probeval(nrow(probeval),1) > mdmax).
compute highwarn = 1.

```

```

compute probeval(nrow(probeval),1)=mdmax.
end if.
do if (quantile = 1).
compute tmp = mdvar.
compute tmp(GRADE(mdvar(:,1)),:)= mdvar.
compute mdvar = tmp.
compute
probeval={mdvar(trunc(0.1*n),1);mdvar(trunc(0
.25*n),1);mdvar(trunc(0.5*n),1);mdvar(trunc(0.7
5*n),1);mdvar(trunc(0.9*n),1)}.
end if.
end if.
do if (nval = 2).
compute probeval = make(2,1,0).
compute probeval(1,1)=cmin(mdvar).
loop i = 1 to n.
do if (mdvar(i,1) <> probeval(1,1)).
compute probeval(2,1) = mdvar(i,1).
BREAK.
end if.
end loop.
end if.
do if (mmodval <> 999).
compute probeval = mmodval.
end if.
compute outp = make(nrow(probeval),7,0).
do if (mcfoc > 0 or mcmmod > 0).
compute focvals=make(1,ncol(x)+1,1).
do if (mcfoc > 0).
print/title = 'Conditional effect of X on Y at
values of the moderator:'.
print/title = ' '/space=0.
compute rnn2=mdtr.
compute matt=make(nx,6,0).
end if.
loop jj=1 to nrow(probeval).
do if (mcfoc > 0).
loop ii=1 to nx.
compute g1=b((ncol(x)-(2*nx)+ii),1).
compute g3=b((ncol(x)-nx+ii),1).

```

```

compute vg1=varb((ncol(x)-(2*nx)+ii),(ncol(x)-
(2*nx)+ii)).
compute vg3=varb((ncol(x)-nx+ii),(ncol(x)-
nx+ii)).
compute covg1g3=varb((ncol(x)-
(2*nx)+ii),(ncol(x)-nx+ii)).
compute x2 = probeval(jj,1).
compute w1 = g1+g3*x2.
compute varw1 =
vg1+(2*x2*covg1g3)+((x2*x2)*vg3).
compute sew1 = sqrt(varw1).
compute t1 = w1/sew1.
do if (ovals <> 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-tcdf(abs(t1), dfres)).
compute cnms = {'Coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)={w1,sew1,t1,p,llci,ulci}.
else if (ovals = 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-cdfnorm(abs(t1))).
compute cnms = {'Coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)={w1,sew1,t1,p,llci,ulci}.
end if.
end loop.
compute rnms=t(xname).
compute mdvalpr=probeval(jj,1).
print mdvalpr/title = 'Moderator
value:.'/rnames=rnn2/format= F10.4 /space=0.
print matt/title='
'/cnames=cnms/rnames=rnms/format= F10.4
/space=0.
compute xprob=x(:,(ncol(x)-(2*nx)))-mdvalpr.
loop kk = 1 to nx.
compute xprob={xprob,
(xprob(:,1)&*x(:,(ncol(x)-(2*nx)+kk)))}.
end loop.

```

```

compute xprob={x(:,1:((ncol(x)-(2*nx)-
1)),xprob}.
do if (ovals <> 2).
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute residc = y-(xprob*bmultc).
compute ssresidc=csum(residc*residc).
compute r2c = r2-(1-(ssresidc/sstotal)).
compute fcha2=(dfres*r2c)/(nx*(1-r2)).
do if (hc3 = 1).
loop kk = 1 to nx.
compute xprob={xprob, x(:,(ncol(x)-
(2*nx)+kk))}.
end loop.
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute k3 = nrow(bmultc).
compute xhc=xprob.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)=
xhc(i3,)*inv(t(xprob)*xprob)*t(xhc(i3,)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).
end loop.
compute
varbc=(inv(t(xprob)*xprob)*t(xhc)*xhc*inv(t(xpr
ob)*xprob)).
compute lmat=make((nrow(bmultc)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha2 =
(t(t(lmat)*bmultc)*inv(t(lmat)*varbc*lmat)*((t(lm
at)*bmultc)))/nx).
end if.
compute pvalr2cc=1-fcdf(fcha2,nx,dfres).
compute rcha2 = {r2c, fcha2, nx,
dfres,pvalr2cc}.

```



```

print rcha2/title = 'Test of equality of conditional
means at this value of the moderator'/clabels
'R2-chng' 'F' 'df1' 'df2' 'p'/format F10.4.
else if (ovals = 2).
compute btmp=b.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=xprob.
compute bt1 = make(ncol(xlp),1,0).

.

compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*ln(pt1lp)+(1-ylp)*ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).

```

```

compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL4.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(LLdiff,nx).
compute rcha2 = {LLdiff, nx,pchi}.
print rcha2/title = 'Test of equality of log odds
conditioned on this moderator value'/clabels
'Chi-sq' 'df' 'p'/format F10.4.
end if.
end if.
compute
tttt=make(nrow(dummat),1,probeval(jj,1)).
compute tttt={tttt,dummat(:,2:ncol(dummat))}.
loop kkk=1 to nx.
compute tttt={tttt,tttt(:,1)*tttt(:,1+kkk)}.
end loop.
compute ones=make(nrow(dummat),1,1).
do if (ncol(x) > (2*nx+2)).
compute
covmnmat=make(nrow(tttt),ncol(covmns),0).
loop kkk=1 to nrow(tttt).
compute covmnmat(kkk,:)=covmns.
end loop.
compute tttt={covmnmat,tttt}.
else.
compute tttt={ones,tttt}.
end if.
compute focvals={focvals;dummat(:,1),tttt}.
do if (mcfoc > 0).
compute yhat={dummat(:,1),(tttt*b)}.
compute cmnms={fciv, 'yhat'}.
do if (ovals <> 2).

```

```

print yhat/title = 'Estimated conditional means
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (ovals = 2).
print yhat/title = 'Estimated conditional log odds
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (jj <> nrow(probeval)).
print/title='-----'/space=0.
end if.
end if.
end loop.
compute focvals=focvals(2:nrow(focvals),:).
compute yhat=focvals(:,2:ncol(focvals))*b.
compute
focvals={focvals(:,1),focvals(:,(ncol(focvals)-
(2*nx))),yhat}.
compute cnms={fciv,mdtr,'yhat'}.
do if (mcmod > 0).
compute cnms={mdtr,fciv,'yhat'}.
end if.
end if.
do if (mcmod > 0).
compute outp=make((nx+1),7,0).
compute bcatm={b((ncol(x)-
(2*nx)),1);b((ncol(x)-nx+1):ncol(x),1)}.
compute outp(:,1)=dummat(:,1).
compute bcatcov=varb((ncol(x)-
nx):ncol(x),(ncol(x)-nx):ncol(x)).
compute bcatcov(1,1)=varb((ncol(x)-
(2*nx)),(ncol(x)-(2*nx))).
compute
bcatcov(2:nrow(bcatcov),1)=varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx)).
compute
bcatcov(1,2:nrow(bcatcov))=t(varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx))).
loop i = 1 to nrow(dummat).

```

```

compute
catmval={1,dummat(i,2:ncol(dummat))}.
compute condeff=catmval*bcatm.
compute
condse=sqrt(catmval*bcatcov*t(catmval)).
compute outp(i,2:3)={condeff,condse}.
end loop.
compute outp(:,4)=outp(:,2)&/outp(:,3).
compute outp(:,5) = 2*(1-tcdf(abs(outp(:,4)),
dfres)).
compute outp(:,6) = (outp(:,2)-
sqrt(tval)&*outp(:,3)).
compute outp(:,7) =
(outp(:,2)+sqrt(tval)&*outp(:,3)).
do if (ovals <> 2).
compute cnmms = {xname2(1,1), 'coeff', 'se',
't', 'p', 'LLCI', 'ULCI'}.
end if.
do if (ovals = 2).
compute outp(:,5) = 2*(1-
cdfnorm(abs(outp(:,4)))).
compute cnmms = {xname2(1,1), 'coeff', 'se',
'Z', 'p', 'LLCI', 'ULCI'}.
end if.
print outp/title = 'Conditional Effect of Focal
Predictor in Groups Defined by the Moderator
Variable: '/cnames=cnmms/format = F10.4.
end if.
do if (nval > 2 and (mmodval = 999) and
mcmmod = 0).
do if (quantile <> 1).
print/title = 'Moderator values are the sample
mean and plus/minus one SD from mean'.
else.
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
end if.
do if (highwarn = 1 and quantile = 0).

```

```

print/title = 'NOTE: For the moderator values
above, one SD above the mean was'.
print/title = 'replaced with the maximum
because one SD above the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
do if (lowwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD below the mean was'.
print/title = 'replaced with the minimum
because one SD below the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
end if.
do if (nval = 2 and (mmodval = 999)).
print/title = 'The moderator variable is
dichotomous'.
end if.
do if (mmodval <> 999 and mcmmod = 0 and
((mmodval < mdmin) or (mmodval > mdmax))).
print/title = 'Warning: Moderator value specified
is outside of the range of the data'.
end if.
do if (plot = 1).
do if (mcfoc = 0 and mcmmod=0).
compute cnms = {t(nms((ncol(x)-2):(ncol(x)-
1),1)), 'yhat'}.
end if.
do if (ovals = 2).
compute prob =
exp(focvals(:,3))/(1+exp(focvals(:,3))).
compute focvals = {focvals, prob}.
compute cnms = {cnms, 'prob'}.
end if.
print/title =
'*****
*****!

```

```
print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (ovals = 2).
do if (mcfoc=0).
print/title = 'DATA LIST FREE/Eco_Innovation
Opra_Pract logodds prob.' /space=1.
end if.
do if (mcmod=0).
print/title = 'DATA LIST FREE/Opra_Pract
Eco_Innovation logodds prob.' /space=1.
end if.
else if (ovals <> 2).
do if (mcfoc=0).
print/title = 'DATA LIST FREE/Eco_Innovation
Opra_Pract Financial_Performance.' /space=1.
end if.
do if (mcmod=0).
print/title = 'DATA LIST FREE/Opra_Pract
Eco_Innovation Financial_Performance.'
/space=1.
end if.
end if.
print/title = 'BEGIN DATA.' /space=0.
print focvals/title = ' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (ovals <> 2 and mcmod=0).
```

```

print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Financial_Performance BY Opra_Pract.'
/space=0.
end if.
do if (ovals <> 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
Financial_Performance BY Eco_Innovation.'
/space=0.
end if.
do if (ovals = 2 and mcmmod=0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Opra_Pract.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Opra_Pract.' /space=0.
end if.
do if (ovals = 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
logodds BY Eco_Innovation.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
prob BY Eco_Innovation.' /space=0.
end if.
do if (ncovs > 0).
print/title = 'NOTE: For data above, covariates
are set to their sample means.'.
end if.
end if.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.

```

```

end if.
end if.
END IF.
end if.
do if (bad > 0).
compute note(notes,1) = 9.
compute notes = notes + 1.
end if.
print/title = '***** ANALYSIS
NOTES AND WARNINGS
*****'.
loop i = 1 to errs.
do if (runerrs(i,1) = 1).
print/title = 'ERROR: One of your declared
mediators is dichotomous. This procedure
can"t be used.'.
end if.
do if (runerrs(i,1) = 2).
print/title = 'ERROR: For model 6, this
procedure limits the number of mediators to
four.'.
end if.
do if (runerrs(i,1) = 3).
print/title = 'ERROR: For models 1, 2, and 3,
only a single variable can be listed in the M
list.'.
end if.
do if (runerrs(i,1) = 4).
print/title = 'ERROR: You requested a model
involving W but did not provide a valid W
variable name.'.
end if.
do if (runerrs(i,1) = 5).
print/title = 'ERROR: You requested a model
involving Z but did not provide a valid Z
variable name.'.
end if.
do if (runerrs(i,1) = 6).

```



```
print/title = 'ERROR: You requested a model
involving Q but did not provide a valid Q
variable name.'.
end if.
do if (runerrs(i,1) = 7).
print/title = 'ERROR: You requested a model
involving V but did not provide a valid V
variable name.'.
end if.
do if (runerrs(i,1) = 8).
print/title = 'ERROR: You specified a W
variable for a model that does not need it'.
end if.
do if (runerrs(i,1) = 9).
print/title = 'ERROR: You specified a Z variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 10).
print/title = 'ERROR: You specified a Q variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 11).
print/title = 'ERROR: You specified a V variable
for a model that does not need it.'.
end if.
do if (runerrs(i,1) = 12).
print/title = 'ERROR: The variable specified for
W has already been assigned.'.
end if.
do if (runerrs(i,1) = 13).
print/title = 'ERROR: The variable specified for
Z has already been assigned.'.
end if.
do if (runerrs(i,1) = 14).
print/title = 'ERROR: The variable specified for
Q has already been assigned.'.
end if.
do if (runerrs(i,1) = 15).
print/title = 'ERROR: The variable specified for
V has already been assigned.'.
```

```
end if.  
do if (runerrs(i,1) = 16).  
print/title = 'ERROR: You did not provide a  
valid Y variable name.'  
end if.  
do if (runerrs(i,1) = 17).  
print/title = 'ERROR: The variable specified for  
Y has already been assigned.'  
end if.  
do if (runerrs(i,1) = 18).  
print/title = 'ERROR: Model 6 requires more  
than one mediator.'  
end if.  
do if (runerrs(i,1) = 19).  
print/title = 'ERROR: You have not specified a  
valid model number.'  
end if.  
do if (runerrs(i,1) = 20).  
print/title = 'ERROR: At least one and only one  
variable must be listed for X.'  
end if.  
do if (runerrs(i,1) = 21).  
print/title = 'ERROR: At least one and only one  
variable must be listed for Y.'  
end if.  
do if (runerrs(i,1) = 22).  
print/title = 'ERROR: Iteration didn't converge  
to a solution. Interpret results with caution.'  
end if.  
do if (runerrs(i,1) = 23).  
print/title = 'ERROR: You specified a clustering  
variable that does not exist in your variable  
list.'  
end if.  
do if (runerrs(i,1) = 24).  
print/title = 'ERROR: You specified a clustering  
variable that has already been assigned.'  
end if.  
do if (runerrs(i,1) = 25).
```

```

print/title = 'ERROR: One or more of your M
variables is not listed in the variables list.'.
end if.
do if (runerrs(i,1) = 26).
print/title = 'ERROR: A maximum of 20 cluster
units is allowed. Use multilevel modeling
instead.'.
end if.
do if (runerrs(i,1) = 27).
print/title = 'ERROR: One of the variables in
your model is a constant.'.
end if.
do if (runerrs(i,1) = 28).
print/title = 'ERROR: Dichotomous Y is not
permitted with WS option.'.
end if.
do if (runerrs(i,1) = 29).
print/title = 'ERROR: Insufficient number of
variables in vars= list when using WS option.'.
end if.
do if (runerrs(i,1) = 30).
print/title = 'ERROR: Too many variables in
vars= list when using WS option. Covariates
not allowed.'.
end if.
do if (runerrs(i,1) = 31).
print/title = 'ERROR: mmodval and wmodval
can"t both be set to zero with contrast option.'.
end if.
do if (runerrs(i,1) = 32).
print/title = 'ERROR: You did not provide a
valid X variable name.'.
end if.
do if (runerrs(i,1) = 33).
print/title = 'ERROR: PROCESS requires all
variable names to be eight characters or
fewer.'.
print/title = '      Please shorten variable names
and reexecute.'/space=0.
end if.

```

```

do if (runerrs(i,1)=34).
print/title = 'ERROR: X and M cannot both be
specified as multicategorical.'.
end if.
do if (runerrs(i,1)=35).
print/title = 'ERROR: Multicategorical variable
specification available only for models 1 and
4.'.
end if.
do if (runerrs(i,1)=36).
print/title = 'ERROR: Categorical variables
cannot have more than 10 categories.'.
end if.
do if (runerrs(i,1)=37).
print/title = 'ERROR: Each group must have at
least two cases.'.
end if.
do if (runerrs(i,1)=38).
print/title = 'ERROR: Cluster option not
available with multicategorical variables.'.
end if.
do if (runerrs(i,1)=39).
print/title = 'ERROR: A maximum of 10
variables is allowed in the M= list.'.
end if.
do if (runerrs(i,1) = 41).
print/title = 'ERROR: MCM option not permitted
in model 4.'.
end if.
end loop.
do if (errs = 0).
do if (boot > 1 or mc > 0).
do if (bconoff = 1 and boot > 1).
print boot/title = 'Number of bootstrap samples
for bias corrected bootstrap confidence
intervals:'//format = F8.0.
end if.
do if (bconoff = 0 and boot > 1).

```

```

print boot/title = 'Number of bootstrap samples
for percentile bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (mc > 1).
print mc/title = 'Number of samples for Monte
Carlo confidence intervals:'/format = F8.0.
end if.
do if (booterr = 1).
compute badend = badend(1,2:ncol(badend)).
print badend/title = 'WARNING: Bootstrap CI
endpoints below not trustworthy. Decrease
confidence or increase bootstraps'/format =
F10.4.
end if.
end if.
print conf/title = 'Level of confidence for all
confidence intervals in output:'/format = F8.2.
do if ((center = 1 or ws=1) and (ncol(centvar) >
1)).
compute centvar = centvar(1,2:ncol(centvar)).
print centvar/title = 'NOTE: The following
variables were mean centered prior to
analysis:'/format = a8.
end if.
loop i = 1 to notes.
do if (note(i,1) = 1).
print/title = 'NOTE: Confidence level restricted
to between 50 and 99.9999%. 95% confidence
is provided in output'.
end if.
do if (note(i,1) = 2).
print/title = 'NOTE: Effect size measures for
indirect effects are not available for models
with dichotomous outcomes'.
end if.
do if (note(i,1) = 3).
print/title = 'NOTE: All standard errors for
continuous outcome models are based on the
HC3 estimator'.

```

```

end if.
do if (note(i,1) = 6).
print/title = 'NOTE: The number of bootstrap
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 7).
print/title = 'NOTE: The Johnson-Neyman
method is available only for Models 1 and 3'.
end if.
do if (note(i,1) = 8).
print/title = 'NOTE: The Johnson-Neyman
method cannot be used with a dichotomous
moderator'.
end if.
do if (note(i,1)=9).
print bad/title = 'NOTE: Some bootstrap
samples had to be replaced. The number of
such replacements was:'.
end if.
do if (note(i,1) = 11).
print nmiss/title = 'NOTE: Some cases were
deleted due to missing data. The number of
such cases was:'.
end if.
do if (note(i,1) = 12).
print /title = 'NOTE: Monte Carlo method
available only for models 4 and 5.
Bootstrapping was used instead.'.
end if.
do if (note(i,1) = 13).
print/title = 'NOTE: The number of Monte Carlo
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 19).
print/title = 'NOTE: Effect sizes not available for
within-subject analyses.'.
end if.
do if (note(i,1) = 16).

```

```
print/title = 'NOTE: Normal theory tests not
available for within-subject analyses.'.
end if.
do if (note(i,1) = 17).
print/title = 'NOTE: Monte Carlo confidence
intervals not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 18 and warnrep=0).
print/title = 'WARNING: You have requested
OLS estimation with a dichotomous criterion.'.
print/title = 'Interpret model coefficients and
inferential statistics with caution.'/space=0.
compute warnrep=1.
end if.
do if (note(i,1) = 20).
print/title = 'NOTE: Saving of bootstrap
estimates not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 22).
print/title = 'NOTE: Effect size option with
covariates requires covariates in models of M
and Y.'.
end if.
do if (note(i,1) = 23).
print/title = 'NOTE: Johnson-Neyman method
not available with multicategorical X'.
end if.
do if (note(i,1) = 24).
print/title = 'NOTE: MMODVAL option not
available with a multicategorical moderator'.
end if.
do if (note(i,1) = 25).
print/title = 'NOTE: Omnibus test for relative
indirect effects is not available with the MC
option.'.
end if.
do if (note(i,1) = 26).
```

| | | |
|-----------|----------------|---|
| | | <pre> print/title = 'NOTE: CONTRAST option not available with multicategorical X.'. end if. do if (note(i,1) = 27). print/title = 'NOTE: EFFSIZE option not available with multicategorical X.'. end if. do if (note(i,1) = 28). print/title = 'NOTE: NORMAL option not available with multicategorical X.'. end if. do if (note(i,1) = 31). print/title = 'NOTE: The TOTAL option is not available when Y is dichotomous.'. end if. do if (note(i,1) = 32). print/title = 'NOTE: Kappa-squared is disabled from output as of version 2.16.'. end if. do if (note(i,1) = 33). print/title = 'NOTE: COVMY option ignored in models 1, 2, and 3.'. end if. end loop. end if. end matrix. </pre> |
| Resources | Processor Time | 00:00:04.94 |
| | Elapsed Time | 00:00:04.97 |

Run MATRIX procedure:

***** PROCESS Procedure for SPSS Release 2.16.3 *****

Written by Andrew F. Hayes, Ph.D. www.afhayes.com
Documentation available in Hayes (2013). www.guilford.com/p/hayes3

Model = 4
Y = Financia
X = Opra_Pra
M = Eco_Inno

Sample size

Outcome: Eco_Inno

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|-----------|--------|----------|-------|
| .8594 | .7385 | .3016 | 2411.8474 | 1.0000 | 854.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|-------|-------|---------|-------|-------|-------|
| constant | .5645 | .0677 | 8.3404 | .0000 | .4317 | .6974 |
| Opra_Pra | .8185 | .0167 | 49.1106 | .0000 | .7858 | .8512 |

Outcome: Financia

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|-----------|--------|----------|-------|
| .9024 | .8143 | .2252 | 1870.1558 | 2.0000 | 853.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|-------|-------|---------|-------|-------|-------|
| constant | .1797 | .0608 | 2.9546 | .0032 | .0603 | .2991 |
| Eco_Inno | .5488 | .0296 | 18.5616 | .0000 | .4908 | .6069 |
| Opra_Pra | .3900 | .0282 | 13.8478 | .0000 | .3347 | .4453 |

***** DIRECT AND INDIRECT EFFECTS *****

Direct effect of X on Y

| Effect | SE | t | p | LLCI | ULCI |
|--------|-------|---------|-------|-------|-------|
| .3900 | .0282 | 13.8478 | .0000 | .3347 | .4453 |

Indirect effect of X on Y

| | Effect | Boot SE | BootLLCI | BootULCI |
|----------|--------|---------|----------|----------|
| Eco_Inno | .4492 | .0256 | .4028 | .5022 |

Normal theory tests for indirect effect

| Effect | se | Z | p |
|--------|-------|---------|-------|
| .4492 | .0259 | 17.3597 | .0000 |

***** ANALYSIS NOTES AND WARNINGS *****

Number of bootstrap samples for bias corrected bootstrap confidence intervals:

5000

Level of confidence for all confidence intervals in output:

95.00

----- END MATRIX -----

restore.

Matrix

| Notes | | |
|----------------|--------------------------------|----------------------|
| Output Created | | 20-OCT-2020 16:00:57 |
| Comments | | |
| Input | Active Dataset | DataSet1 |
| | Filter | <none> |
| | Weight | <none> |
| | Split File | <none> |
| | N of Rows in Working Data File | 856 |

Syntax

```
matrix.  
get dat/file = */variables =  
Financial_Performance Comm_Pract  
Eco_Innovation /names = vnames/missing =  
9999.  
compute ninit = nrow(dat).  
get dat/file = */variables =  
Financial_Performance Comm_Pract  
Eco_Innovation /names = vnames/missing =  
omit.  
get tmp/file = */variables =  
Financial_Performance /names =  
yname/missing = omit.  
get tmp2/file = */variables = Comm_Pract  
/names = xname/missing = omit.  
get tmp/file = */variables = Eco_Innovation  
/names = mnames/missing = omit.  
get tmp/file = */variables = w999999t z999999t  
v999999t q999999t.  
compute wname=tmp(1,1).  
do if (wname = ' ').  
compute wname = 'xxx'.  
end if.  
compute zname=tmp(1,2).  
do if (zname = ' ').  
compute zname = 'xxx'.  
end if.  
compute vname=tmp(1,3).  
do if (vname = ' ').  
compute vname = 'xxx'.  
end if.  
compute qname=tmp(1,4).  
do if (qname = ' ').  
compute qname = 'xxx'.  
end if.  
compute n = nrow(dat).  
compute p0=-.322232431088.  
compute p1 = -1.  
compute p2 = -.342242088547.  
compute p3 = -.0204231210245.
```

```

compute p4 = -.0000453642210148.
compute q0 = .0993484626060.
compute q1 = .588581570495.
compute q2 = .531103462366.
compute q3 = .103537752850.
compute q4 = .0038560700634.
compute badend = 0.
compute priorlo = -9999999.
compute priorhi = 9999999.
compute criterr = 0.
compute cluster = 0.
compute clsdmy = 0.
compute jndich = 0.
compute wvdich=0.
compute zqdich=0.
compute mod74dic=0.
compute warnrep=0.
compute contrast = trunc( 0 ).
do if (contrast > 2 or contrast < 0).
compute contrast = 0.
end if.
compute booterr = 0.
compute effsize = ( 0 = 1).
compute note = make(10,1,0).
compute notes = 1.
compute iterr = 0.
compute clsmtn = 0.
compute quantile = ( 0 = 1).
compute mmodval= 999.
compute jn = ( 0 = 1).
compute covcoeff=( 0 =1).
compute center = ( 0 = 1).
compute detail = ( 1 = 1).
compute coeffci = ( 1 = 1).
compute conf = 95.
compute longname = ( 1 =1).
compute bconoff=( 0 <> 1).
compute covmy = trunc( 0 ).
do if (covmy < 0 or covmy > 2).
compute covmy = 0.

```

```

end if.
do if (trunc(conf) >= 100 or (trunc(conf) <=
50)).
compute conf = 95.
compute note(notes,1) = 1.
compute notes = notes + 1.
end if.
do if (n < ninit).
compute nmiss = ninit-n.
compute note(notes,1) = 11.
compute notes = notes + 1.
end if.
compute errs = 0.
compute quantd = {0,0,0,0,0,0}.
compute quantc = {0,0,0,0,0,0}.
compute mcheck = 0.
compute ttt = 0.
compute plot = trunc( 0 ).
compute runerrs = make(50,1,0).
compute model = trunc( 4 ).
compute normal = 1.
compute ws= ( 0 =1).
compute olsdichm= ( 0 =1).
compute olsdichy= ( 0 =1).
compute mcx=trunc( 0 ).
compute mcm=trunc( 0 ).
do if (ws = 1).
do if (effsize = 1).
compute note(notes,1) = 19.
compute notes = notes + 1.
compute effsize=0.
end if.
do if (normal=1).
compute note(notes,1) = 16.
compute notes = notes + 1.
compute normal=0.
end if.
end if.
do if (model <> 4).
compute ws = 0.

```

```

end if.
do if (model > 0 and model < 4 and covmy <>
0).
compute covmy=0.
compute note(notes,1) = 33.
compute notes = notes + 1.
end if.
print/space=0.
do if (jn = 1 and model <> 1 and model <> 3).
compute note(notes,1) = 7.
compute notes = notes + 1.
end if.
do if ((model > 76) or (model < 1)).
compute model = 77.
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 19.
end if.
compute itprob=0.
do if (mcx < 0).
compute mcx=0.
end if.
do if (mcx > 4).
compute mcx=0.
end if.
do if (mcm < 0).
compute mcm=0.
end if.
do if (mcm > 4).
compute mcm=0.
end if.
do if (mcx > 0 and mcm > 0).
compute errs=errs+1.
compute runerrs(errs,1)=34.
compute criterr=1.
end if.
do if ((model <> 1 and model <> 4) and (mcx >
0 or mcm > 0)).
compute errs=errs+1.
compute runerrs(errs,1)=35.

```

```

compute criterr=1.
end if.
do if ((model = 4) and (mcm > 0)).
compute errs=errs+1.
compute runerrs(errs,1)=41.
compute criterr=1.
end if.
do if ((mcx > 0) and jn <> 0).
compute jn=0.
compute note(notes,1) = 23.
compute notes = notes + 1.
end if.
do if (mmodval <> 999 and mcm > 0).
compute note(notes,1) = 24.
compute notes = notes + 1.
end if.
compute toteff = 0.
compute toteff = (( 0 = 1)*( 4 = 4 or 4 = 6)).
compute varorder = 2.
compute hc3 = ( 0 <> 0).
compute cname = 'xxx'.
compute centvar = {'xxx'}.
compute nmeds = ncol(mnames).
do if (longname=0).

```

```
compute criterr = 1.
```

```
compute errs = errs+1.
```

```
compute runerrs(errs,1) = 33.
```


|

|

|

end if.

compute modelm =

```
{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,2;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,3;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,5;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,6;  
1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,7;  
1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,8;  
1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,9;  
1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,3,10;  
1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,5,11;  
1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,0,5,12;  
1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5,13;  
0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,2,14;  
0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,2,15;  
0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,4,16;  
0,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,4,17;  
0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,6,18;  
0,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,6,19;  
0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,6,20;  
1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,7,21;  
1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,7,22;  
1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,23;  
1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,24; 1,1,1,1
```

```
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,25;  
1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,26;  
1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,27;  
1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,7,28;  
1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,7,29;  
1,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,30;
```

1,1,0,1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,31;
1,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,32;
1,1,1,1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,33;
1,1,1,1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,34;
1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,35;
1,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,36;
1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,37;
1,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,38;
1,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,39;
1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,40;
1,0,0,1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,0,41;
1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,42;
1,0,0,1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,0,43;
1,0,0,1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,0,44;
1,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,45;
1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,46;
1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,47;
1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,48;
1,1,0,1,1,0,1,1,0,0,0,

0,0,0,0,0,0,0,0,0,49;
1,1,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,50;
1,1,1,1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,51;
1,1,1,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,52;
1,1,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,53;
1,1,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,54;
1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,55;
1,1,1,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,56;
1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,57;
1,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,58;
1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,59;
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,60;
1,1,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,61;
1,1,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,62;
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,0,0,0,0,63;
1,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,64;
1,0,0,1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,65;
1,0,0,1,0,0,0,0,0,0,1,0,0,1,0,0,0,0,0,66;
1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,67;
1,1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,68;

```

1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,69;
1,0,0,1,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,70;
1,0,0,1,0,0,1,0,0,1,0,0,1,1,1,0,0,0,0,71;
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,1,1,0,0,72;
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,1,1,0,0

,73; 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,74;
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,75;
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,1,0,0,0,76;
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,77}.
compute wm = modelm(model, 1).
compute zm = modelm(model, 2).
compute wzm = modelm(model, 3).
compute vy = modelm(model, 4).
compute qy = modelm(model, 5).
compute vqy = modelm(model, 6).
compute wy = modelm(model, 7).
compute zy = modelm(model, 8).
compute wzy = modelm(model, 9).
compute vxy = modelm(model, 10).
compute qxy = modelm(model, 11).
compute vqxy = modelm(model, 12).
compute wmy = modelm(model, 13).
compute wvmy = modelm(model, 14).
compute wvxy = modelm(model, 15).
compute zmy = modelm(model, 16).
compute wzmy = modelm(model, 17).
compute xmy = modelm(model, 18).
compute imm=modelm(model, 19).
do if (model = 74).
compute vdich=0.
end if.
do if (ncol(xname) <> 1).
compute errs = errs+1.
compute runerrs(errs, 1) = 20.
compute criterr = 1.
end if.
do if (ncol(mnames) > 10).
compute errs = errs+1.
compute runerrs(errs, 1) = 39.

```

```

compute criterr = 1.
end if.
do if (ncol(yname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 21.
compute criterr = 1.
end if.
do if (( 999 =0) and ( 999 =0) and (model=3 or
model=2) and (contrast=1)).
compute errs = errs+1.
compute runerrs(errs,1) = 31.
compute criterr = 1.
end if.
compute xlist = (wm or zm or wzm or wy or zy
or wzy or vxy or qxy or vqxy or wvxy or xmy).
compute mlist = (vy or qy or vqy or zmy or wmy
or wzmy or xmy or (model < 4)).
compute bad=0.
do if (criterr = 0).
compute werr = 0.
compute verr = 0.
compute qerr = 0.
compute zerr = 0.
compute yerr = 1.
compute xerr = 1.
compute wlist = (wm or wzm or wy or wzy or
wm or wvmy or wvmy or wvxy or wzmy).
do if (wlist = 1 and wname = 'xxx').
compute werr = 1.
compute wlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (wlist = 1 and ((wname = qname) or
(wname = vname) or (wname = zname) or
(wname = xname) or (wname = yname))).
compute werr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 12).
end if.

```

```

compute zlist = (zm or wzm or zy or wzy or
zmy or wzmy).
do if (zlist = 1 and zname = 'xxx').
compute zerr = 1.
compute zlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (zlist = 1 and ((zname = qname) or
(zname = vname) or (zname = wname) or
(zname = xname) or (zname = yname))).
compute zerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 13.
end if.
compute qlist = (qy or vqy or qxy or vqxy).
do if (qlist = 1 and qname = 'xxx').
compute qerr = 1.
compute qlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (qlist = 1 and ((qname = zname) or
(qname = vname) or (qname = wname) or
(qname = xname) or (qname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 14.
end if.
compute vlist = (vy or vqy or vxy or vqxy or
wvmy or wvxy).
do if (vlist = 1 and vname = 'xxx').
compute verr = 1.
compute vlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (vlist = 1 and ((vname = zname) or
(vname = qname) or (vname = wname) or
(vname = xname) or (vname = yname))).

```

```

compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 15.
end if.
do if (wlist = 0 and wname <> 'xxx').
compute werr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 8.
end if.
do if (zlist = 0 and zname <> 'xxx').
compute zerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 9.
end if.
do if (qlist = 0 and qname <> 'xxx').
compute qerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 10.
end if.
do if (vlist = 0 and vname <> 'xxx').
compute verr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 11.
end if.
do if (hc3 = 1).
compute note(notes,1) = 3.
compute notes = notes+1.
end if.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute cons = make(n,1,1).
compute temp = (n*sscp(dat))-
(t(csum(dat))*(csum(dat))).
compute temp = temp/(n*(n-1)).
compute temp = csum((((diag(temp))=0)).
compute temp2=1.
do if (ws = 1).

```

```

compute temp2 = (n*sscp(tmp2))-
(t(csum(tmp2))*(csum(tmp2))).
compute temp2 = temp2/(n*(n-1)).
end if.
do if ((temp > 0 and ws = 0) or (temp > 0 and
ws = 1 and temp2 > 0)).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 27.
end if.
compute nmeds = ncol(mnames).
compute mcmats=ident(nmeds*2).
compute mccoef=make(nmeds*2,1,0).
compute sobel = make(nmeds,4,-999).
do if (model = 6 and nmeds > 4).
compute errs = errs+1.
compute runerrs(errs,1)=2.
end if.
do if (model < 4 and nmeds > 1).
compute errs = errs+1.
compute runerrs(errs,1) = 3.
end if.
compute nmods = (model = 74).
compute bad = 0.
compute intcnt = 1.
compute modvals = 0.
compute modvalsd = 0.
compute yintemp = {'int_1', 'int_2', 'int_3',
'int_4', 'int_5', 'int_6', 'int_7', 'int_8', 'int_9',
'int_10', 'int_11', 'int_12', 'int_13', 'int_14',
'int_15'}.
compute yintemp = {yintemp, 'int_16', 'int_17',
'int_18', 'int_19', 'int_20', 'int_21', 'int_22',
'int_23', 'int_24', 'int_25', 'int_26', 'int_27',
'int_28'}.
compute yintemp = {yintemp, 'int_29', 'int_30',
'int_31', 'int_32', 'int_33', 'int_34', 'int_35',
'int_36', 'int_37', 'int_38', 'int_39', 'int_40'}.

```



```

compute yintemp = {yintemp, 'int_41', 'int_42',
'int_43', 'int_44', 'int_45', 'int_46', 'int_47',
'int_48', 'int_49', 'int_50', 'int_51', 'int_52'}.
compute yintemp = {yintemp, 'int_53', 'int_54',
'int_55', 'int_56', 'int_57', 'int_58', 'int_59',
'int_60', 'int_61', 'int_62', 'int_63', 'int_64'}.
compute
cntname={('C1'),('C2'),('C3'),('C4'),('C5'),('C6'),('
C7'),('C8'),('C9'),('C10'),('C11'),('C12'),('C13'),('
C14'),('C15'),('C16'),('C17')}.
compute
cntname={cntname;('C18'),('C19'),('C20'),('C21
'),('C22'),('C23'),('C24'),('C25'),('C26'),('C27'),('
C28'),('C29'),('C30'),('C31')}.
compute
cntname={cntname;('C32'),('C33'),('C34'),('C35
'),('C36'),('C37'),('C38'),('C39'),('C40'),('C41'),('
C42'),('C43'),('C44'),('C45')}.
compute
cntname={cntname;('C46'),('C47'),('C48'),('C49
'),('C50'),('C51'),('C52'),('C53'),('C54'),('C55'),('
C56'),('C57'),('C58'),('C59')}.
compute
cntname={cntname;('C60'),('C61'),('C62'),('C63
'),('C64'),('C65'),('C66'),('C67'),('C68'),('C69'),('
C70'),('C71'),('C72'),('C73')}.
compute
cntname={cntname;('C74'),('C75'),('C76'),('C77
'),('C78'),('C79'),('C80'),('C81'),('C82'),('C83'),('
C84'),('C85'),('C86'),('C87')}.
compute
cntname={cntname;('C88'),('C89'),('C90'),('C91
'),('C92'),('C93'),('C94'),('C95'),('C96'),('C97'),('
C98'),('C99'),('C100'),('C101')}.
compute
cntname={cntname;('C102'),('C103'),('C104'),('
C105')}.
compute apathnam={'a path', 'a1 path', 'a2
path', 'a3 path', 'a4 path', 'a5 path', 'a6 path',
'a7 path', 'a8 path', 'a9 path', 'a10 path'}.

```

```

compute bpathnam={'b path'; 'b1 path'; 'b2
path'; 'b3 path'; 'b4path'; 'b5path'; 'b6 path'; 'b7
path'; 'b8 path'; 'b9 path'; 'b10 path'}.
compute modvnm = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute modvnm2 = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute mlab = { ' M1 ='; ' M2 ='; ' M3 ='; '
M4 ='; ' M5 ='; ' M6 ='; ' M7 ='; ' M8 ='; '
M9 ='; ' M10 ='}.
compute m = make(n,nmeds,1).
compute mmat = make(16,nmeds,0).
compute ymat = make(8,nmeds,0).
compute deco = make(10,1,0).
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
compute iterate = abs(trunc( 10000 )).
compute converge = abs( 0.00000001 ).
compute boot = abs(trunc( 5000 )).
do if (ws=1 and 0 > 0).
compute note(notes,1)=17.
compute notes=notes+1.
end if.
compute mc=(abs(trunc( 0 ))*(1-ws)).
do if (mc > 0 and model > 5).
do if (boot = 0).
compute boot=mc.
end if.
compute mc=0.
compute note(notes,1)=12.
compute notes = notes+1.
end if.
compute adjust = 0.
do if (mc > 0 and (model > 3 and model < 6)).
compute boot=0.
compute bconoff=0.
end if.
do if (boot > 0 and mc > 0).

```

```

compute mc = 0.
end if.
compute savboot = 0.
compute saveboot = ( 0 = 1).
do if (ws=1 and saveboot=1).
compute saveboot=0.
compute note(notes,1) = 20.
compute notes = notes + 1.
compute normal=0.
end if.
do if (saveboot =1 and boot > 0 and model >
3).
compute savboot = 1.
end if.
do if (boot <> 0 or mc <> 0).
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
loop.
compute cilow = rnd(bootsz*(1-(conf/100))/2).
compute cihigh =
trunc((bootsz*(conf/100)+(bootsz*(1-
(conf/100))/2)))+1.
do if (cilow < 1 or cihigh > bootsz).
compute
bootsz=trunc((bootsz+1000)/1000)*1000.
compute adjust = 1.
end if.
end loop if (cilow gt 0 and cihigh le bootsz).
compute boot=bootsz.
do if (mc > 0).
compute mc = bootsz.
end if.
do if (boot > 0 and mc > 0 and (model > 3 and
model < 6)).
compute boot=0.
end if.
do if (adjust = 1 and boot > 0).
compute note(notes,1)=6.

```

```

compute notes = notes+1.
end if.
do if (adjust = 1 and mc > 0).
compute note(notes,1)=13.
compute notes = notes+1.
end if.
end if.
do if (model = 6 and nmeds > 1).
compute mmpaths =
make((nmeds+2),(nmeds+2),0).
do if (nmeds = 2).
compute indboot = make(boot+1, 3, 999).
else if (nmeds = 3).
compute indboot = make(boot+1, 7, 999).
else if (nmeds = 4).
compute indboot = make(boot+1, 15, 999).
end if.
compute indlbl = {'Total:','Ind1 :','Ind2 :','Ind3
:','Ind4 :','Ind5 :','Ind6 :','Ind7 :','Ind8 :','Ind9
:','Ind10:','Ind11:','Ind12:','Ind13:','Ind14:','
Ind15:'}.
compute indlbl2 = {'Ind1','Ind2','Ind3','Ind4';
'Ind5','Ind6','Ind7','Ind8','Ind9','Ind10';
'Ind11','Ind12','Ind13','Ind14','Ind15'}.
compute indces = make(boot+1, 4, 999).
end if.
do if (model < 4).
compute boot = 0.
compute cmat = make(10,1,0).
compute zmat = make(10,1,0).
end if.
compute nvarch = make(1,ncol(dat),0).
compute wmatch = 0.
compute zmatch = 0.
compute vmatch = 0.
compute qmatch = 0.
compute mmatch=0.
compute minprobe=0.
compute maxprobe=0.
loop i = 1 to ncol(vnames).

```

```

do if (vnames(:,i)=yname).
compute y = dat(:,i).
compute nvarch(1,i)=1.
compute yerr = 0.
do if ((yname = xname) or (yname = wname) or
(yname = zname) or (yname = vname) or
(yname = qname)).
compute errs = errs+1.
compute runerrs(errs,1)=17.
end if.
end if.
do if (vnames(:,i)=xname).
compute x = dat(:,i).
compute nvarch(1,i)=1.
compute xdich = 1.
compute xerr=0.
loop jj = 1 to n.
do if ((x(jj,1) <> cmin(x)) and (x(jj,1) <>
cmax(x))).
compute xdich = 0.
break.
end if.
end loop.
compute xmean = csum(x)/n.
do if (center = 1 and (model < 4 or xlist > 0)
and mcx = 0).
compute meanvec = make(n,1, xmean).
compute x = x-meanvec.
compute centvar = {centvar, xname}.
end if.
compute xmean = csum(x)/n.
compute tmp = x-(cons*xmean).
compute xsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
do if (xdich = 0).
compute quantc(1,6) = 1.
compute matx = {xmean-xsd; xmean;
xmean+xsd}.
do if ( 999 = 999 and quantile = 0).
do if ((matx(1,1) < cmin(x)) and model = 74).
compute matx(1,1)=cmin(x).

```

```

compute minprobe=1.
end if.
do if ((matx(3,1) > cmax(x)) and model = 74).
compute matx(3,1)=cmax(x).
compute maxprobe=1.
end if.
end if.
do if (quantile = 1).
compute quantd(1,6) = 1.
compute quantc(1,6) = 0.
compute tmp = x.
compute tmp(GRADE(x)) = x.
compute matx =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (xdich = 1).
compute matx = {cmin(x); cmax(x)}.
do if (model=74).
compute matx=cmin(x).
compute mod74dic=1.
end if.
end if.
do if ( 999 <> 999).
compute matx = 999.
compute quantd(1,6) = 0.
compute quantc(1,6) = 0.
end if.
end if.
do if (werr = 0 and wlist = 1).
do if (vnames(:,i)=wname).
compute werr = 0.
compute wmatch = 1.
compute w = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(w)/n).
compute w = w-meanvec.
compute centvar = {centvar, wname}.

```

```

end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute wmean = csum(w)/n.
compute tmp = w-(cons*wmean).
compute wsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute wdich = 1.
loop jj = 1 to n.
do if ((w(jj,1) <> cmin(w)) and (w(jj,1) <>
cmax(w))).
compute wdich = 0.
break.
end if.
end loop.
do if (model = 3).
compute jndich=wdich.
compute jnmin=cmin(w).
compute jnmax=cmax(w).
end if.
do if (wdich = 0).
compute matw = {wmean-wsd; wmean;
wmean+wsd}.
do if ( 999 = 999 and quantile = 0).
do if (matw(1,1) < cmin(w)).
compute matw(1,1)=cmin(w).
compute minprobe=1.
end if.
do if (matw(3,1) > cmax(w)).
compute matw(3,1)=cmax(w).
compute maxprobe=1.
end if.
end if.
compute quantc(1,1) = 1.
do if (quantile = 1).
compute quantd(1,1) = 1.
compute quantc(1,1) = 0.
compute tmp = w.
compute tmp(GRADE(w)) = w.
compute matw =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);

```

```

tmp(trunc(n*.50),1); tmp(trunc(n*0.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (wdich = 1).
compute matw = {cmin(w); cmax(w)}.
compute wvdich=1.
compute cmaxw=cmax(w).
compute cminw=cmin(w).
end if.
do if ( 999 <> 999).
compute matw = 999.
compute quantd(1,1) = 0.
compute quantc(1,1) = 0.
end if.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm(1,1)=wname.
compute modmatp(1,1) = 1.
end if.
end if.
do if (zerr = 0 and zlist = 1).
do if (vnames(:,i)=zname).
compute zerr = 0.
compute zmatch = 1.
compute z = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(z)/n).
compute z = z-meanvec.
compute centvar = {centvar, zname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute zmean = csum(z)/n.
compute tmp = z-(cons*zmean).
compute zsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute zdich = 1.
loop jj = 1 to n.
do if ((z(jj,1) <> cmin(z)) and (z(jj,1) <>
cmax(z))).

```



```

compute zdich = 0.
break.
end if.
end loop.
do if (zdich = 0).
compute matz = {zmean-zsd; zmean;
zmean+zsd}.
do if ( 999 = 999 and quantile = 0).
do if (matz(1,1) < cmin(z)).
compute matz(1,1)=cmin(z).
compute minprobe=1.
end if.
do if (matz(3,1) > cmax(z)).
compute matz(3,1)=cmax(z).
compute maxprobe=1.
end if.
end if.
compute quantc(1,2) = 1.
do if (quantile = 1).
compute quantd(1,2) = 1.
compute quantc(1,2) = 0.
compute tmp = z.
compute tmp(GRADE(z)) = z.
compute matz = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (zdich = 1).
compute matz = {cmin(z); cmax(z)}.
compute cmaxz=cmax(z).
compute cminz=cmin(z).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matz = 999.
compute quantd(1,2) = 0.
compute quantc(1,2) = 0.
end if.
compute modmatv(1,2)=nrow(matz).

```

```

compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm(1,2)=zname.
compute modmatp(1,2) = 1.
end if.
end if.
do if (verr = 0 and vlist = 1).
do if (vnames(:,i)=vname).
compute verr = 0.
compute vmatch = 1.
compute v = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(v)/n).
compute v = v-meanvec.
compute centvar = {centvar, vname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute vmean = csum(v)/n.
compute tmp = v-(cons*vmean).
compute vsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute vdich = 1.
loop jj = 1 to n.
do if ((v(jj,1) <> cmin(v)) and (v(jj,1) <>
cmax(v))).
compute vdich = 0.
break.
end if.
end loop.
do if (vdich = 0).
compute matv = {vmean-vsd; vmean;
vmean+vsd}.
do if ( 999 = 999 and quantile = 0).
do if (matv(1,1) < cmin(v)).
compute matv(1,1)=cmin(v).
compute minprobe=1.
end if.
do if (matv(3,1) > cmax(v)).
compute matv(3,1)=cmax(v).
compute maxprobe=1.
end if.

```

```

end if.
compute quantc(1,3) = 1.
do if (quantile = 1).
compute quantd(1,3) = 1.
compute quantc(1,3) = 0.
compute tmp = v.
compute tmp(GRADE(v)) = v.
compute matv = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (vdich = 1).
compute matv = {cmin(v); cmax(v)}.
compute wvdich=1.
compute cmaxv=cmax(v).
compute cminv=cmin(v).
end if.
do if ( 999 <> 999).
compute matv = 999.
compute quantd(1,3) = 0.
compute quantc(1,3) = 0.
end if.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm(1,3)=vname.
compute modmatp(1,3) = 1.
end if.
end if.
do if (qerr = 0 and qlist = 1).
do if (vnames(:,i)=qname).
compute qerr = 0.
compute qmatch = 1.
compute q = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(q)/n).
compute q = q-meanvec.
compute centvar = {centvar, qname}.
end if.
compute nvarch(1,i)=1.

```

```

compute nmods = nmods + 1.
compute qmean = csum(q)/n.
compute tmp = q-(cons*qmean).
compute qsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute qdich = 1.
loop jj = 1 to n.
do if ((q(jj,1) <> cmin(q)) and (q(jj,1) <>
cmax(q))).
compute qdich = 0.
break.
end if.
end loop.
do if (qdich = 0).
compute matq = {qmean-qsd; qmean;
qmean+qsd}.
do if ( 999 = 999 and quantile = 0).
do if (matq(1,1) < cmin(q)).
compute matq(1,1)=cmin(q).
compute minprobe=1.
end if.
do if (matq(3,1) > cmax(q)).
compute matq(3,1)=cmax(q).
compute maxprobe=1.
end if.
end if.
compute quantc(1,4) = 1.
do if (quantile = 1).
compute quantd(1,4) = 1.
compute quantc(1,4) = 0.
compute tmp = q.
compute tmp(GRADE(q)) = q.
compute matq = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (qdich = 1).
compute matq = {cmin(q); cmax(q)}.
compute cmaxq=cmax(q).
compute cminq=cmin(q).

```

```

compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matq = 999.
compute quantd(1,4) = 0.
compute quantc(1,4) = 0.
end if.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm(1,4)=qname.
compute modmatp(1,4) = 1.
end if.
end if.
do if (vnames(:,i)=cname).
compute cld = dat(:,i).
compute cvname = vnames(:,i).
compute nvarch(1,i)=1.
compute clsmatch = 1.
end if.
loop j = 1 to ncol(mnames).
do if (vnames(:,i)=mnames(1,j)).
compute mmatch = mmatch + 1.
compute m(:,j)=dat(:,i).
do if (center = 1 and nvarch(1,i) = 0 and mlist >
0 and mcm = 0).
compute meanvec = make(n,1,csum(m(:,j))/n).
compute m(:,j) = m(:,j)-meanvec.
compute centvar = {centvar, mnames(1,j)}.
end if.
compute nvarch(1,i)=1.
compute dichm = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute dichm = 0.
break.
end if.
end loop.
do if ((dichm = 1) and (olstdichm = 1)).
compute note(notes,1) = 18.

```

```

compute notes = notes + 1.
end if.
do if (dichm = 1 and model > 3 and mcheck = 0
and oldsdichm = 0).
compute errs= errs+1.
compute runerrs(errs,1) = 1.
compute mcheck = 1.
end if.
do if ((model <= 3) and (ncol(mnames)=1)).
compute nmods = nmods + 1.
compute mmean = csum(m(:,j))/n.
compute tmp = m(:,j)-(cons*mmean).
compute msd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute mdich = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute mdich = 0.
break.
end if.
end loop.
do if (model = 1).
compute jndich=mdich.
compute jnmin=cmin(m(:,j)).
compute jnmax=cmax(m(:,j)).
end if.
do if (mdich = 0).
compute matm = {mmean-msd; mmean;
mmean+msd}.
do if ( 999 = 999 and quantile = 0).
do if (matm(1,1) < cmin(m(:,j))).
compute matm(1,1)=cmin(m(:,j)).
compute minprobe=1.
end if.
do if (matm(3,1) > cmax(m(:,j))).
compute matm(3,1)=cmax(m(:,j)).
compute maxprobe=1.
end if.
end if.
compute quantc(1,5) = 1.

```

```

do if (quantile = 1).
compute quantd(1,5) = 1.
compute quantc(1,5) = 0.
compute tmp = m(:,j).
compute tmp(GRADE(m(:,j))) = m(:,j).
compute matm = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (mdich = 1).
compute matm = {cmin(m); cmax(m)}.
end if.
do if ( 999 <> 999).
compute matm = 999.
compute quantd(1,5) = 0.
compute quantc(1,5) = 0.
end if.
compute modmatv(1,5)=nrow(matm).
compute modmat((1:nrow(matm)), 5) = matm.
compute modvnm(1,5)=mnames(1,j).
compute modmatp(1,5) = 1.
end if.
end if.
end loop.
end loop.
do if (minprobe=1).
compute note(notes,1) = 14.
compute notes = notes + 1.
end if.
do if (maxprobe=1).
compute note(notes,1) = 15.
compute notes = notes + 1.
end if.
do if (cname <> 'xxx' and clsmth = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 23.
end if.
do if (cname <> 'xxx').

```

```

do if ((cname = zname) or (cname = vname)
or (cname = wname) or (cname = xname) or
(cname = yname) or (cname = qname)).
compute errs = errs+1.
compute runerrs(errs,1) = 24.
end if.
end if.
do if (wlist = 1 and werr = 0 and wmatch = 0).
compute werr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (zlist = 1 and zerr = 0 and zmatch = 0).
compute zerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (qlist = 1 and qerr = 0 and qmatch = 0).
compute qerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (vlist = 1 and verr = 0 and vmatch = 0).
compute verr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (yerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 16.
end if.
do if (xerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 32.
end if.
do if (model = 6 and nmeds < 2).
compute errs = errs+1.
compute runerrs(errs,1) = 18.
end if.
do if (mmatch < ncol(mnames)).

```



```

compute errs=errs+1.
compute runerrs(errs,1) = 25.
end if.
do if (cname <> 'xxx').
compute cld = design(cld).
compute cluster = ncol(cld).
compute cld = cld(:,2:ncol(cld)).
compute clsdmy = ncol(cld).
do if (clsdmy > 19).
compute errs = errs+1.
compute runerrs(errs,1) = 26.
end if.
do if (mcx > 0 or mcm > 0).
compute errs = errs+1.
compute runerrs(errs,1) = 38.
end if.
end if.
compute dichy = 1.
loop jj = 1 to n.
do if ((y(jj,1) <> cmin(y)) and (y(jj,1) <>
cmax(y))).
compute dichy = 0.
break.
end if.
end loop.
do if ((dichy = 1) and (olsdichy = 1)).
compute dichy=0.
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichy = 1).
do if (toteff=1).
compute toteff=0.
compute note(notes,1) = 31.
compute notes = notes + 1.
end if.
compute jncrit=xp2*xp2.
do if (ws = 1).
compute criterr = 1.
compute errs = errs+1.

```

```

compute runerrs(errs,1) = 28.
end if.
end if.
compute ncovs = ncol(dat)-rsum(nvarch).
do if (effsize = 1 and covmy <> 0 and ncovs >
0 and model > 3 and model < 7).
compute note(notes,1) = 22.
compute notes = notes + 1.
compute effsize = 0.
end if.
do if (ws = 1 and ncovs < nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 29.
end if.
do if (ws = 1 and ncovs > nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 30.
end if.
end if.
do if (errs = 0).
do if (rsum(quantd) > 0).
compute note(notes,1) = 4.
compute notes = notes+1.
end if.
do if (rsum(quantc) > 0).
compute note(notes,1) = 5.
compute notes = notes+1.
end if.
do if (ncovs > 0).
compute c = make(n,ncovs,0).
compute cnames = {'x'}.
compute j = 1.
loop i = 1 to ncol(vnames).
do if (nvarch(1,i)) = 0.
compute c(:,j) = dat(:,i).
compute nvarch(1,i)=1.
compute j=j+1.
compute cnames = {cnames, vnames(:,i)}.

```

```

end if.
end loop.
compute cnames = cnames(1,2:ncol(cnames)).
do if (ws = 1).
compute covmean=csum(c)/n.
loop i = 1 to ncovs.
compute meanvec = make(n,1, covmean(1,i)).
compute c(:,i) = c(:,i)-meanvec.
end loop.
compute centvar = {centvar, cnames}.
compute x=c(:,1:ncovs).
do if (ncovs=nmeds).
compute ncovs=0.
end if.
end if.
end if.
compute names = {yname, xname, mnames,
wname, zname, vname, qname}.
do if (ncovs > 0).
compute names = {names, cnames}.
end if.
do if (dichy = 1 and effsize = 1).
compute note(notes,1) = 2.
compute notes = notes+1.
end if.
do if (model > 3 and model < 6).
compute indeff=make(nmeds,1,0).
compute indboot=make(boot+1,nmeds,999).
do if (mc > 0).
compute indboot=make(mc+1,nmeds,999).
end if.
do if (effsize = 1 and dichy = 0).
compute rmeff=make(boot+1,nmeds+1,999).
compute abpseff=make(boot+1,nmeds+1,999).
compute abcseff=make(boot+1,nmeds+1,999).
compute pmeff=make(boot+1,nmeds+1,999).
compute r245 = make(boot+1,1,999).
compute kappa2 = make(boot+1,1,999).
end if.
end if.

```

```

do if (model = 6 and effsize = 1 and dichy = 0).
compute
rmeff=make(boot+1,ncol(indboot),999).
compute
abpseff=make(boot+1,ncol(indboot),999).
compute
abcseff=make(boot+1,ncol(indboot),999).
compute
pmeff=make(boot+1,ncol(indboot),999).
end if.
do if (nmods > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm(1,tmp) = modvnm(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:nmods).
compute modvnm=modvnm(:,1:nmods).
compute modmatv=modmatv(:,1:nmods).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvals =
make((modmatv(1,1)*modprod(1,1)),
nmods,0).
loop i = 1 to nmods.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvals)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).

```

```

compute fnsh = fnsh+nrow(tmp).
compute modvals(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
do if (model = 74).
compute modvals=matx.
compute modvnm=xname.
end if.
compute vmat = make(8,nrow(modvals),0).
compute vmat(1,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute vmat(5,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute indeff=make(nrow(modvals),1,0).
do if (model <> 5).
compute indboot = make(((boot+1)*nmeds),
nrow(modvals), -99999999).
compute indbootp =
make(boot+1,(nmeds*nmods), -99999999).
do if (imm > 4 and imm < 8).
do if (imm = 7).
compute immmv1=matw.
compute immmv2=matv.
end if.
do if (imm = 5).
compute immmv1=matw.
compute immmv2=matz.
end if.
do if (imm = 6).
compute immmv1=matv.
compute immmv2=matq.
end if.
compute
indboop1=make((boot+1),(nmeds*(nrow(immm
v2))),-99999999).
compute
indboop2=make((boot+1),(nmeds*(nrow(immm
v1))),-99999999).

```

```

compute onesmv1=make(1,nrow(immmv1),1).
compute onesmv2=make(1,nrow(immmv2),1).
compute
nimvs=nrow(immmv1)+nrow(immmv2).
compute cmmval=0.
compute cmmlbs1={'xx'}.
compute cmmlbs2={'xx'}.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv2}.
compute
cmmlbt=make(nrow(immmv2),1,mnames(1,i)).
compute cmmlbs1={cmmlbs1;cmmlbt}.
end loop.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv1}.
compute
cmmlbt=make(nrow(immmv1),1,mnames(1,i)).
compute cmmlbs2={cmmlbs2;cmmlbt}.
end loop.
compute cmmval=cmmval(2:nrow(cmmval),:).
end if.
end if.
end if.
do if (nmods> 0).
loop i = 1 to ncol(modvals).
do if (modvnm(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm(1,i)=qname).
compute qcol = i.
end if.
end loop.
end if.
do if (dichy = 1).

```

```

compute omx = cmax(y).
compute omn = cmin(y).
compute y = (y = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute data = {cons,y,m,x}.
compute datamed = data.
compute datayed = data.
compute datanm = {'constant'; yname;
t(mnames); xname}.
compute datanmm = {'constant'; yname;
t(mnames); xname}.
compute datanmy = {'constant'; yname;
t(mnames); xname}.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
do if (model < 4 and errs = 0).
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' ', ' '}.
compute datayed = {datayed, x&*m}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(1,i) = 1.
compute vmat(2,i) = modvals(i,1).
end loop.
compute mmat = make(16,nmeds,1).
end if.
do if (model = 2 or model = 3).
compute int1 = x&*w.
compute datayed = {datayed, w, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,2).
compute vmat(3,i) = modvals(i,1).
compute vmat(4,i) = modvals(i,1)*modvals(i,2).

```

```

end loop.
end if.
do if (model = 3).
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames, ' X', wname, '', ''}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*m.
compute int2 = x&*w&*m.
compute datayed = {datayed, int1, int2}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' X', wname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (model = 4 or model = 5).
compute vmat = make(8,1,1).
end if.
compute yintkey2 = yintkey.
do if (wm = 1).
compute int1 = x&*w.
compute datamed = {datamed, w,int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, '', ''}.
compute datanmm = {datanmm; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,wcol).
end loop.
do if (zm = 1).
compute int1 = x&*z.
compute datamed = {datamed, z, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, '', ''}.
compute datanmm = {datanmm; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.

```



```

loop i = 1 to nrow(modvals).
compute vmat(3,i) = modvals(i,zcol).
end loop.
end if.
do if (wzm = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*z.
compute int2 = x&*w&*z.
compute datamed = {datamed, int1, int2}.
loop i = 1 to nrow(modvals).
compute vmat(4,i) =
modvals(i,wcol)*modvals(i,zcol).
end loop.
end if.
end if.
compute mdatacol=ncol(datamed).
compute mintkey = yintkey.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
compute medints = intcnt-1.
do if (vy = 1 or xmy = 1).
compute mp = 1.
do if (xmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,1).
end loop.
end if.
do if (vy = 1).
compute datayed = {datayed, v}.
compute datanmy = {datanmy; vname}.
compute mmodes = 1.
loop i = 1 to nrow(modvals).

```

```

compute vmat(6,i) = modvals(i,vcol).
end loop.
do if (qy = 1).
compute mp = 2.
compute datayed = {datayed,q}.
compute datanmy = {datanmy; qname}.
compute mmods = 2.
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,qcol).
end loop.
end if.
do if (vqy = 1).
compute mp = 3.
compute datayed = {datayed, v&*q}.
compute mmods = 3.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,vcol)*modvals(i,qcol).
end loop.
end if.
end if.
compute mints = make(n,(nmeds*mp),0).
loop i = 0 to (nmeds-1).
do if (i = 0 and vqy = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
vname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (xmy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))*x.

```

```

compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', xname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (qy = 1).
compute mints(:,((i*mp)+2))=m(:,(i+1))&*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqy = 1).
compute mints(:,((i*mp)+3))=m(:,(i+1))&*v&*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end loop.
compute datayed = {datayed, mints}.
end if.
compute mp = 1.
do if (wvmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,wcol)*modvals(i,vcol).
end loop.
end if.
compute mints2 = make(n,(nmeds*mp),0).
do if (wmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,wcol).
end loop.
do if (wy = 0 and model > 3).
compute datayed = {datayed, w}.

```

```

compute datanmy = {datanmy; wname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wvmy = 1).
compute datayed = {datayed,w&*v}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints2(:,((i*mp)+1))= m(:,(i+1))&*w.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wvmy = 1).
compute mints2(:,((i*mp)+2))=
m(:,(i+1))&*w&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints2}.
end if.
compute mp = 1.
do if (zmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,zcol).
end loop.
do if (wzmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,zcol)&*modvals(i,wcol).
end loop.

```

```

end if.
end if.
do if (zmy = 1).
compute mints3 = make(n,(nmeds*mp),0).
do if (zy = 0).
compute datayed = {datayed, z}.
compute datanmy = {datanmy; zname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wzmy = 1 and wzy = 0).
compute datayed = {datayed,w&*z}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints3(:,((i*mp)+1))= m(:,(i+1))&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wzmy = 1).
compute mints3(:,((i*mp)+2))=
m(:,(i+1))&*w&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', zname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints3}.
end if.
compute decoc = 1.
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.

```

```

do if (wy = 1 and model > 3).
compute datayed = {datayed, w, x&*w}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm2(1,1)=wname.
compute modmatp(1,1) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (zy = 1).
compute datayed = {datayed,z,x&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm2(1,2)=zname.
compute modmatp(1,2) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (wzy = 1).
compute datayed = {datayed,w&*z,x&*w&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.

```

```

compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vxy = 1).
compute datayed = {datayed, x&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm2(1,3)=vname.
compute modmatp(1,3) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, ',', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (qxy = 1).
compute datayed = {datayed, x&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm2(1,4)=qname.
compute modmatp(1,4) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', qname, ',', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqxy = 1).
compute datayed = {datayed, x&*v&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, 'X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.

```

```

end if.
end if.
do if (wvxy = 1).
compute datayed = {datayed, x&*w&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, 'X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute modvalsd = 0.
compute ttt = rsum(modmatp).
do if (rsum(modmatp) > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm2(1,tmp) = modvnm2(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:ttt).
compute modvnm2=modvnm2(:,1:ttt).
compute modmatv=modmatv(:,1:ttt).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvalsd =
make((modmatv(1,1)*modprod(1,1)), ttt,0).
loop i = 1 to ttt.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvalsd)).

```



```

loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvalsd(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
end if.
do if (ttt > 0).
loop i = 1 to ncol(modvalsd).
do if (modvnm2(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm2(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm2(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm2(1,i)=qname).
compute qcol = i.
end if.
end loop.
compute directv = make(nrow(modvalsd),1,1).
do if (wy = 1).
compute directv ={directv, modvalsd(:,wcol)}.
end if.
do if (zy = 1).
compute directv ={directv, modvalsd(:,zcol)}.
end if.
do if (wzy = 1).
compute directv ={directv,
(modvalsd(:,wcol)&*modvalsd(:,zcol))}.
end if.
do if (vxy = 1).
compute directv ={directv, modvalsd(:,vcol)}.
end if.
do if (qxy = 1).

```

```

compute directv = {directv, modvalsd(:,qcol)}.
end if.
do if (vqxy = 1).
compute directv = {directv,
(modvalsd(:,vcol)*modvalsd(:,qcol))}.
end if.
do if (wvxy = 1).
compute directv = {directv,
(modvalsd(:,vcol)*modvalsd(:,wcol))}.
end if.
end if.
compute ydatacol = ncol(datayed).
do if (ncovs > 0).
do if (covmy <> 2).
compute datamed = {datamed,c}.
end if.
do if (covmy <> 1).
compute datayed = {datayed,c}.
end if.
do if (covmy = 1 and model = 4 and mcx > 0).
compute datayed = {datayed,c}.
end if.
compute covmeans = csum(c)/n.
end if.
do if (cluster > 0).
compute datamed = {datamed,cld}.
compute datayed = {datayed, cld}.
compute cldmeans = csum(cld)/n.
end if.
compute mst = 3.
compute mnd = mst+nmeds-1.
compute ydatacol = ncol(datayed).
compute mdatacol = ncol(datamed).
do if (ncovs > 0).
compute datanmy = {datanmy; t(cnames)}.
do if (model > 3).
compute datanmm = {datanmm; t(cnames)}.
end if.
end if.

```

```

compute datanmy = {'constant';
datanmy(3:nrow(datanmy),1)}.
do if (model > 3).
compute datanmm = {'constant';
datanmm(3:nrow(datanmm),1)}.
end if.
compute amm = make(2,1,0).
compute abmm = make(2,1,0).
compute mnv = csum(datayed(:,2)/n).
compute mnv = make(n,1,mnv).
compute ssty = csum((datayed(:,2)-mnv)&**2).
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
do if (ws=1).
compute stddevm=diag(sigma).
compute
stddevm=sqrt(stddevm(3:(nmeds+2),1)).
end if.
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute r2xy =
(sigma(2,(3+nmeds))/(stddevy*stddevx))&**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))&**2.
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).
do if (model = 4 and nmeds = 1 and cluster = 0
and ncovs = 0).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).

```

```

compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
compute datatm = datamed.
compute dataty = datayed.
compute mdlrms2 = { '4' ; yname; xname}.
compute mdlrms = {'Model ='; ' Y ='; ' X ='
}.
do if (ws = 1).
compute mdlrms2 = { '4' ; yname}.
compute mdlrms = {'Model ='; ' Y ='}.
end if.
loop i = 1 to ncol(mnames).
compute mdlrms2 = {mdlrms2; mnames(1,i)}.
do if (i = 1 and ncol(mnames) = 1).
compute mdlrms = {mdlrms; ' M ='}.
else.
compute mdlrms = {mdlrms; mlab(i,1)}.
end if.
end loop.
do if (wname <> 'xxx').
compute mdlrms2 = {mdlrms2; wname}.
compute mdlrms = {mdlrms; ' W ='}.
end if.
do if (zname <> 'xxx').
compute mdlrms2 = {mdlrms2; zname}.
compute mdlrms = {mdlrms; ' Z ='}.

```

```

end if.
do if (vname <> 'xxx').
compute mdlms2 = {mdlms2; vname}.
compute mdlms = {mdlms; '  V = '}.
end if.
do if (qname <> 'xxx').
compute mdlms2 = {mdlms2; qname}.
compute mdlms = {mdlms; '  Q = '}.
end if.
do if (jn = 1 and model = 1 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
do if (jn = 1 and model = 3 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
compute mdpbe=0.
compute medte2=0.
do if ( 4 =4 and mcx > 0).
compute medte2=1.
end if.
do if (((mcx > 0) or (mcm > 0)) AND model =
1).
compute mdpbe=1.
compute dd={datayed(:,4),datayed(:,3)}.
do if (ncovs > 0).
compute dd={datayed(:,6:(5+ncovs)),dd}.
end if.
compute dd={datayed(:,2),dd}.
end if.
compute yes = ((nrow(modvals)=1) and
(contrast = 1) and (model = 3 or model = 2)).
compute yes2 = ((nrow(modvals)=1) and
(contrast = 1) and (model = 2)).
print/title = '***** PROCESS Procedure
for SPSS Release 2.16.3 *****'.
print/title = '      Written by Andrew F. Hayes,
Ph.D.      www.afhayes.com'.

```

```

print/title = ' Documentation available in
Hayes (2013).
www.guilford.com/p/hayes3'/space=0.
print mdlrms2/title =
*****
*****'/rnames = mdlrms/format =
a8.
do if (ncovs > 0).
print cnames/title = 'Statistical Controls: '/rlabels
= 'CONTROL=' /format a8.
end if.
print n/title = 'Sample size' /format F10.0.
do if (cluster > 0).
print cluster/rnames = cvname/title =
'Clustering Variable and Number of Clusters'.
end if.
do if (model > 3 and ( 'random' <> 'random')).
compute seedt= 'random'.
print seedt/title='Custom' /format=A12/rlabel =
'Seed:'.
end if.
do if (mdpbe=0 and medte2 = 0).
loop bt = 1 to (boot+1).
do if (bt = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef = 0.
do if (bt > 1).
loop.
compute v=trunc(uniform(n,1)*n)+1.
compute datayed = dataty(v,:).
do if (ws=1).
compute
bmeans=csum(datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed)))/n.
compute cntmat=make(n,nmeds,1).
compute bmeans2=bmeans.
do if (nmeds > 1).
compute bmeans2=mdiag(bmeans).

```

```

end if.
compute cntmat=cntmat*bmeans2.
compute datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed))=datayed(:,(ncol(data
yed)-nmeds+1):ncol(datayed))-cntmat.
end if.
compute detcheck=det(t(datayed)*datayed).
compute rk=(detcheck <> 0).
do if (model > 3).
compute datamed = datatm(v,:).
compute detcheck=det(t(datamed)*datamed).
do if (rk=1).
compute rk=(detcheck <> 0).
end if.
end if.
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
compute temp=diag(sigma).
compute bad = bad+(1-rk).
compute false = 1.
end loop if (rk = 1).
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nme
ds)).
do if (model = 4 and nmeds = 1 and ncovs = 0
and cluster = 0).
compute r2xy =
(sigma(2,4)/(stddevy*stddevx))&**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))&**2.
compute sstot = sigma(2,2)*(n-1).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).

```

```

compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
end if.
do if (model > 3).
loop im = 1 to nmeds.
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant';
datanmm((2+nmeds):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
do if (model = 6).
do if (im = 1).
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
end if.
do if (im > 1).

```



```

compute xm={cons, datamed(:,3:(im+1)),
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant'; datanmm(2:im,1);
datanmm((mnd):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
compute
mmpaths((im+1),(2:im))=t(coeff(2:im,1)).
end if.
end if.
do if (ws = 1).
compute
coeff(1,1)=csum(datamed(:,(2+im)))/n.
end if.
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
do if (bt = 1).
compute resid=datamed(:,(2+im))-xm*coeff.
compute sse =cssq(resid).
compute mse = sse/(n-ncol(xm)).
compute mnv = csum(data(:,(2+im)))/n.
compute mnv = make(n,1,mnv).
compute sstm = csum((data(:,(2+im))-
mnv)&^2).
compute k3 = nrow(coeff).
do if (hc3 = 1).
compute h = xm(:,1).
loop i3=1 to n.
compute h(i3,1)= xm(i3,:)*invXtX*t(xm(i3,:)).
end loop.
loop i3=1 to k3.
compute xm(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xm(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xm(:,i3) = sqrt(mse)&*xm(:,i3).

```

```

end loop.
end if.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute hccov=invXtX*t(xm)*xm*invXtX.
compute mcmats(im,im)=hccov(2,2).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*hccov*lmat)*((t(lmat)
)*coeff))/dfnum.
compute coeff = coeff(1:(nrow(coeff)-
clsdmy),1).
compute mcoeff(im,1)=coeff(2,1).
compute standerr =
sqrt(diag(invXtX*t(xm)*xm*invXtX)).
compute standerr = standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1).
compute standerr(1,1)=stddevm(im,1)/sqrt(n).
end if.
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-
ncol(xm)))).
compute temp=(n-ncol(xm)).
do if (ws=1).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute temp=(n-1).
end if.
compute xd = abs(xp2).
compute temp = (temp* (exp((temp-
(5/6))*((xd/(temp-(2/3)+(.11/temp)))*(xd/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = coeff-
sqrt(abs(temp))*standerr.
compute temp2 =
coeff+sqrt(abs(temp))*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (ws=1).

```

```

compute op=op(1,:).
compute xnmn =
apathnam(((nmeds>1)+1+(im-1)),1).
end if.
compute sobel(im,1) = coeff(2,1).
compute sobel(im,2)=standerr(2,1).
compute temp = mnames(1,im).
compute r2full = 1-(sse/sstm).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute summ = {sqrt(r2full), r2full, mse,
Fratio, dfnum, dfden, pfr}.
do if (detail = 1).
print temp/title =
*****
*****/rlabels = 'Outcome:'/format
= A8.
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels 'R',
'R-sq', 'MSE','F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
print op/title = 'Model'/rnames = xnmn/clabels
'coeff', 'se', 't', 'p', 'LLCI', 'ULCI'/format = F10.4.
do if (covcoeff=1 and ws <> 1).
compute
hccovtmp=hccov(1:nrow(op),1:nrow(op)).
compute cnamestp=t(xnmn).
print hccovtmp/title='Covariance matrix of
regression parameter
estimates'/rnames=xnmn/cnames=cnamestp/f
ormat= F10.4.
end if.
do if (nmods > 0 and nrow(mintkey) > 1).
print mintkey/title = 'Product terms key:'/format
= A8.
end if.
do if (ws = 1).
compute wsdf=n-1.

```

```

print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
end if.
end if.
compute ymat(1,im) = coeff(2,1).
do if (ws = 1).
compute ymat(1,im)=coeff(1,1).
end if.
do if (wm = 1).
compute ymat(2,im) = coeff(4,1).
do if (zm = 1).
compute ymat(3,im) = coeff(6,1).
do if (wzm = 1).
compute ymat(4, im) = coeff(8,1).
end if.
end if.
end if.
do if (model = 6).
compute mmpaths((im+1),1)=coeff((im+1),1).
end if.
end loop.
end if.
loop totlp = 1 to (1+(toteff*(bt = 1))).
do if (toteff = 1 and totlp =2).
compute xy =
{cons,datayed(:,(3+nmeds):ydatacol)}.
else.
compute xy={cons, datayed(:,3:ydatacol)}.
end if.
do if (dichy = 1).
compute pt2 =
make(nrow(datayed(:,2)),1,(csum(datayed(:,2))
/n)).
compute LL3 = datayed(:,2)*ln(pt2)+(1-
datayed(:,2))*ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xy),1,0).
compute LL1 = 0.
compute xy22=xy.

```

```

loop jjj = 1 to iterate.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.
compute coeff =
bt1+inv(t(xy22)*xy)*t(xy)*(datayed(:,2)-pt1).
compute pt1 = 1/(1+exp(-(xy*coeff))).
compute itprob = csum((pt1 <
.00000000000001) or (pt1 >
.99999999999999)).
do if (itprob = 0).
compute LL = datayed(:,2)*ln(pt1)+(1-
datayed(:,2))*ln(1-pt1).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
break.
end if.
compute bt1 = coeff.
compute LL1 = LL2.
end loop.
do if (jjj >= iterate and iterr = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
compute iterr = 1.
end if.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.
compute covmat = inv(t(xy22)*xy).
release xy22.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
end if.
do if (dichy = 0).
compute invXtX = inv(t(xy)*xy).
compute coeff =invXtX*t(xy)*datayed(:,2).

```

```

do if (effsize=1 and model > 3 and model < 7
and (ncovs > 0 or clsdmy > 0)).
compute xysdy =
{cons,datayed(:,(4+nmeds):ydatacol)}.
compute coeffsd
=inv(t(xysdy)*xysdy)*t(xysdy)*datayed(:,2).
compute resid=datayed(:,2)-(xysdy*coeffsd).
compute sdy cov =sqrt(csum(resid*&resid)/(n-
ncol(xysdy))).
compute xvaron=datayed(:,(3+nmeds)).
compute
coeffx2=inv(t(xysdy)*xysdy)*t(xysdy)*xvaron.
compute residx2=xvaron-(xysdy*coeffx2).
compute sdx cov
=sqrt(csum(residx2*&residx2)/(n-ncol(xysdy))).
end if.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4 and bt > 1).
compute resid=datayed(:,2)-xy*coeff.
compute sse = cssq(resid).
compute r2full = 1-(sse/sstot).
end if.
do if (bt =1).
compute resid=data(:,2)-xy*coeff.
compute k3 = nrow(coeff).
compute sse = cssq(resid).
compute mse = sse/(n-ncol(xy)).
compute
hcnot=coeff&/(sqrt(diag(mse*invXtX))).
do if (hc3 = 1).
compute h = xy(:,1).
loop i3=1 to n.
compute h(i3,1)= xy(i3,:)*invXtX*t(xy(i3,:)).
end loop.
loop i3=1 to k3.

```

```

compute xy(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xy(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xy(:,i3) = sqrt(mse)&*xy(:,i3).
end loop.
end if.
compute covmat = (invXtX*t(xy)*xy*invXtX).
end if.
end if.
do if (bt = 1).
do if (model = 2).
compute xy2={cons, datayed(:,3:ydatacol)}.
compute temp = ncol(xy2).
do if (temp > 6).
compute xy3=xy2(:,7:temp)}.
end if.
compute xy2={xy2(:,1:3), xy2(:,5)}.
do if (temp > 6).
compute xy2={xy2, xy3}.
release xy3.
end if.
compute invXtX = inv(t(xy2)*xy2).
compute coeff2 =invXtX*t(xy2)*datayed(:,2).
compute ssem2=cssq(datayed(:,2)-xy2*coeff2).
release xy2.
end if.
compute standerr = sqrt(diag(covmat)).
do if (totlp =1).
compute
mcmats((nmeds+1):ncol(mcmats),(nmeds+1):n
col(mcmats))=covmat(2:(1+nmeds),2:(1+nmed
s)).
end if.
compute standerr=standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1 and totp=2).
compute standerr(1,1)=stddevy/sqrt(n).

```

```

end if.
compute coeffplt = coeff.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*covmat*lmat)*((t(lmat)*coeff)))/dfnum).
compute coeff=coeff(1:(nrow(coeff)-clsdmy),1).
do if (totlp=1).
compute
mccoeff((nmeds+1):nrow(mccoeff))=coeff(2:(1+nmeds),1).
end if.
compute bbbb=coeff(2,1).
do if totp = 1.
compute deco(1,1)=2+nmeds.
compute deco = deco(1:decoc,1).
compute covdirt = make((nrow(covmat)-clsdmy),(ncol(covmat)-clsdmy),0).
compute covdirt = covmat(deco,:).
compute covdir =
make(nrow(covdirt),ncol(covdirt),0).
compute covdir = covdirt(:,t(deco)).
compute deco=coeff(deco,1).
do if (ttt > 0).
compute sedir =
sqrt(diag(directv*covdir*t(directv))).
compute directv = directv*deco.
end if.
compute sobel(:,3)=coeff(2:(1+nmeds),1).
compute sobel(:,4) = standerr(2:(1+nmeds),1).
compute sobel2 = sobel*sobel.
do if (varorder <> 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)*sobel2(:,4)+sobel2(:,3)*sobel2(:,2)).
end if.
do if (varorder = 2).

```



```

compute sobel(:,2) =
sqrt(sobel2(:,1)*sobel2(:,4)+sobel2(:,3)*sobel2(:,2)+sobel2(:,2)*sobel2(:,4)).
end if.
compute sobel(:,1) = sobel(:,1)*sobel(:,3).
compute sobel(:,3) = sobel(:,1)/sobel(:,2).
compute sobel(:,4) = 2*(1-
cdfnorm(abs(sobel(:,3)))).
end if.
do if (dichy = 0).
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
do if (ws= 1 and totlp=2).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute dfden=n-1.
end if.
compute cnms = {'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute op = {coeff, standerr, tratio, p}.
end if.
do if (dichy = 1).
compute tratio = (coeff&/standerr).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute wald = tratio*tratio.
compute cnms = {'coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute temp=coeff-abs(xp2)*standerr.
compute op = {coeff, standerr, tratio, p, temp}.
compute temp=coeff+abs(xp2)*standerr.
compute op = {op, temp}.
end if.
do if (detail = 1).
do if (totlp = 2).
print yname/title = '*****
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome: '/format = A8.
end if.
do if (totlp <> 2).

```

```

print yname/title =
*****
*****'/rlabels = 'Outcome:.'/format
= A8.
end if.
end if.
do if (dichy = 1 and bt = 1 and totlp = 1).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:.'/cnames = nmsd/format = F9.2.
end if.
do if (dichy = 0).
compute r2full = 1-(sse/ssty).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute jndf=dfden.
compute wsdf=dfden.
compute xd = abs(xp2).
compute jncrit = (dfden* (exp((dfden-
(5/6))*((xd/(dfden-
(2/3)+(.11/dfden))))*(xd/(dfden-
(2/3)+(.11/dfden)))))-1)).
compute summ = {sqrt(r2full), r2full, mse,
fratio, dfnum, dfden, pfr}.
compute temp1=coeff-sqrt(jncrit)*standerr.
compute temp2=coeff+sqrt(jncrit)*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (detail = 1).
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels =
'R', 'R-sq', 'MSE' 'F', 'df1', 'df2', 'p'/format =
F10.4.
end if.
end if.
end if.
do if (dichy = 1).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(coeff)-
1)).
compute mcF = LLdiff/LL3.

```

```

compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
do if (detail = 1).
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format =
F10.4.
end if.
end if.
do if (totlp = 2).
compute datanmy={constant';
datanmy((nmeds+2):nrow(datanmy),1)}.
end if.
do if (detail = 1).
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
do if (ws=1).
compute op2=op(1:(1+nmeds),:).
compute datanmy2 = {'c" path'; bpathnam(
(1+(nmeds>1)): ((1+(nmeds>1))+(nmeds-
1)),1)}.
do if (totlp=2).
compute datanmy2={'c path'}.
compute op2=op(1,:).
compute wsdf=n-1.
end if.
print op2/title = 'Model'/rnames =
datanmy2/cnames = cnms/format = F10.4.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
do if (ws <> 1).
print op/title = 'Model'/rnames =
datanmy/cnames = cnms/format = F10.4.
do if (covcoeff=1).
compute
covmattp=covmat(1:nrow(op),1:nrow(op)).
compute cnamestp=t(datanmy).

```

```

print covmattp/title='Covariance matrix of
regression parameter
estimates'/rnames=datanmy/cnames=cnamest
p/format= F10.4.
end if.
end if.
end if.
do if (ttt = 0 and totlp = 1).
compute deco = op((nmeds+2),:).
do if (ws = 1).
compute deco=op(1,:).
end if.
end if.
do if (ttt = 0 and totlp = 2).
compute decotot = op(2,:).
do if (ws = 1).
compute decotot=op(1,:).
end if.
end if.
do if (nmods > 0 and model > 4 and detail = 1
and nrow(yintkey) > 1)).
print yintkey/title = 'Product terms key:'/format
= A8.
end if.
do if (nmods > 0 and model < 4 and detail = 1).
print yintkey2/title = 'Product terms key:'/format
= A8.
do if ((model = 1 or model = 2) and dichy = 0).
compute temp={((op(4,3)**2)*(1-r2full))/dfden,
op(4,3)**2, 1, dfden, op(4,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(4,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(2,1).
do if (model = 2).
compute temp={temp;((op(6,3)**2)*(1-
r2full))/dfden, op(6,3)**2, 1, dfden, op(6,4)}.
do if (hc3=1).

```

```

compute temp(2,1)=((hcnot(6,1)**2)*(1-
r2full))/dfden.
end if.
compute frat2=(dfden*(r2full-(1-
(ssem2/ssty))))/(2*(1-r2full)).
do if (hc3=1).
compute lmat=make(nrow(coeffplt),2,0).
compute lmat(4,1)=1.
compute lmat(6,2)=1.
compute frat2 =
(t(t(lmat)*coeffplt)*inv(t(lmat)*covmat*lmat)*((t(l
mat)*coeffplt))/2).
end if.
compute temp={temp;(r2full-(1-
(ssem2/ssty))),frat2,2,dfden,1-
fcd(frat2,2,dfden)}.
compute rnms={rnms; yintkey2(3,1);'Both'}.
end if.
print temp/title = 'R-square increase due to
interaction(s):'/rnames=rnms/clabels = 'R2-
chng', 'F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (model = 3 and dichy = 0).
compute temp={((op(8,3)**2)*(1-r2full))/dfden,
op(8,3)**2, dfden, op(8,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(8,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(5,1).
print temp/title = 'R-square increase due to
three-way interaction:.'/rnames=rnms/clabels =
'R2-chng', 'F(1,df2)', 'df2', 'p'/format = F10.4.
end if.
end if.
end if.
do if (model = 6 and totlp = 1).
compute
mmpaths(nrow(mmpaths),1)=coeff(nrow(mmpa
ths),1).

```

```

compute
mmpaths(nrow(mmpaths),(2:(nmeds+1)))=t(co
eff(2:(nmeds+1),1)).
end if.
do if (totlp = 1).
loop im = 1 to nmeds.
do if (model < 4).
compute ymat(1,im) = coeff(3,1)*(1-yes).
compute ymat(2,im) = coeff(4,1).
compute cmat(1,im) = covmat(3,3).
compute cmat(2,im) = covmat(4,4).
compute cmat(5,im) = covmat(3,4).
compute jnb1=coeff(3,1).
compute jnb3=coeff(4,1).
compute jnsb1=covmat(3,3).
compute jnsb3=covmat(4,4).
compute jnsb1b3=covmat(3,4).
do if (model = 2 or model = 3).
compute ymat(3,im) = coeff(6,1).
compute cmat(3,im) = covmat(6,6).
compute cmat(6,im) = covmat(3,6).
compute cmat(8,im) = covmat(4,6).
end if.
do if (model = 3).
compute ymat(4,im) = coeff(8,1).
compute cmat(4,im) = covmat(8,8).
compute cmat(7,im) = covmat(3,8).
compute cmat(9,im) = covmat(4,8).
compute cmat(10,im) = covmat(6,8).
compute jnb1=coeff(4,1).
compute jnb3=coeff(8,1).
compute jnsb1=covmat(4,4).
compute jnsb3=covmat(8,8).
compute jnsb1b3=covmat(4,8).
end if.
end if.
do if (model > 3).
compute ymat(5,im) = coeff((1+im),1).
end if.
do if (xmy = 1).

```

```

compute ymat(6,im) = coeff((2+nmeds+im),1).
end if.
do if (vy = 1).
compute ymat(6,im) = coeff((3+nmeds+im),1).
end if.
do if (qy = 1 and vy = 1).
compute ymat(6,im) = coeff((5+nmeds+((im-1)*2)),1).
compute ymat(7,im) = coeff((6+nmeds+((im-1)*2)),1).
end if.
do if (vqy = 1).
compute ymat(6,im) = coeff((6+nmeds+((im-1)*3)),1).
compute ymat(7,im) = coeff((7+nmeds+((im-1)*3)),1).
compute ymat(8,im) = coeff((8+nmeds+((im-1)*3)),1).
end if.
do if (wmy = 1).
compute ymat(7,im) = coeff((3+nmeds+im-wy),1).
end if.
do if (wmy = 1 and vy = 1).
compute ymat(7,im) =coeff((4+(nmeds*2)+im-wy),1).
end if.
do if (wmy = 1 and vy = 1 and wvmy = 1).
compute ymat(7,im) =
coeff((6+(nmeds*2)+((im-1)*2)-wy),1).
compute ymat(8,im) =
coeff((7+(nmeds*2)+((im-1)*2)-wy),1).
end if.
do if (wmy = 1 and zmy = 1).
compute ymat(6,im) = coeff((6-(wzy*3)+(wzm-1)+(nmeds*2)+((im-1)*2)*wzm)+((im-1)*(1-wzm))-((zy-wzy)*2),1).
compute ymat(7,im) = coeff((3-zy+im+nmeds),1).
do if (wzmy = 1).

```

```

compute ymat(8,im) = coeff((7-
(wzy*3)+(nmeds*2)+((im-1)*2)),1).
end if.
end if.
do if (nmods > 0 and model <> 5).
loop indlp = 1 to nrow(modvals).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp)).
do if (model > 6).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp))*csum(ym
at(5:8,im)*vmat(5:8,indlp)).
end if.
end loop.
compute indboot((bt+(im-
1)*(boot+1)),:)=t(indeff).
do if (imm =1 or imm = 3).
compute
indbootp(bt,im)=ymat(2,im)*ymat(5,im).
do if (imm=3).
compute
indbootp(bt,(im+nmeds))=ymat(3,im)*ymat(5,i
m).
do if (zdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxz-cminz).
end if.
end if.
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
end if.
end if.
do if ((imm= 2 or imm = 4) or model = 74).
compute
indbootp(bt,im)=ymat(1,im)*ymat(6,im).
do if (imm=4).

```



```

compute
indbootp(bt,(im+nmeds))=ymat(1,im)*ymat(7,i
m).
do if (qdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxq-cminq).
end if.
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
end if.
end if.
do if (imm=7).
compute
indbootp(bt,im)=ymat(2,im)*ymat(6,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(2,im)*y
mat(6,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(6,im)*onesmv1)+(ymat(2,im)*y
mat(6,im)*t(immmv1)).
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).

```

```

compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1))))*(c
maxv-cminv).
end if.
end if.
do if (imm=5).
compute
indbootp(bt,im)=ymat(4,im)*ymat(5,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(4,im)*y
mat(5,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(3,im)*ymat(5,im)*onesmv1)+(ymat(4,im)*y
mat(5,im)*t(immmv1)).
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2))))*(c
maxw-cminw).
end if.
do if (zdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxz-
cminz).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1))))*(c
maxz-cminz).
end if.
end if.
do if (imm=6).

```

```

compute
indbootp(bt,im)=ymat(1,im)*ymat(8,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(1,im)*ymat(6,im)*onesmv2)+(ymat(1,im)*y
mat(8,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(7,im)*onesmv1)+(ymat(1,im)*y
mat(8,im)*t(immmv1)).
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxv-cminv).
end if.
do if (qdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxq-
cminq).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxq-cminq).
end if.
end if.
do if ((model = 58 or model = 59) and
wvdich=1).
compute
indbootp(bt,im)=(ymat(1,im)*ymat(7,im)*(cmax
w-cminw))+(ymat(2,im)*ymat(5,im)*(cmaxw-
cminw))+(ymat(2,im)*ymat(7,im)*((cmaxw*cma
xw)-(cminw*cminw))).
end if.
end if.

```

```

do if (model = 4 or model = 5).
compute
indboot(bt,im)=csum(yamat(1:4,im)*vmat(1:4,1
))*csum(yamat(5:8,im)*vmat(5:8,1)).
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.00000000000001.
end if.
compute sumind=rsum(indboot(bt,:)).
do if (im=nmeds).
compute
pmeff(bt,2:(im+1))=indboot(bt,:)/(sumind+coeff(
(2+nmeds),1)).
end if.
compute
rmeff(bt,(im+1))=indboot(bt,im)/coeff((2+nmeds
),1).
compute
abpseff(bt,(im+1))=indboot(bt,im)/stddevy.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*stddevx.
do if (ncovs > 0 or clsdmy > 0).
compute
abpseff(bt,(im+1))=indboot(bt,im)/sdy cov.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*sdx cov.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,1) = r2my-(r2full-r2xy).
compute abmmr = 1.
do if (indboot(bt,im) < 0).
compute abmmr = cmin(abmm).
end if.
do if (indboot(bt,im) > 0).
compute abmmr = cmax(abmm).
end if.
compute kappa2(bt,1)=indboot(bt,im)/abmmr.
compute tmp = indboot(bt,im)/abmmr.
end if.

```

```

end if.
end if.
do if (model = 6).
do if (nmeds = 2).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(4,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3).
compute
indboot(bt,3)=mmpaths(3,1)*mmpaths(4,3).
end if.
do if (nmeds = 3).
compute indboot(bt,1) =
mmpaths(2,1)*mmpaths(5,2).
compute indboot(bt,2) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3).
compute indboot(bt,3) =
mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4).
compute indboot(bt,4) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4).
compute indboot(bt,5) =
mmpaths(3,1)*mmpaths(5,3).
compute indboot(bt,6) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4).
compute indboot(bt,7) =
mmpaths(4,1)*mmpaths(5,4).
end if.
do if (nmeds = 4).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(6,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(6,3).
compute
indboot(bt,3)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(6,4).

```

```

compute
indboot(bt,4)=mmpaths(2,1)*mmpaths(5,2)*mmpaths(6,5).
compute
indboot(bt,5)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(6,4).
compute
indboot(bt,6)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3)*mmpaths(6,5).
compute
indboot(bt,7)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,8)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,9)=mmpaths(3,1)*mmpaths(6,3).
compute
indboot(bt,10)=mmpaths(3,1)*mmpaths(4,3)*mmpaths(6,4).
compute indboot(bt,11)
=mmpaths(3,1)*mmpaths(5,3)*mmpaths(6,5).
compute indboot(bt,12) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,13) =
mmpaths(4,1)*mmpaths(6,4).
compute indboot(bt,14) =
mmpaths(4,1)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,15) =
mmpaths(5,1)*mmpaths(6,5).
end if.
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.00000000000001.
end if.
compute
pmeff(bt,:)=indboot(bt,:)/(rsum(indboot(bt,:))+mmpaths(nrow(mmpaths),1)).

```

```

compute rmeff(bt,:) =
indboot(bt,:)/mmpaths(nrow(mmpaths),1).
compute abpseff(bt,:)=indboot(bt,:)/stddevy.
compute
abcseff(bt,:)=(stddevx*indboot(bt,:))/stddevy.
do if (ncovs > 0 or clsdmy > 0).
compute abpseff(bt,:)=indboot(bt,:)/sdy cov.
compute
abcseff(bt,:)=(sdxcov*indboot(bt,:))/sdy cov.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,:) = r2my-(r2full-r2xy).
end if.
end if.
end if.
end loop.
end if.
end loop.
do if (savboot = 1 and bt > 1).
compute bootstrp((bt-1),:)=bootcoef.
end if.
end loop.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
release datayed, dat, datamed, data, y, x,
datatm, dataty, mints2, cons, mnv, tmp.
do if (mc > 0).
compute x1 = sqrt(-
2*ln(uniform(mc,nrow(mcmats))))&*cos((2*3.14
159265358979)*uniform(mc,nrow(mcmats))).
compute x1=x1*chol(mcmats).
loop i=1 to nrow(x1).
compute x1(i,:)=x1(i,:)+t(mccoeff).
end loop.
loop i = 1 to nmeds.

```

```

compute
indboot(2:nrow(indboot),i)=x1(:,i)*x1(:,(i+nme
ds)).
end loop.
end if.
do if (ttt = 0 and model > 3).
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
end if.
do if (model <> 74).
do if (dichy = 0).
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 't', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 't', 'p', 'LLCI', 'ULCI'/format =
F10.4.
else.
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 'Z', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 'Z', 'p', 'LLCI', 'ULCI' /format =
F10.4.
end if.
end if.
end if.
do if (ttt > 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
compute clbs = {modvnm2, 'Effect', 'SE', 't', 'p',
'LLCI', 'ULCI'}.

```



```

compute tratio = (directv&/sedir).
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
compute outp = {modvalsd, directv, sedir,
tratio, p}.
do if (dichy = 0).
compute temp1=directv-sqrt(jncrit)*sedir.
compute temp2=directv+sqrt(jncrit)*sedir.
compute outp = {outp, temp1, temp2}.
end if.
do if (dichy = 1).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute temp = directv-abs(xp2)*sedir.
compute outp = {outp, temp}.
compute temp = directv+abs(xp2)*sedir.
compute outp = {outp, temp}.
compute clbs = {modvnm2, 'Effect', 'SE', 'Z', 'p',
'LLCI', 'ULCI'}.
end if.
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
print outp/title = 'Conditional direct effect(s) of
X on Y at values of the moderator(s):'/format =
F10.4 /cnames = clbs.
end if.
do if (nmods > 0 and model <> 5).
do if (model < 4).
print/title =
'*****'
'*****'.
compute zmat(1,1) = 1*(1-yes).
compute cfse = make(nrow(modvals),1,0).
loop #m = 1 to nrow(modvals).
do if (model = 1).
compute zmat(2,1)=modvals(#m,1)**2.
compute zmat(5,1)=2*modvals(#m,1).
end if.
do if (model = 2 or model = 3).
compute zmat(2,1)=modvals(#m,2)**2.
compute zmat(3,1)=modvals(#m,1)**2.

```

```

compute
zmat(4,1)=(modvals(#m,1)**2)*(modvals(#m,2)
**2)*(1-yes2).
compute zmat(5,1)=2*modvals(#m,2)*(1-yes).
compute zmat(6,1)=2*modvals(#m,1)*(1-yes).
compute
zmat(7,1)=2*modvals(#m,1)*modvals(#m,2)*(1-
yes).
compute
zmat(8,1)=2*modvals(#m,1)*modvals(#m,2).
compute
zmat(9,1)=2*modvals(#m,1)*(modvals(#m,2)**
2)*(1-yes2).
compute
zmat(10,1)=2*(modvals(#m,1)**2)*modvals(#m
,2)*(1-yes2).
end if.
compute cfse(#m,1)=sqrt(csum(zmat&*cmat)).
end loop.
end if.
do if (nmods > 0).
compute clbs = {modvnm, 'Effect'}.
loop im = 1 to nmeds.
compute obs = t(indboot(1+(im-1)*(boot+1),:)).
compute outp = {modvals, obs}.
do if (model < 4).
compute tstat = obs&/cfse.
do if (dichy = 0).
compute pval = 2*(1-tcdf(abs(tstat), (n-
ncol(xy)))).
compute temp=obs-sqrt(jncrit)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp=obs+sqrt(jncrit)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 't', 'p', 'LLCI', 'ULCI'}.
compute jnclbs=clbs.
end if.
do if (dichy = 1).
compute pval = 2*(1-cdfnorm(abs(tstat))).
compute temp = obs-abs(xp2)*cfse.

```

```

compute outp = {outp, cfse, tstat, pval, temp}.
compute temp = obs+abs(xp2)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute jnclbs=clbs.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indboot((1+(im-
1)*(boot+1)),:).
compute indboot2 = indboot((2+(im-
1)*(boot+1)):(1+(im-1)*(boot+1)+boot),:).
compute mnind = t(csum(indboot2)/boot).
compute stdind=t(sqrt((cssq(indboot2)-
((csum(indboot2)**2)/boot))/(boot-1))).
compute llci=make(1,ncol(indboot2),-999).
compute ulci=make(1,ncol(indboot2),-999).
loop #e = 1 to ncol(indboot2).

.
compute temp = indboot2(:,#e).
compute temp(GRADE( indboot2(:,#e) )) =
indboot2(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).

```

```

do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {modvals, obs, stdind, t(llci),
t(ulci)}.
compute clbs = {modvnm, 'Effect', 'Boot SE',
'BootLLCI', 'BootULCI'}.

```

```

end if.
compute mtemp = mnames(1, im).
compute rlbs =
make(nrow(modvals),1,mnames(1,im)).
do if (model < 4).
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
do if (yes=0).
print outp/title = 'Conditional effect of X on Y at
values of the moderator(s):'/cnames =
clbs/format = F10.4.
else.
compute outp=outp(:,3:ncol(outp)).
compute clbs=clbs(:,3:ncol(clbs)).
compute clbs(1,1)='Contrast'.
print outp/title = 'Contrast of conditional effects
of X on Y'/cnames = clbs/format = F10.4.
end if.
end if.
do if (model > 5 and mod74dic <> 1).
do if (im = 1).
print/title = 'Conditional indirect effect(s) of X on
Y at values of the moderator(s):'.
end if.
print outp/title = 'Mediator'/rnames =
rlbs/cnames = clbs/format = F10.4.
else if (mod74dic = 1 and model = 74).
do if (im = 1).
print/title = 'Indirect effect(s) of X on Y:'.
compute clbs3=clbs(1,2:ncol(clbs)).
end if.
compute outp3=outp(1,2:ncol(outp)).
print outp3/title = 'Mediator'/rnames =
rlbs/cnames = clbs3/format = F10.4.
end if.
end loop.
loop i = notes to 1 by -1.
do if (note(i,1) = 4 and yes=0).

```

```

print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 5 and yes=0).
print/title = 'Values for quantitative moderators
are the mean and plus/minus one SD from
mean.'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 14 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    below the mean was replaced
with the minimum because one SD below the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.
do if (note(i,1) = 15 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    above the mean was replaced
with the maximum because one SD above the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.
end loop.
do if (model = 3 and yes=0).
compute jnvals=make(nrow(matw),7,0).
compute jnvals(:,1)=matw.
compute jnvals(:,2)=jnb1+jnb3*jnvals(:,1).
compute
jnvals(:,3)=sqrt(jnsb1+2*jnvals(:,1)*jnsb1b3+(jn
vals(:,1)&*jnvals(:,1))*jnsb3).
compute jnvals(:,4)=jnvals(:,2)&/jnvals(:,3).

```

```

do if (dichy = 0).
compute jnvals(:,5)=2*(1-tcdf(abs(jnvals(:,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(:,5)=2*(1-
cdfnorm(abs(jnvals(:,4)))).
end if.
compute jnvals(:,6)=jnvals(:,2)-
sqrt(jncrit)*jnvals(:,3).
compute
jnvals(:,7)=jnvals(:,2)+sqrt(jncrit)*jnvals(:,3).
compute clbs={clbs(:,1),clbs(:,3:ncol(clbs))}.
do if (coeffci = 0).
compute jnvals =jnvals(:,1:(ncol(jnvals)-2)).
end if.
print jnvals/title = 'Conditional effect of X*M
interaction at values of W:'/cnames
=clbs/format = F10.4.
end if.
do if (jn = 1 and (model = 1 or model = 3) and
jndich = 0 and yes=0).
print/title = '***** JOHNSON-
NEYMAN TECHNIQUE
*****'.
compute ajn =(jncrit*jnsb3)-(jnb3*jnb3).
compute bjn = 2*((jncrit*jnsb1b3)-(jnb1*jnb3)).
compute cjn = (jncrit*jnsb1)-(jnb1*jnb1).
compute radarg = (bjn*bjn)-(4*ajn*cjn).
compute den = 2*ajn.
compute nrts = 0.
do if (radarg >= 0 and den <> 0).
compute x21 = (-bjn+sqrt(radarg))/den.
compute x22 = (-bjn-sqrt(radarg))/den.
compute roots = 0.
do if (x21 >= jnmin and x21 <= jnmax).
compute nrts = 1.
compute roots = {roots; x21}.
end if.
do if (x22 >= jnmin and x22 <= jnmax).

```

```

compute nrts = nrts + 1.
compute roots = {roots; x22}.
end if.
compute roots={roots,make(nrow(roots),2,0)}.
compute modtemp=m.
do if (model=3).
compute modtemp=w.
end if.
do if (nrts > 0).
compute roots = roots(2:nrow(roots),1:3).
compute roots(1,2)=(csum(modtemp <
roots(1,1))/n)*100.
compute roots(1,3)=(csum(modtemp >
roots(1,1))/n)*100.
do if (nrow(roots)=2).
compute roots(2,2)=(csum(modtemp <
roots(2,1))/n)*100.
compute roots(2,3)=(csum(modtemp >
roots(2,1))/n)*100.
end if.
print roots/title = 'Moderator value(s) defining
Johnson-Neyman significance
region(s)'/clabels = 'Value', '% below', '%
above'/format = F10.4.
compute jnvals=make((21+nrts),7,0).
loop i= 0 to 20.
compute jnvals((i+1),1)=jnmin+(i*((jnmax-
jnmin)/20)).
end loop.
loop i = 1 to nrts.
loop j = 2 to nrow(jnvals).
do if ((roots(i,1) > jnvals((j-1),1)) and (roots(i,1)
< jnvals(j,1))).
compute
jnvals((j+1):(21+i),1)=jnvals(j:(20+i),1).
compute jnvals(j,1)=roots(i,1).
end if.
end loop.
end loop.
loop i = 1 to nrow(jnvals).

```



```

compute jnvals(i,2)=jnb1+jnb3*jnvals(i,1).
compute
jnvals(i,3)=sqrt(jnsb1+2*jnvals(i,1)*jnsb1b3+(jn
vals(i,1)*jnvals(i,1))*jnsb3).
compute jnvals(i,4)=jnvals(i,2)/jnvals(i,3).
do if (dichy = 0).
compute jnvals(i,5)=2*(1-tcdf(abs(jnvals(i,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(i,5)=2*(1-
cdfnorm(abs(jnvals(i,4)))).
end if.
compute jnvals(i,6)=jnvals(i,2)-
sqrt(jncrit)*jnvals(i,3).
compute
jnvals(i,7)=jnvals(i,2)+sqrt(jncrit)*jnvals(i,3).
end loop.
do if (model = 1).
print jnvals/title = 'Conditional effect of X on Y
at values of the moderator (M)'/cnames
=jnclbs/format = F10.4.
end if.
do if (model = 3).
compute
jnclbs={jnclbs(:,1),jnclbs(:,3:ncol(jnclbs))}.
print jnvals/title = 'Conditional effect of X*M on
Y at values of the moderator (W)'/cnames
=jnclbs/format = F10.4.
end if.
end if.
do if (nrts = 0).
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
else.
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.

```

```

end if.
end if.
end if.
do if (model < 4 and (plot = 1)).
compute dataplot =
make((nrow(modvals)*nrow(matx)),(ncol(modv
als)+1),0).
compute tmp = 1.
loop i = 1 to nrow(modvals).
loop j = 1 to nrow(matx).
compute dataplot(tmp,:)= {matx(j,1),
modvals(i,:)}.
compute tmp=tmp+1.
end loop.
end loop.
compute dataplot = {dataplot,
make(nrow(dataplot),(1+dichy),0)}.
compute dataplo2 = make(nrow(dataplot),1,1).
do if (model = 1).
compute dataplo2 = {dataplo2, dataplot(:,2),
dataplot(:,1), (dataplot(:,1)*dataplot(:,2))}.
end if.
do if (model = 2 or model = 3).
compute dataplo2 = {dataplo2, dataplot(:,3),
dataplot(:,1), (dataplot(:,1)*dataplot(:,3)),
dataplot(:,2), (dataplot(:,1)*dataplot(:,2))}.
do if (model = 3).
compute dataplo2 = {dataplo2,
(dataplot(:,2)*dataplot(:,3)),
(dataplot(:,1)*dataplot(:,2)*dataplot(:,3))}.
end if.
end if.
loop i = 1 to nrow(dataplot).
compute tmp=dataplo2(i,:).
do if (ncovs > 0).
compute tmp = {tmp, covmeans}.
end if.
do if (cluster > 0).
compute tmp = {tmp, cldmeans}.
end if.

```

```

compute dataplot(i,(ncol(dataplot)-
(dichy)))=tmp*coeffplt.
do if (dichy = 1).
compute
dataplot(i,(ncol(dataplot)))=exp(tmp*coeffplt)/(1
+exp(tmp*coeffplt)).
end if.
end loop.
compute clbs = {xname, modvnm, 'yhat'}.
do if (dichy = 1).
compute clbs = {xname, modvnm, 'ln(odds)',
'prob'}.
end if.
print/title =
*****
*****!

```

```

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (model = 1).

```

```

do if (dichy = 0).
print/title = 'DATA LIST FREE/Comm_Pract
Eco_Innovation Financial_Performance.'
/space=1.
else if (dichy = 1).
print/title = 'DATA LIST FREE/Comm_Pract
Eco_Innovation logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
Financial_Performance BY Eco_Innovation.'
/space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Financial_Performance BY
Comm_Pract.' /space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
logodds BY Eco_Innovation.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
prob BY Eco_Innovation.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Comm_Pract.' /space=0.

```

```

print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Comm_Pract.' /space=0.
end if.
end if.
end if.
do if (model = 2 or model = 3).
do if (dichy = 0).
print/title = 'DATA LIST FREE/Comm_Pract
Eco_Innovation Financial_Performance.'
/space=1.
end if.
do if (dichy = 1).
print/title = 'DATA LIST FREE/Comm_Pract
Eco_Innovation logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
Financial_Performance BY
Eco_Innovation/PANEL ROWVAR=.' /space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Financial_Performance BY
Comm_Pract/PANEL ROWVAR=.' /space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
logodds BY Eco_Innovation/PANEL
ROWVAR=.' /space=0.

```

```

print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
prob BY Eco_Innovation/PANEL ROWVAR='
/space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Comm_Pract/PANEL
ROWVAR=' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Comm_Pract/PANEL
ROWVAR=' /space=0.
end if.
end if.
end if.
do if (ncovs > 0).
print/title = '* Estimates are based on setting
covariates to their sample means.'.
end if.
end if.
end if.
do if ((imm > 0) or (model = 74 and
mod74dic=0) or ((model = 58 or model = 59)
and wvdich=1))).
do if (imm > 4 and imm < 8).
compute
indbootp={indbootp(:,1:nmeds),indboop1,indbo
op2}.
end if.
compute obsprod = t(indbootp(1,:)).
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indbootp(1,:).
compute indbootp = indbootp(2:(boot+1),:).
compute mnindp = t(csum(indbootp)/boot).
compute stdindp=t(sqrt((cssq(indbootp)-
((csum(indbootp)**2)/boot))/(boot-1))).
compute llcip=make(1,ncol(indbootp),-999).

```

```

compute ulcip=make(1,ncol(indbootp),-999).
loop #e = 1 to ncol(indbootp).

.

compute temp = indbootp(:,#e).
compute temp(GRADE( indbootp(:,#e) )) =
indbootp(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).

```

```

end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcip(1,#e)=llcit.
compute ulcip(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {obsprod, stdindp, t(llcip),
t(ulcip)}.
compute clbs = {'Effect', 'SE(Boot)', 'BootLLCI',
'BootULCI'}.
do if (model = 8 or model = 12).
print/title='-----'.
compute outpimm=outp(1:nmeds,:).
print/title = 'Indirect effect of highest order
product:'/space=0.
print outpimm/title = 'Mediator'/cnames =
clbs/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).

```



```

print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'

end if.
compute clbs3 = {'Index', 'SE(Boot)',
'BootLLCI', 'BootULCI'}.
do if (imm < 5).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=outp((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'

compute outpimm=outp(1:nmeds,:).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
compute
outpimm=outp((nmeds+1):nrow(outp),:).
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).

```

```

compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),clbs3}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3(1,1)=modvnm(1,1).
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator:'/'format=A8.
print outpimm1/title =
'Mediator'/'cnames=clbs3/'rnames=cmmlbs/'form
at= F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/'space=0.
end if.
end if.
end if.
do if (boot = 0).
do if (model = 8 or model = 12).
print/title='-----'.
compute obsimm=obsprod(1:nmeds,:).
print/title = 'Indirect effect of highest order
product:'/'space=0.
print obsimm/title = 'Mediator'/'clabels =
'Effect'/'rnames = mnames/'format = F10.4.
end if.

```

```

do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'
.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'
.
end if.
do if ((imm > 0 and imm < 5) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator: '/format=A8.
end if.
compute outpimm=obsprod((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
end loop.
end if.
do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'
.
compute outpimm=obsprod(1:nmeds,1).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
compute
outpimm=obsprod((nmeds+1):nrow(obsprod),:)
.
compute outpimm={cmmval,outpimm}.

```

```

print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),'Index'}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3={modvnm(1,1),'Index'}.
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
end if.
end if.
end if.
compute conmake=0.
compute concols = 0.
do if ((model > 3 and model < 7) and (contrast
> 0) and nmods = 0 and nmeds > 1 ).
compute concols =
((ncol(indboot)*(ncol(indboot)-1))/2).
compute indcon=make(nrow(indboot),concols,-
999).
compute conkey = { ' ', ' ', ' ' }.
compute temp=1.
compute conmake=1.
loop i = 1 to (ncol(indboot)-1).

```

```

loop j = (i+1) to (ncol(indboot)).
compute indcon(:,temp)=indboot(:,i)-
indboot(:,j).
do if (contrast = 2).
compute indcon(:,temp)=abs(indboot(:,i))-
abs(indboot(:,j)).
end if.
do if (model <> 6).
compute conkey={conkey; mnames(1,i),
'minus', mnames(1,j)}.
end if.
do if (model = 6).
compute conkey={conkey; indlbl2(i,1), 'minus',
indlbl2(j,1)}.
end if.
compute temp=temp+1.
end loop.
end loop.
end if.
do if (model = 4 or model = 5).
compute clbs = {'Effect'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obs = {csum(obs); obs}.
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute rlbs = {rlbs;
cntname(1:ncol(indcon),1)}.
end if.
compute outp = obs.
compute outp2=outp.
do if (effsize = 1 and dichy = 0).
compute
pmeff(:,1)=rsum(pmeff(:,2:ncol(pmeff))).
compute
rmeff(:,1)=rsum(rmeff(:,2:ncol(rmeff))).
compute
abpseff(:,1)=rsum(abpseff(:,2:ncol(abpseff))).
compute
abcseff(:,1)=rsum(abcseff(:,2:ncol(abcseff))).

```

```

compute eff = {pmeff, rmeff, abpseff, abcseff}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute eff = {eff, r245}.
compute r245obs = {r245(1,1);r245(1,1)}.
compute kappa2ob = kappa2(1,1).
end if.
compute pmobs = t(pmeff(1,1:(nmeds+1))).
compute rmobs = t(rmeff(1,1:(nmeds+1))).
compute psobs = t(abpseff(1,1:(nmeds+1))).
compute csobs = t(abcseff(1,1:(nmeds+1))).
do if (contrast = 0).
compute outp2 = {obs, psobs, csobs, pmobs,
rmobs}.
end if.
do if (contrast > 0).
compute obs2=obs(1:nrow(psobs),:).
compute outp2 = {obs2, psobs, csobs, pmobs,
rmobs}.
end if.
compute clbs = {'ab', 'ab_ps', 'ab_cs', 'ab/c',
'ab/c'}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute outp2 = {outp2, r245obs}.
compute clbs = {clbs, 'R-sq_med'}.
do if (boot = 0 and mc = 0).
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
do if (boot = 0 and mc = 0).
do if (nmeds = 1).
compute outp2 = outp2(2,:).
compute rlbs = rlbs(2,1).
end if.
do if (model=5).
compute outp2=outp2(:,1:3).
end if.

```

```

print outp2/title = 'Indirect effect(s) of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1 and nmeds >
1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0 or mc > 0).
compute indboot = {rsum(indboot), indboot}.
compute bootSZ=boot.
do if (mc > 0).
compute bootSZ=mc.
end if.
compute ones = make(bootSZ,1,1).
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(bootSZ+1),:).
compute mnind = t(csum(indboot)/bootSZ).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/bootSZ))/(bootSZ-1))).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).
.
compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).

```

```

compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.

```



```

compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
do if (effsize = 1 and dichy = 0 and boot > 0).
compute estmte=eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).

.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2)))+1.

```

```

do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
end if.
end if.
do if (boot > 0 or mc > 0).
compute outp = {obs, stdind, t(llci), t(ulci)}.
do if (nmeds = 1).
compute outp = outp(2,:).
compute rlbs = rlbs(2,1).
end if.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.

```

```

do if (mc > 0).
compute clbs = {'Effect', 'MC SE', 'MC LLCI',
'MC ULCI'}.
end if.
print outp/title = 'Indirect effect of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (dichy = 0 and effsize = 1 and boot > 0).
compute outp = {psobs,
stdindf((2*(nmeds+1)+1):(3*(nmeds+1)),1),
t(llcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))),
t(ulcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Partially standardized indirect
effect of X on Y'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute outp = {csobs,
stdindf((3*(nmeds+1)+1):(4*(nmeds+1)),1),
t(llcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))),
t(ulcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Completely standardized
indirect effect of X on Y'/rnames = rlbs/cnames
= clbs/format = F10.4.
compute outp = {pmobs,
stdindf(1:(nmeds+1),1),t(llcif(1,1:(nmeds+1))),
t(ulcif(1,1:(nmeds+1))))}.
do if (model = 4).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
compute outp = {rmobs, stdindf(((nmeds+1)+1)
:(2*(nmeds+1)),1),

```

```

t(llcif(1,((nmeds+1)+1):(2*(nmeds+1))))),
t(ulcif(1,((nmeds+1)+1):(2*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
do if (nmeds = 1 and cluster = 0 and ncovs = 0
and model = 4).
compute r245obs = r245obs(1,1).
compute outp = {r245obs,
stdindf((4*(nmeds+1)+1):(4*(nmeds+1)+1),1),
t(llcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1))),
t(ulcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1)))}.
print outp/title = 'R-squared mediation effect
size (R-sq_med)'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
end if.
do if (normal = 1).
do if (nmeds = 1).
print sobel/title = 'Normal theory tests for
indirect effect'/clabels = 'Effect', 'se', 'Z',
'p'/format = F10.4.
end if.
do if (nmeds > 1).
compute rlbs2 = rlbs(2:nrow(rlbs),1).
print sobel/title = 'Normal theory tests for
specific indirect effects'/rnames = rlbs2/clabels
= 'Effect', 'se', 'Z', 'p'/format = F10.4.
end if.
end if.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).

```

```

print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
do if (model = 6).
compute clbs = {'eff'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obsfsum=csum(obs).
compute obs = {obsfsum; obs}.
compute indlbl = indlbl(1:nrow(obs),1).
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute indlbl = {indlbl;
cntname(1:ncol(indcon),1)}.
end if.
compute obs2=obs.
do if (boot = 0).
do if (dichy = 0 and effsize = 1).
compute clbs = {'eff', 'eff_ps', 'eff_cs', 'eff/c',
'eff/c'}.
do if (ncovs = 0).
compute obs = {obs, obs/stddevy,
(obs*stddevx/stddevy), obs/ctot,
obs/mmpaths(nrow(mmpaths),1)}.
end if.
do if (ncovs > 0 or clsdmy > 0).
compute obs = {obs, obs/sdycov,
(obs*sdxcov/sdycov),
obs/(obsfsum+mmpaths(nrow(mmpaths),1)),
obs/mmpaths(nrow(mmpaths),1)}.
end if.
compute obs2=obs.
do if (contrast > 0).
compute obs2 = obs(1:(nrow(obs)-concols),:).
end if.
end if.

```

```

print obs2/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute indboot = {rsum(indboot), indboot}.
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(boot+1),:).
compute mnind = t(csum(indboot)/boot).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/boot))/(boot-1))).
compute temp = nrow(indboot).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).
.
compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).

```

```

compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llici(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.

```

```

end if.
end loop.
compute obs = {obs, stdind, t(llci), t(ulci)}.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
print obs/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (effsize = 1 and dichy = 0).
compute indboot=indboot(:,1:(ncol(indboot)-
concols)).
compute eff = {rsum(abpseff), abpseff,
rsum(abcseff), abcseff, rsum(pmeff), pmeff,
rsum(rmeff), rmeff}.
compute effobs = eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).

.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(effobs(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).

```



```

do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( ( effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute temp2 = stdindf(1:ncol(indboot),1).
compute temp3 = effobs(:,1:ncol(indboot)).
compute templow = llcif(1,1:ncol(indboot)).
compute temphi = ulcif(1,1:ncol(indboot)).

```

```

compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Partially standardized indirect
effect of X on Y'/cnames = clbs/rnames =
indlbl/format = F10.4.
compute temp2 =
stdindf((ncol(indboot)+1):(2*ncol(indboot)),1).
compute temp3 =
effobs(:,(ncol(indboot)+1):(2*ncol(indboot))).
compute templow =
llcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute temphi =
ulcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Completely standardized
indirect effect of X on Y'/cnames = clbs/rnames
= indlbl/format = F10.4.
compute temp2 =
stdindf((2*(ncol(indboot))+1):(3*ncol(indboot)),1
).
compute temp3 =
effobs(:,(2*(ncol(indboot))+1):(3*ncol(indboot)))
.
compute templow =
llcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute temphi =
ulcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
compute temp =
eff(:,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temp2 =
stdindf((3*(ncol(indboot))+1):(4*ncol(indboot)),1
).

```

```

compute temp3 =
effobs(:,(3*(ncol(indboot))+1):(4*ncol(indboot)))
.
compute templow =
llcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temphi =
ulcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
end if.
end if.
do if (nmeds = 2).
compute effkey = {xname, '->', mnames(1,1), '->', yname, ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, ' ', ' '}.
compute effkey = {indlbl(2:4,1), effkey}.
end if.
do if (nmeds = 3).
compute effkey = {xname, '->', mnames(1,1), '->', yname, ' ', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname, ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,3), '->', yname, ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, ' ', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', mnames(1,3), '->', yname, ' ', ' '}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, '', '', '', ''}.
compute effkey = {indl(2:8,1), effkey}.
end if.
do if (nmeds = 4).
compute effkey = {xname, '->' , mnames(1,1), '->', yname, '', '', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->', yname, '
', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->', yname, '
', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,4), '->', yname, '
', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname, '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,4), '->', yname, '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', mnames(1,4), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, '', '', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,4), '->', yname, '
', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, '', ''}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, '', '', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', mnames(1,4), '->', yname, '
', '', '', ''}.
compute effkey = {effkey; xname, '->' ,
mnames(1,4), '->', yname, '', '', '', '', ''}.
compute effkey = {indlbl(2:16,1), effkey}.
end if.
print effkey/title = 'Indirect effect key'/format =
A8.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
end if.
DO IF (medte2=1).
compute
name=t(mdlnms2(2:nrow(mdlnms2),1)).
compute tmpdat=datayed(:,2:ncol(datayed)).
compute dd=tmpdat(:,1).
compute
dd={dd,tmpdat(:,(2+nmeds)),tmpdat(:,2:(1+nmeds
ds)))}.
compute stratify=0.
do if (ncovs > 0).
compute cname = cnames.
compute
dd={dd,tmpdat(:,(3+nmeds):(2+nmeds+ncovs))
}.
end if.
compute method = 1.
compute samples = boot.
do if (mc > 0).
compute samples = mc.
compute method=2.

```

```

end if.
compute omnibus = 1.
compute booterr = 0.
do if (mcx < 0 or mcx > 4).
compute mcx = 1.
end if.
compute ovals = ncol(design(dd(:,1))).
compute nte = notes.
compute criterr = 0.
compute xskip = 0.
compute dumok = 0.
compute badend = 0.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute priorlo = -9999999.
compute priorhi = 9999999.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute reps = samples.
do if (contrast = 1).
compute note(nte,1) = 26.
compute nte=nte+1.
end if.
do if (effsize = 1).
compute note(nte,1) = 27.
compute nte=nte+1.
end if.
do if (normal = 1).
compute note(nte,1) = 28.
compute nte=nte+1.
end if.
do if (method = 2).
do if (omnibus = 1).
compute note(nte,1) = 25.
compute nte=nte+1.
end if.
compute samples = 0.

```

```

do if (reps < 1000).
compute reps = 1000.
end if.
end if.
do if (method = 1).
compute samples = reps.
end if.
compute temp = ncol(dd).
compute n = nrow(dd).
compute ones = make(n,1,1).
compute temp=dd(:,2).
compute nxd = ncol(temp).
compute nx = nxd.
do if (nx = 1 and mcx > 0).
compute temp = dd.
compute temp(GRADE(dd(:,2)),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,2)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute mnvls = cmin(t(nnvls)).
do if (mnvls < 2).
compute errs = errs + 1.
compute runerrs(errs,1) = 37.
compute criterr = 1.
end if.
do if (nvls > 9).
compute errs = errs+1.
compute runerrs(errs,1) = 36.
compute criterr = 1.
end if.
do if (criterr = 0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,2).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,2) <> nnvls((temp-1),1)).
compute nnvls(temp,1)=dd(i,2).
compute temp = temp+1.

```

```

end if.
end loop.
do if (mcx > 0).
compute x = dummy(:,2:ncol(dummy)).
compute nx = ncol(x).
compute minus1 = make(1,ncol(x),-1).
compute xdes=make((nx+1),3,0).
compute xdes(1,1)=dd(1,2).
compute xdes(1,2)=1.
compute temp = 2.
loop k = 2 to n.
do if (dd(k,2) <> dd((k-1),2)).
compute xdes(temp,2) = k.
compute xdes(temp,1) = dd(k,2).
compute xdes((temp-1),3) = k-1.
compute temp=temp+1.
end if.
end loop.
compute xdes((temp-1),3)=n.
compute xdes = {xdes, (xdes(:,3)-xdes(:,2)+1)}.
do if (stratify <> 0).
compute note(nte,1) = 7.
compute nte=nte+1.
end if.
do if (mcx = 4).
loop k = 1 to n.
do if (rsum(x(k,:)) = 0).
compute x(k,:) = minus1.
end if.
end loop.
end if.
do if (mcx = 2 or mcx = 3).
compute note(nte,1)=4.
compute nte = nte+1.
loop k = 1 to n.
do if (rsum(x(k,:)) > 0).
loop i = 1 to ncol(x).
do if (x(k,i) = 0).
compute x(k,i) = 1.
else.

```



```

break.
end if.
end loop.
end if.
end loop.
do if (mcx = 3).
compute conmat1={-8,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1; 0,0,0,0,0,-
3,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.
loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute x(k,:)=conmat1((rsum(x(k,:))+1),:).
end loop.
end if.
end if.
compute xdata = x.
compute xskip = 1.
compute xname = ddd(1,1:nx).
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcx = 4).
compute dummat(1,:) = minus1.
end if.
do if (mcx = 2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcx = 3).
compute dummat=conmat1.
end if.

```

```

compute dummat={nnvls, dummat}.
compute xname2={name(1,2), xname}.
end if.
end if.
end if.
compute nm=nmeds.
compute r2b = make(nm,1,-999).
compute indeff = make((nx*nm),1,-999).
compute cov =
make((nx*nm+nm),(nx*nm+nm),0).
compute mns=make(nrow(cov),1,0).
compute ny = ncol(dd)-nm-nxd-ncovs.
compute y = dd(:,1).
compute ydata = y.
compute yname = name(1,1).
do if (xskip = 0).
compute x = dd(:,2:(1+nxd)).
compute xdata = x.
compute xname = name(1,2:(1+nxd)).
compute indlbs = t(xname).
end if.
compute m = dd(:,(2+nxd):(1+nxd+nm)).
compute mdata = m.
compute mlab = { ' M1 = ' ; ' M2 = ' ; ' M3 = ' ; '
M4 = ' ; ' M5 = ' ; ' M6 = ' ; ' M7 = ' ; ' M8 = ' ; '
M9 = ' ; ' M10 = ' }.
compute mlab = {mlab ; ' M11 = ' ; ' M12 = ' ; '
M13 = ' ; ' M14 = ' ; ' M15 = ' }.
compute mname =
name(1,(2+nxd):(1+nxd+nm)).
do if (ncovs > 0).
compute c =
dd(:,(2+nxd+nm):(1+nxd+nm+ncovs)).
compute x = {x, c}.
compute xdata = x.
compute cdata = c.
compute xname = {xname, cname}.
end if.
compute nms={yname; t(mname);
t(xname(1,1:(ncol(xname)-ncovs)))}.

```

```

do if (dumok = 1).
compute nms={yname; t(mname); name(1,2)}.
end if.
do if (nm < 16).
do if (dumok = 1).
print dummat/title = 'Coding of categorical X
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
compute
boots=make((samples+1),((nx+1)*nm),-999).
do if (errs = 0 and criterr = 0).
loop k = 1 to (samples+1).
do if (k = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef=0.
do if (k > 1).
compute u=trunc(uniform(n,1)*n)+1.
compute xdata = x(u,:).
compute mdata = m(u,:).
compute ydata = y(u,:).
do if (ncovs > 0).
compute cdata = c(u,:).
end if.
end if.
loop i = 1 to nm.
compute start = 1+(i-1)*nx.
compute xmat = {ones, xdata}.
do if (covmy=2).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = mdata(:,i).
compute amat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*amat.

```

```

compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute r2cha = r2.
compute adjr2cha = adjr2.
do if (ncovs > 0 and covmy <> 2).
compute xmat = {ones, cdata}.
compute atmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*atmat)**2).
compute r2step1 = (sstotal-ssresid2)/sstotal.
compute r2cha = r2-r2step1.
compute adjr2cha = adjr2- (1-((1-r2step1)*(n-
1)/(n-ncol(xmat)))).
compute xmat = {ones, xdata}.
end if.
compute bootcoef={bootcoef,t(amat)}.
compute r2b(i,1)=adjr2cha.
do if (k = 1).
.
compute hcinvXtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtXtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.

```

```

end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute tratio = amat&/sqrt(diag(covmat)).
compute rnms = mname(1,i).
print rnms/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.
compute lmat = ident(nrow(amat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*amat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*amat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/'format = F10.4
/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'.
compute op = {amat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/'clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/'rnames = rnms/'format =
F10.4.
compute cov(start:(start+(nx-
1)),start:(start+(nx-

```

```

1)))=covmat(2:(ncol(covmat)-(ncovs*(covmy <>
2))),2:(ncol(covmat)-(ncovs*(covmy <> 2)))).
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
=F10.4.
end if.
end if.
compute mns(start:(start+(nx-
1)),1)=amat(2:(1+nx)).
end loop.
compute xmat = {ones,mdata,xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (k = 1).
print yname/title =
*****
*****'/rlabels = 'Outcome:'/format
=A8.
do if (ovals = 2).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:'/cnames = nmsd/format = F9.2.
end if.
end if.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(ymat)/n)).
compute LL3 = ymat*&ln(pt2)+(1-ymat)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.
.
compute pt1lp= pt1.
compute xlp= xmat.

```

```

compute ylp= ymat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp&*ln(pt1lp)+(1-ylp)&*ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.

```

```

compute bootcoef={bootcoef,t(bmat)}.
compute LL1 = LL2.
compute covmat=varb.
end if.
do if (ovals <> 2).
compute bmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*bmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute bootcoef={bootcoef,t(bmat)}.
end if.
do if (k = 1).
do if (ovals = 2).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).
compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)*sqrt(diag(covmat)).
compute outp = {outp, temp}.

```



```

compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms2/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute
opdirsum=outp((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
end if.
do if (ovals <> 2).

.
compute hcinvtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(bmat)).
compute lmat = lmat(:,2:ncol(lmat)).

```

```

compute fratio =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/(ncol(xmat)-1).
compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat((2+nmeds):(2+nmeds+(nx-1)),:)
= lmat2.
compute fratiodr =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/nx.
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute df2=(n-ncol(xmat)).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), df2, pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = bmat&/sqrt(diag(covmat)).
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute op = {bmat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat)))) }.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*(xp2/(temp-
(2/3)+(.11/temp)))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms2/format =
F10.4.
compute
opdirsum=op((2+nmeds):(1+nmeds+nx),:).

```

```

compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
compute xmat = {ones,mdata}.
do if (ncovs > 0).
compute xmat = {xmat, cdata}.
end if.
compute btmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*btmat)**2).
compute r2direct = r2-((sstotal-
ssresid2)/sstotal).
compute pfr = 1-fcdf(fratiodr,(nx),(n-1-nx-nm-
ncovs)).
compute r2direct = {r2direct, fratiodr, nx, (n-1-
nx-nm-ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms2/cnames=rnms2/form
at= F10.4.
end if.
end if.
do if (k = 1 and toteff = 1).
print yname/title = '*****
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome: '/format = A8.
compute xmat = {ones, xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(ymat)/n)).
compute LL3 = ymat*ln(pt2)+(1-ymat)*ln(1-
pt2).
compute LL3 = -2*csum(LL3).

```

```

compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.

compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= ymat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)*&ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.

```

```

end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute LL1 = LL2.
compute covmat=varb.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(bmat)-
1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms = {'Constant'; t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).
compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute optotsum=outp(2:(1+nx),:).
end if.
do if (ovals <> 2).

```

```

compute cmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*cmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).

.

compute hcinvtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvtX*t( xmat )* xmat
*hcinvtX
.
compute lmat = ident(nrow(cmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).

```

```

compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = cmat&/sqrt(diag(covmat)).
compute op = {cmat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute optotsum=op(2:(1+nx),:).
compute ssresid2 = ssresid.
compute r2total = r2.
compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat(2:(1+nx),:) = lmat2.
compute fratio3 =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/nx.
do if (ncovs > 0).
compute xmat = {ones, cdata}.
compute ctmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*ctmat)**2).
compute r2total = r2-((sstotal-ssresid2)/sstotal).
end if.

```

```

compute pfr = 1-fcdf(fratio3,(nx),(n-1-nx-
ncovs)).
compute r2tot2 = {r2total, fratio3, nx, (n-1-nx-
ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns((nrow(mns)-
nm+1):nrow(mns),1)=bmat(2:(1+nm),1).
compute tmp = 1.
loop i = 1 to (nm*nx).
compute
boots(k,i)=mns(i,1)*mns(((nm*nx)+tmp),1).
do if ((i/nx)=trunc(i/nx)).
compute
boots(k,((nm*nx)+tmp))=r2b(tmp,1)*mns(((nm*
nx)+tmp),1).
compute tmp = tmp+1.
end if.
end loop.
do if (savboot = 1 and k > 1).
compute bootstrp((k-1),:)=bootcoef.
end if.
end loop.
do if (method = 1).
compute llci=make(1,ncol(boots),-999).
compute ulci=make(1,ncol(boots),-999).
compute indeff = t(boots(1,1:(nm*nx))).
compute indeff22=t(boots(1,:)).
compute omni = t(boots(1,((nm*nx)+1) :
ncol(boots))).
do if (nrow(boots) > 1).
compute boots = boots(2:nrow(boots),:).
compute bootse = t(sqrt((((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).

```



```

loop i = 1 to ncol(boots).

.

compute temp = boots(:,i).
compute temp(GRADE( boots(:,i) )) = boots(:,i).
compute badlo = 0.
compute badhi = 0.
do if ( (indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(indeff22(i,1)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.

```

```

do if ( (indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,i)=llcit.
compute ulci(1,i)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute llci = t(llci).
compute ulci= t(ulci).
compute llci={llci,ulci}.
end if.
end if.
do if (method = 2).
compute tmp=1.
loop i = 1 to (nm*nx).
compute
indeff(i,1)=mns(i,1)*mns(((nm*nx)+tmp),1).
do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = sqrt(-
2*ln(uniform(reps,nrow(cov))))&*cos((2*3.1415
9265358979)*uniform(reps,nrow(cov))).
compute boots=boots*chol(cov).
loop i = 1 to ncol(boots).
compute boots(:,i)=boots(:,i)+mns(i,1).
end loop.
compute tmp=1.
loop i = 1 to (nm*nx).

```

```

compute
boots(:,i)=boots(:,i)*boots(:,((nm*nx)+tmp)).
compute temp = boots(:,i).
compute temp(GRADE(boots(:,i))) = boots(:,i).
compute boots(:,i) = temp.
do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = boots(:,1:(nm*nx)).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
compute llci =
t(((boots(cilow,:));(boots(cihigh,:)))).
end if.
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
compute rnmssum=rnms(2:(1+nx),1).
do if (ovals <> 2).
print optotsum/title = 'Relative total effects of X
of Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.
print r2tot2/title = 'Omnibus test of total effect of
X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print optotsum/title = 'Relative total effects of X
on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
end if.
compute
rnmssum=rnms2((2+nmeds):(1+nmeds+nx),1).
do if (ovals <> 2).

```

```

print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.
print r2direct/title = 'Omnibus test of direct
effect of X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/'space=0.
do if (omnibus = 1 and (method = 1 or method
= 3)).
compute indlbs = {indlbs; 'Omnibus'}.
end if.
loop i = 1 to nm.
compute op = mname(1,i).
print op/title = 'Relative indirect effect(s) of X on
Y through:.'/format = A8/space=0.
do if (method = 2).
compute clbs = {'Effect', 'SE MC', 'MC LLCI',
'MC ULCI'}.
compute op = {indeff, bootse(1:(nm*nx)), llci}.
end if.
do if (method = 1).
compute clbs = {'Effect', 'SE(boot)', 'LLCI',
'ULCI'}.
compute op = indeff.
do if (samples > 0).
compute op = {indeff, bootse(1:(nm*nx),:),
llci(1:(nm*nx),:)};.
end if.
end if.
compute temp = op((((i-1)*nx)+1):(nx*i),:).
do if ((method = 1) and omnibus = 1).
do if (samples = 0).
compute temp={temp;omni(i,1)}.
end if.

```

```

do if (samples > 0).
compute temp = {temp; omni(i,1),
bootse(((nm*nx)+i,:), llci(((nm*nx)+i,:),:))}.
end if.
end if.
print temp/title = ' '/rnames = indlbs/cnames =
clbs/format = F10.4 /space=0.
print/title = '-----'.
end loop.
end if.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
END IF.
DO IF (mdpbe=1).
compute nm={yname,xname,mnames}.
do if (ncovs > 0).
compute
nm={yname,cnames,xname,mnames}.
end if.
compute tpx=xname.
compute x = dd(:,2:ncol(dd)).
compute tpy = yname.
compute tempy= dd(:,1).
compute n = nrow(dd).
compute nx=1.
compute criterr=0.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute ddd1 = {'int_1', 'int_2', 'int_3', 'int_4',
'int_5', 'int_6', 'int_7', 'int_8', 'int_9'}.

```

```

compute dumok = 0.
compute mcfoc=mcx.
compute mcmmod=mcm.
compute mcloc=(mcfoc>0).
compute con = make(n,1,1).
compute ncovs=ncol(x)-2.
do if (mcfoc > 0 or mcmmod > 0).
compute temp = dd.
compute temp(GRADE(dd(:,(ncol(dd)-
mcloc))),:)= dd.
compute dd = temp.
compute dummy = design(dd(:,ncol(dd)-
mcloc)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute toosmall=rsum(nnvls < 2).
compute mnvls = cmin(t(nnvls)).
do if ((rsum(nnvls < 2)) > 0).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=37.
end if.
do if (nvls > 10).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=36.
end if.
do if (criterr=0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,ncol(dd)-mcloc).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,ncol(dd)-mcloc) <> nnvls((temp-
1),1)).
compute nnvls(temp,1)=dd(i,ncol(dd)-mcloc).
compute temp = temp+1.
end if.
end loop.
compute dummy = dummy(:,2:ncol(dummy)).

```

```

do if (mcfoc=4 or mcmmod=4).
compute minus1=make(1,ncol(dummy),-1).
loop k = 1 to n.
do if (rsum(dummy(k,:)) = 0).
compute dummy(k,:)=minus1.
end if.
end loop.
end if.
do if (mcfoc = 2 or mcfoc = 3) or (mcmmod = 2 or
mcmmod = 3)).
loop k = 1 to n.
do if (rsum(dummy(k,:)) > 0).
loop i = 1 to ncol(dummy).
do if (dummy(k,i) = 0).
compute dummy(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcfoc = 3 or mcmmod=3).
compute conmat1={-8,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1; 0,0,0,0,0,-
3,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.
loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute
dummy(k,:)=conmat1((rsum(dummy(k,:))+1),:).
end loop.
end if.
end if.
compute nx = ncol(dummy).
compute xname = ddd(1,1:nx).
compute xdata = dummy.

```

```

compute xname = ddd(1,1:nx).
compute xname2={nm(1,(ncol(nm)-mcloc)),
xname}.
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcfoc = 2 or mcmmod=2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcfoc = 3 or mcmmod=3).
compute dummat=conmat1.
end if.
do if (mcfoc = 4 or mcmmod = 4).
compute dummat(1,:)=minus1.
end if.
compute dummat={nnvls, dummat}.
end if.
end if.
do if (criterr=0).
compute y = dd(:,1).
compute ovals = ncol(design(dd(:,1))).
compute itprob = 0.
do if (ovals = 2).
compute omx = cmax(y(:,1)).
compute omn = cmin(y(:,1)).
compute y(:,1) = (y(:,1) = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute sstotal = csum((y-(csum(y)/n))&**2).
compute outv = t(nm(1,1)).
compute mdtr = nm(1,ncol(dd)).
compute fciv = nm(1,(ncol(dd)-1)).
do if (mcfoc <> 0 or mcmmod <> 0).
compute inter=make(n,nx,0).
loop i = 1 to nx.

```



```

compute inter(:,i)=dummy(:,i)*dd(:,ncol(dd)-
(1-mcloc)).
end loop.
do if (ncovs = 0).
compute x={con,dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y)))}.
end if.
do if (ncovs > 0).
compute
x={con,dd(:,2:(1+ncovs)),dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={t(nm(1,2:(1+ncovs)));
nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y)))}.
end if.
end if.
compute xmns = csum(x)/n.
compute focvals = {1,0,0}.
compute highwarn = 0.
compute lowwarn = 0.
do if (ncol(x) > (2*nx+2)).
compute covmns = {1, xmns(1,2:(ncol(x)-1-
(2*nx)))}.
compute focvals = {covmns,0,0}.
end if.
compute dfres = n-ncol(x).
do if (ovals = 2).
compute pt2 = make(n,1,(csum(y)/n)).
compute LL3 = y*&ln(pt2)+(1-y)*&ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(x),1,0).
compute LL1 = 0.

```

```

.
compute pt1lp= pt1.
compute xlp= x.
compute ylp= y.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)*&ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.

```

```

end if.
compute tval=xp2*xp2.
else if (ovals <> 2).
compute invXtX=inv(t(x)*x).
compute b = invXtX*t(x)*y.
compute resid = y-(x*b).
compute ssresid=csum(resid&*resid).
compute msresid=ssresid/dfres.
compute varb = msresid*invXtX.
compute
seint=sqrt(varb(nrow(varb),nrow(varb))).
compute k3 = nrow(b).
do if (hc3 = 1).
compute xhc=x.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)= xhc(i3,:)*invXtX*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).
end loop.
compute varb=(invXtX*t(xhc)*xhc*invXtX).
end if.
compute seb = sqrt(diag(varb)).
compute r2 = 1-(ssresid/sstotal).
do if (mcfoc <> 0 or mcmmod <> 0).
compute ytp=y.
compute xtp=x(:,1:(ncol(x)-nx)).
compute btp = inv(t(xtp)*xtp)*t(xtp)*ytp.
compute residt = ytp-(xtp*btp).
compute ssresidt=csum(residt&*residt).
compute r2noint = 1-(ssresidt/sstotal).
end if.
compute pr = ncol(x)-1.
compute lmat = ident(nrow(b)).
compute lmat = lmat(:,2:ncol(lmat)).
compute f =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
pr).

```

```

compute pf = 1-fcdf(f,pr,dfres).
compute pf = {sqrt(r2),r2,msresid,f,pr,dfres,pf}.
compute xd=abs(xp2).
end if.
do if (itprob = 0).
compute tstat = b&/seb.
do if (ovals <> 2).
compute p = 2*(1-tcdf(abs(tstat), dfres)).
else if (ovals = 2).
compute p = 2*(1-cdfnorm(abs(tstat))).
end if.
compute outp = {b,seb,tstat,p}.
compute nms = {'constant'; nms; 'interact'}.
do if (ovals <> 2).
compute tval = (dfres* (exp((dfres-
(5/6))*((xd/(dfres-(2/3)+(.11/dfres)))*(xd/(dfres-
(2/3)+(.11/dfres)))))-1)).
compute outp={outp,(b-
sqrt(tval)&*seb),(b+sqrt(tval)&*seb)}.
end if.
compute bb = tval.
do if (dumok = 1).
do if (mcfoc > 0).
print dummat/title = 'Coding of categorical X
variable for analysis:.'/cnames = xname2/format
= F5.2.
else if (mcmod > 0).
print dummat/title = 'Coding of categorical M
variable for analysis:.'/cnames = xname2/format
= F5.2.
end if.
end if.
print yname/title =
*****
*****'/rlabels = 'Outcome:.'/format
= A8.
do if (ovals = 2).
compute nmsd = {outv, 'Analysis'}.
print rcd/title = 'Coding of binary Y for
analysis:.'/cnames = nmsd/format = F10.4.

```

```

compute outp = {outp,(outp(:,1)-
sqrt(tval)&*outp(:,2)),(outp(:,1)+sqrt(tval)&*outp
(:,2))}.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
print outp/title = 'Logistic Regression
Model'/rnames = nms/clabels 'Coeff' 'se' 'Z' 'p'
'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
compute varbtmp=varb.
compute btmp=b.
compute LL2f=LL2.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=x(:,1:(ncol(x)-nx)).
compute bt1 = make(ncol(xlp),1,0).

.

compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).

```

```

compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&*ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL2f.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(LLdiff,nx).
compute rcha={LLdiff,nx,pchi}.
end if.
do if (ovals <> 2).
print pf/title = 'Model Summary'/clabels = 'R' 'R-
sq' 'MSE' 'F' 'df1' 'df2' 'p'/format F10.4.
compute lmat=make(nrow(b),1,0).

```

```

compute lmat(nrow(lmat),1)=1.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
1).
compute
rcha=((b(nrow(b),1)/seint)*(b(nrow(b),1)/seint))*
(1-r2)/dfres.
compute rcha = {rcha, fcha, 1, dfres,
outp((pr+1),4)}.
print outp/title = 'Model'/rnames = nms/clabels
'coeff' 'se' 't' 'p' 'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute rcha=r2-r2noint.
compute lmat=make((nrow(b)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
nx).
compute pvalr2c=1-fcdf(fcha,nx,dfres).
compute rcha = {rcha, fcha, nx, dfres,pvalr2c}.
end if.
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute intkey = {'a', 'b', 'c', 'd', 'e'}.
loop i = 1 to nx.
compute intkey={intkey; ddd1(1,i), ' : ', ddd1(1,i),
' X ', nm(1,ncol(dd)-(1-mcloc))}.
end loop.
compute intkey=intkey(2:nrow(intkey),:).
print intkey/title='Product terms key:.'/format =
a8.
end if.
do if (ovals <> 2).

```

```

print rcha/title = 'R-square increase due to
interaction:.'/clabels 'R2-chng' 'F' 'df1' 'df2'
'p'/format F10.4.
else if (ovals = 2).
print rcha/title = 'Likelihood ratio test for
interaction:.'/clabels 'Chi-sq' 'df' 'p'/format F10.4.
end if.
print/title =
*****
*****!
compute mdvar = x(:,(ncol(x)-1)).
do if (mcfoc = 0 and mcmod = 0).
compute g1 = b((ncol(x)-2),1).
compute g3 = b(ncol(x),1).
compute vg1 = varb((ncol(x)-2),(ncol(x)-2)).
compute vg3 = varb(ncol(x),ncol(x)).
compute covg1g3 = varb((ncol(x)-2), ncol(x)).
end if.
do if (mcfoc > 0 or mcmod > 0).
compute mdvar = x(:,(ncol(x)-(2*nx))).
end if.
compute mdmin = cmin(mdvar).
compute mdmax = cmax(mdvar).
compute fvar = x(:,(ncol(x)-2)).
compute nval = ncol(design(mdvar)).
compute fvmin = cmin(fvar).
compute fvmax = cmax(fvar).
do if (mmodval = 999).
compute mnmd = csum(mdvar)/n.
compute tmp = make(n,1,mnmd).
compute sdmd = sqrt(csum(((mdvar-
tmp)**2))/(n-1)).
compute probeval = {mnmd-sdmd; mnmd;
mnmd+sdmd}.
do if (probeval(1,1) < mdmin).
compute lowwarn = 1.
compute probeval(1,1)=mdmin.
end if.
do if (probeval(nrow(probeval),1) > mdmax).
compute highwarn = 1.

```



```

compute probeval(nrow(probeval),1)=mdmax.
end if.
do if (quantile = 1).
compute tmp = mdvar.
compute tmp(GRADE(mdvar(:,1)),:)= mdvar.
compute mdvar = tmp.
compute
probeval={mdvar(trunc(0.1*n),1);mdvar(trunc(0
.25*n),1);mdvar(trunc(0.5*n),1);mdvar(trunc(0.7
5*n),1);mdvar(trunc(0.9*n),1)}.
end if.
end if.
do if (nval = 2).
compute probeval = make(2,1,0).
compute probeval(1,1)=cmin(mdvar).
loop i = 1 to n.
do if (mdvar(i,1) <> probeval(1,1)).
compute probeval(2,1) = mdvar(i,1).
BREAK.
end if.
end loop.
end if.
do if (mmodval <> 999).
compute probeval = mmodval.
end if.
compute outp = make(nrow(probeval),7,0).
do if (mcfoc > 0 or mcmmod > 0).
compute focvals=make(1,ncol(x)+1,1).
do if (mcfoc > 0).
print/title = 'Conditional effect of X on Y at
values of the moderator:'.
print/title = ' '/space=0.
compute rnn2=mdtr.
compute matt=make(nx,6,0).
end if.
loop jj=1 to nrow(probeval).
do if (mcfoc > 0).
loop ii=1 to nx.
compute g1=b((ncol(x)-(2*nx)+ii),1).
compute g3=b((ncol(x)-nx+ii),1).

```

```

compute vg1=varb((ncol(x)-(2*nx)+ii),(ncol(x)-
(2*nx)+ii)).
compute vg3=varb((ncol(x)-nx+ii),(ncol(x)-
nx+ii)).
compute covg1g3=varb((ncol(x)-
(2*nx)+ii),(ncol(x)-nx+ii)).
compute x2 = probeval(jj,1).
compute w1 = g1+g3*x2.
compute varw1 =
vg1+(2*x2*covg1g3)+((x2*x2)*vg3).
compute sew1 = sqrt(varw1).
compute t1 = w1/sew1.
do if (ovals <> 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-tcdf(abs(t1), dfres)).
compute cnms = {'Coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)={w1,sew1,t1,p,llci,ulci}.
else if (ovals = 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-cdfnorm(abs(t1))).
compute cnms = {'Coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)={w1,sew1,t1,p,llci,ulci}.
end if.
end loop.
compute rnms=t(xname).
compute mdvalpr=probeval(jj,1).
print mdvalpr/title = 'Moderator
value:.'/rnames=rnn2/format= F10.4 /space=0.
print matt/title='
'/cnames=cnms/rnames=rnms/format= F10.4
/space=0.
compute xprob=x(:,(ncol(x)-(2*nx)))-mdvalpr.
loop kk = 1 to nx.
compute xprob={xprob,
(xprob(:,1)&*x(:,(ncol(x)-(2*nx)+kk)))}.
end loop.

```

```

compute xprob={x(:,1:((ncol(x)-(2*nx)-
1)),xprob}.
do if (ovals <> 2).
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute residc = y-(xprob*bmultc).
compute ssresidc=csum(residc*residc).
compute r2c = r2-(1-(ssresidc/sstotal)).
compute fcha2=(dfres*r2c)/(nx*(1-r2)).
do if (hc3 = 1).
loop kk = 1 to nx.
compute xprob={xprob, x(:,(ncol(x)-
(2*nx)+kk))}.
end loop.
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute k3 = nrow(bmultc).
compute xhc=xprob.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)=
xhc(i3,)*inv(t(xprob)*xprob)*t(xhc(i3,)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).
end loop.
compute
varbc=(inv(t(xprob)*xprob)*t(xhc)*xhc*inv(t(xpr
ob)*xprob)).
compute lmat=make((nrow(bmultc)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha2 =
(t(t(lmat)*bmultc)*inv(t(lmat)*varbc*lmat)*((t(lm
at)*bmultc)))/nx).
end if.
compute pvalr2cc=1-fcdf(fcha2,nx,dfres).
compute rcha2 = {r2c, fcha2, nx,
dfres,pvalr2cc}.

```

```

print rcha2/title = 'Test of equality of conditional
means at this value of the moderator'/clabels
'R2-chng' 'F' 'df1' 'df2' 'p'/format F10.4.
else if (ovals = 2).
compute btmp=b.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=xprob.
compute bt1 = make(ncol(xlp),1,0).

.

compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*ln(pt1lp)+(1-ylp)*ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).

```

```

compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL4.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(LLdiff,nx).
compute rcha2 = {LLdiff, nx,pchi}.
print rcha2/title = 'Test of equality of log odds
conditioned on this moderator value'/clabels
'Chi-sq' 'df' 'p'/format F10.4.
end if.
end if.
compute
tttt=make(nrow(dummat),1,probeval(jj,1)).
compute tttt={tttt,dummat(:,2:ncol(dummat))}.
loop kkk=1 to nx.
compute tttt={tttt,tttt(:,1)*tttt(:,1+kkk)}.
end loop.
compute ones=make(nrow(dummat),1,1).
do if (ncol(x) > (2*nx+2)).
compute
covmnmat=make(nrow(tttt),ncol(covmns),0).
loop kkk=1 to nrow(tttt).
compute covmnmat(kkk,:)=covmns.
end loop.
compute tttt={covmnmat,tttt}.
else.
compute tttt={ones,tttt}.
end if.
compute focvals={focvals;dummat(:,1),tttt}.
do if (mcfoc > 0).
compute yhat={dummat(:,1),(tttt*b)}.
compute cmnms={fciv, 'yhat'}.
do if (ovals <> 2).

```

```

print yhat/title = 'Estimated conditional means
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (ovals = 2).
print yhat/title = 'Estimated conditional log odds
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (jj <> nrow(probeval)).
print/title='-----'/space=0.
end if.
end if.
end loop.
compute focvals=focvals(2:nrow(focvals),:).
compute yhat=focvals(:,2:ncol(focvals))*b.
compute
focvals={focvals(:,1),focvals(:,(ncol(focvals)-
(2*nx))),yhat}.
compute cnms={fciv,mdtr,'yhat'}.
do if (mcmod > 0).
compute cnms={mdtr,fciv,'yhat'}.
end if.
end if.
do if (mcmod > 0).
compute outp=make((nx+1),7,0).
compute bcatm={b((ncol(x)-
(2*nx)),1);b((ncol(x)-nx+1):ncol(x),1)}.
compute outp(:,1)=dummat(:,1).
compute bcatcov=varb((ncol(x)-
nx):ncol(x),(ncol(x)-nx):ncol(x)).
compute bcatcov(1,1)=varb((ncol(x)-
(2*nx)),(ncol(x)-(2*nx))).
compute
bcatcov(2:nrow(bcatcov),1)=varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx)).
compute
bcatcov(1,2:nrow(bcatcov))=t(varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx))).
loop i = 1 to nrow(dummat).

```

```

compute
catmval={1,dummat(i,2:ncol(dummat))}.
compute condeff=catmval*bcatm.
compute
condse=sqrt(catmval*bcatcov*t(catmval)).
compute outp(i,2:3)={condeff,condse}.
end loop.
compute outp(:,4)=outp(:,2)&/outp(:,3).
compute outp(:,5) = 2*(1-tcdf(abs(outp(:,4)),
dfres)).
compute outp(:,6) = (outp(:,2)-
sqrt(tval)&*outp(:,3)).
compute outp(:,7) =
(outp(:,2)+sqrt(tval)&*outp(:,3)).
do if (ovals <> 2).
compute cnmms = {xname2(1,1), 'coeff', 'se',
't', 'p', 'LLCI', 'ULCI'}.
end if.
do if (ovals = 2).
compute outp(:,5) = 2*(1-
cdfnorm(abs(outp(:,4)))).
compute cnmms = {xname2(1,1), 'coeff', 'se',
'Z', 'p', 'LLCI', 'ULCI'}.
end if.
print outp/title = 'Conditional Effect of Focal
Predictor in Groups Defined by the Moderator
Variable: '/cnames=cnmms/format = F10.4.
end if.
do if (nval > 2 and (mmodval = 999) and
mcmmod = 0).
do if (quantile <> 1).
print/title = 'Moderator values are the sample
mean and plus/minus one SD from mean'.
else.
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
end if.
do if (highwarn = 1 and quantile = 0).

```

```

print/title = 'NOTE: For the moderator values
above, one SD above the mean was'.
print/title = 'replaced with the maximum
because one SD above the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
do if (lowwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD below the mean was'.
print/title = 'replaced with the minimum
because one SD below the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
end if.
do if (nval = 2 and (mmodval = 999)).
print/title = 'The moderator variable is
dichotomous'.
end if.
do if (mmodval <> 999 and mcmmod = 0 and
((mmodval < mdmin) or (mmodval > mdmax))).
print/title = 'Warning: Moderator value specified
is outside of the range of the data'.
end if.
do if (plot = 1).
do if (mcfoc = 0 and mcmmod=0).
compute cnms = {t(nms((ncol(x)-2):(ncol(x)-
1),1)), 'yhat'}.
end if.
do if (ovals = 2).
compute prob =
exp(focvals(:,3))/(1+exp(focvals(:,3))).
compute focvals = {focvals, prob}.
compute cnms = {cnms, 'prob'}.
end if.
print/title =
*****
*****!

```



```
print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (ovals = 2).
do if (mcfoc=0).
print/title = 'DATA LIST FREE/Eco_Innovation
Comm_Pract logodds prob.' /space=1.
end if.
do if (mcmod=0).
print/title = 'DATA LIST FREE/Comm_Pract
Eco_Innovation logodds prob.' /space=1.
end if.
else if (ovals <> 2).
do if (mcfoc=0).
print/title = 'DATA LIST FREE/Eco_Innovation
Comm_Pract Financial_Performance.'
/space=1.
end if.
do if (mcmod=0).
print/title = 'DATA LIST FREE/Comm_Pract
Eco_Innovation Financial_Performance.'
/space=1.
end if.
end if.
print/title = 'BEGIN DATA.' /space=0.
print focvals/title = ' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
```

```

do if (ovals <> 2 and mcmmod=0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH Financial_Performance BY
Comm_Pract.' /space=0.
end if.
do if (ovals <> 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
Financial_Performance BY Eco_Innovation.'
/space=0.
end if.
do if (ovals = 2 and mcmmod=0).
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH logodds BY Comm_Pract.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Eco_Innovation
WITH prob BY Comm_Pract.' /space=0.
end if.
do if (ovals = 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
logodds BY Eco_Innovation.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
prob BY Eco_Innovation.' /space=0.
end if.
do if (ncovs > 0).
print/title = 'NOTE: For data above, covariates
are set to their sample means.'.
end if.
end if.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.

```

```

compute runerrs(errs,1) = 22.
end if.
end if.
END IF.
end if.
do if (bad > 0).
compute note(notes,1) = 9.
compute notes = notes + 1.
end if.
print/title = '***** ANALYSIS
NOTES AND WARNINGS
*****'.

loop i = 1 to errs.
do if (runerrs(i,1) = 1).
print/title = 'ERROR: One of your declared
mediators is dichotomous. This procedure
can"t be used.'.
end if.
do if (runerrs(i,1) = 2).
print/title = 'ERROR: For model 6, this
procedure limits the number of mediators to
four.'.
end if.
do if (runerrs(i,1) = 3).
print/title = 'ERROR: For models 1, 2, and 3,
only a single variable can be listed in the M
list.'.
end if.
do if (runerrs(i,1) = 4).
print/title = 'ERROR: You requested a model
involving W but did not provide a valid W
variable name.'.
end if.
do if (runerrs(i,1) = 5).
print/title = 'ERROR: You requested a model
involving Z but did not provide a valid Z
variable name.'.
end if.
do if (runerrs(i,1) = 6).

```

```
print/title = 'ERROR: You requested a model
involving Q but did not provide a valid Q
variable name.'.
end if.
do if (runerrs(i,1) = 7).
print/title = 'ERROR: You requested a model
involving V but did not provide a valid V
variable name.'.
end if.
do if (runerrs(i,1) = 8).
print/title = 'ERROR: You specified a W
variable for a model that does not need it'.
end if.
do if (runerrs(i,1) = 9).
print/title = 'ERROR: You specified a Z variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 10).
print/title = 'ERROR: You specified a Q variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 11).
print/title = 'ERROR: You specified a V variable
for a model that does not need it.'.
end if.
do if (runerrs(i,1) = 12).
print/title = 'ERROR: The variable specified for
W has already been assigned.'.
end if.
do if (runerrs(i,1) = 13).
print/title = 'ERROR: The variable specified for
Z has already been assigned.'.
end if.
do if (runerrs(i,1) = 14).
print/title = 'ERROR: The variable specified for
Q has already been assigned.'.
end if.
do if (runerrs(i,1) = 15).
print/title = 'ERROR: The variable specified for
V has already been assigned.'.
```

```
end if.  
do if (runerrs(i,1) = 16).  
print/title = 'ERROR: You did not provide a  
valid Y variable name.'  
end if.  
do if (runerrs(i,1) = 17).  
print/title = 'ERROR: The variable specified for  
Y has already been assigned.'  
end if.  
do if (runerrs(i,1) = 18).  
print/title = 'ERROR: Model 6 requires more  
than one mediator.'  
end if.  
do if (runerrs(i,1) = 19).  
print/title = 'ERROR: You have not specified a  
valid model number.'  
end if.  
do if (runerrs(i,1) = 20).  
print/title = 'ERROR: At least one and only one  
variable must be listed for X.'  
end if.  
do if (runerrs(i,1) = 21).  
print/title = 'ERROR: At least one and only one  
variable must be listed for Y.'  
end if.  
do if (runerrs(i,1) = 22).  
print/title = 'ERROR: Iteration didn't converge  
to a solution. Interpret results with caution.'  
end if.  
do if (runerrs(i,1) = 23).  
print/title = 'ERROR: You specified a clustering  
variable that does not exist in your variable  
list.'  
end if.  
do if (runerrs(i,1) = 24).  
print/title = 'ERROR: You specified a clustering  
variable that has already been assigned.'  
end if.  
do if (runerrs(i,1) = 25).
```

```

print/title = 'ERROR: One or more of your M
variables is not listed in the variables list.'.
end if.
do if (runerrs(i,1) = 26).
print/title = 'ERROR: A maximum of 20 cluster
units is allowed. Use multilevel modeling
instead.'.
end if.
do if (runerrs(i,1) = 27).
print/title = 'ERROR: One of the variables in
your model is a constant.'.
end if.
do if (runerrs(i,1) = 28).
print/title = 'ERROR: Dichotomous Y is not
permitted with WS option.'.
end if.
do if (runerrs(i,1) = 29).
print/title = 'ERROR: Insufficient number of
variables in vars= list when using WS option.'.
end if.
do if (runerrs(i,1) = 30).
print/title = 'ERROR: Too many variables in
vars= list when using WS option. Covariates
not allowed.'.
end if.
do if (runerrs(i,1) = 31).
print/title = 'ERROR: mmodval and wmodval
can"t both be set to zero with contrast option.'.
end if.
do if (runerrs(i,1) = 32).
print/title = 'ERROR: You did not provide a
valid X variable name.'.
end if.
do if (runerrs(i,1) = 33).
print/title = 'ERROR: PROCESS requires all
variable names to be eight characters or
fewer.'.
print/title = '      Please shorten variable names
and reexecute.'/space=0.
end if.

```

```

do if (runerrs(i,1)=34).
print/title = 'ERROR: X and M cannot both be
specified as multicategorical.'.
end if.
do if (runerrs(i,1)=35).
print/title = 'ERROR: Multicategorical variable
specification available only for models 1 and
4.'.
end if.
do if (runerrs(i,1)=36).
print/title = 'ERROR: Categorical variables
cannot have more than 10 categories.'.
end if.
do if (runerrs(i,1)=37).
print/title = 'ERROR: Each group must have at
least two cases.'.
end if.
do if (runerrs(i,1)=38).
print/title = 'ERROR: Cluster option not
available with multicategorical variables.'.
end if.
do if (runerrs(i,1)=39).
print/title = 'ERROR: A maximum of 10
variables is allowed in the M= list.'.
end if.
do if (runerrs(i,1) = 41).
print/title = 'ERROR: MCM option not permitted
in model 4.'.
end if.
end loop.
do if (errs = 0).
do if (boot > 1 or mc > 0).
do if (bconoff = 1 and boot > 1).
print boot/title = 'Number of bootstrap samples
for bias corrected bootstrap confidence
intervals:'//format = F8.0.
end if.
do if (bconoff = 0 and boot > 1).

```

```

print boot/title = 'Number of bootstrap samples
for percentile bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (mc > 1).
print mc/title = 'Number of samples for Monte
Carlo confidence intervals:'/format = F8.0.
end if.
do if (booterr = 1).
compute badend = badend(1,2:ncol(badend)).
print badend/title = 'WARNING: Bootstrap CI
endpoints below not trustworthy. Decrease
confidence or increase bootstraps'/format =
F10.4.
end if.
end if.
print conf/title = 'Level of confidence for all
confidence intervals in output:'/format = F8.2.
do if ((center = 1 or ws=1) and (ncol(centvar) >
1)).
compute centvar = centvar(1,2:ncol(centvar)).
print centvar/title = 'NOTE: The following
variables were mean centered prior to
analysis:'/format = a8.
end if.
loop i = 1 to notes.
do if (note(i,1) = 1).
print/title = 'NOTE: Confidence level restricted
to between 50 and 99.9999%. 95% confidence
is provided in output'.
end if.
do if (note(i,1) = 2).
print/title = 'NOTE: Effect size measures for
indirect effects are not available for models
with dichotomous outcomes'.
end if.
do if (note(i,1) = 3).
print/title = 'NOTE: All standard errors for
continuous outcome models are based on the
HC3 estimator'.

```



```

end if.
do if (note(i,1) = 6).
print/title = 'NOTE: The number of bootstrap
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 7).
print/title = 'NOTE: The Johnson-Neyman
method is available only for Models 1 and 3'.
end if.
do if (note(i,1) = 8).
print/title = 'NOTE: The Johnson-Neyman
method cannot be used with a dichotomous
moderator'.
end if.
do if (note(i,1)=9).
print bad/title = 'NOTE: Some bootstrap
samples had to be replaced. The number of
such replacements was:'.
end if.
do if (note(i,1) = 11).
print nmiss/title = 'NOTE: Some cases were
deleted due to missing data. The number of
such cases was:'.
end if.
do if (note(i,1) = 12).
print /title = 'NOTE: Monte Carlo method
available only for models 4 and 5.
Bootstrapping was used instead.'.
end if.
do if (note(i,1) = 13).
print/title = 'NOTE: The number of Monte Carlo
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 19).
print/title = 'NOTE: Effect sizes not available for
within-subject analyses.'.
end if.
do if (note(i,1) = 16).

```

```
print/title = 'NOTE: Normal theory tests not
available for within-subject analyses.'.
end if.
do if (note(i,1) = 17).
print/title = 'NOTE: Monte Carlo confidence
intervals not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 18 and warnrep=0).
print/title = 'WARNING: You have requested
OLS estimation with a dichotomous criterion.'.
print/title = 'Interpret model coefficients and
inferential statistics with caution.'/space=0.
compute warnrep=1.
end if.
do if (note(i,1) = 20).
print/title = 'NOTE: Saving of bootstrap
estimates not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 22).
print/title = 'NOTE: Effect size option with
covariates requires covariates in models of M
and Y.'.
end if.
do if (note(i,1) = 23).
print/title = 'NOTE: Johnson-Neyman method
not available with multicategorical X'.
end if.
do if (note(i,1) = 24).
print/title = 'NOTE: MMODVAL option not
available with a multicategorical moderator'.
end if.
do if (note(i,1) = 25).
print/title = 'NOTE: Omnibus test for relative
indirect effects is not available with the MC
option.'.
end if.
do if (note(i,1) = 26).
```

| | | |
|-----------|----------------|---|
| | | <pre> print/title = 'NOTE: CONTRAST option not available with multicategorical X.'. end if. do if (note(i,1) = 27). print/title = 'NOTE: EFFSIZE option not available with multicategorical X.'. end if. do if (note(i,1) = 28). print/title = 'NOTE: NORMAL option not available with multicategorical X.'. end if. do if (note(i,1) = 31). print/title = 'NOTE: The TOTAL option is not available when Y is dichotomous.'. end if. do if (note(i,1) = 32). print/title = 'NOTE: Kappa-squared is disabled from output as of version 2.16.'. end if. do if (note(i,1) = 33). print/title = 'NOTE: COVMY option ignored in models 1, 2, and 3.'. end if. end loop. end if. end matrix. </pre> |
| Resources | Processor Time | 00:00:04.59 |
| | Elapsed Time | 00:00:04.70 |

Run MATRIX procedure:

***** PROCESS Procedure for SPSS Release 2.16.3 *****

Written by Andrew F. Hayes, Ph.D. www.afhayes.com
Documentation available in Hayes (2013). www.guilford.com/p/hayes3

Model = 4
Y = Financia
X = Comm_Pra
M = Eco_Inno

Sample size

Outcome: Eco_Inno

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|-----------|--------|----------|-------|
| .8312 | .6909 | .3566 | 1908.4213 | 1.0000 | 854.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|-------|-------|---------|-------|-------|-------|
| constant | .5914 | .0753 | 7.8534 | .0000 | .4436 | .7392 |
| Comm_Pra | .8310 | .0190 | 43.6855 | .0000 | .7937 | .8683 |

Outcome: Financia

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|-----------|--------|----------|-------|
| .9167 | .8404 | .1935 | 2246.2625 | 2.0000 | 853.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|-------|-------|---------|-------|--------|-------|
| constant | .0494 | .0574 | .8605 | .3897 | -.0633 | .1622 |
| Eco_Inno | .5016 | .0252 | 19.8979 | .0000 | .4521 | .5511 |
| Comm_Pra | .4801 | .0252 | 19.0487 | .0000 | .4306 | .5296 |

***** DIRECT AND INDIRECT EFFECTS *****

Direct effect of X on Y

| Effect | SE | t | p | LLCI | ULCI |
|--------|-------|---------|-------|-------|-------|
| .4801 | .0252 | 19.0487 | .0000 | .4306 | .5296 |

Indirect effect of X on Y

| | Effect | Boot SE | BootLLCI | BootULCI |
|----------|--------|---------|----------|----------|
| Eco_Inno | .4168 | .0254 | .3667 | .4673 |

Normal theory tests for indirect effect

| Effect | se | Z | p |
|--------|-------|---------|-------|
| .4168 | .0230 | 18.1041 | .0000 |

***** ANALYSIS NOTES AND WARNINGS *****

Number of bootstrap samples for bias corrected bootstrap confidence intervals:

5000

Level of confidence for all confidence intervals in output:

95.00

----- END MATRIX -----

restore.

Matrix

| Notes | | |
|----------------|--------------------------------|----------------------|
| Output Created | | 20-OCT-2020 16:02:52 |
| Comments | | |
| Input | Active Dataset | DataSet1 |
| | Filter | <none> |
| | Weight | <none> |
| | Split File | <none> |
| | N of Rows in Working Data File | 856 |

Syntax

```
matrix.  
get dat/file = */variables =  
Operational_Performance Pro_Env_Strgy  
Technological_Advan /names =  
vnames/missing = 9999.  
compute ninit = nrow(dat).  
get dat/file = */variables =  
Operational_Performance Pro_Env_Strgy  
Technological_Advan /names =  
vnames/missing = omit.  
get tmp/file = */variables =  
Operational_Performance /names =  
yname/missing = omit.  
get tmp2/file = */variables = Pro_Env_Strgy  
/names = xname/missing = omit.  
get tmp/file = */variables =  
Technological_Advan /names =  
mnames/missing = omit.  
get tmp/file = */variables = w999999t z999999t  
v999999t q999999t.  
compute wname=tmp(1,1).  
do if (wname = ' ').  
compute wname = 'xxx'.  
end if.  
compute zname=tmp(1,2).  
do if (zname = ' ').  
compute zname = 'xxx'.  
end if.  
compute vname=tmp(1,3).  
do if (vname = ' ').  
compute vname = 'xxx'.  
end if.  
compute qname=tmp(1,4).  
do if (qname = ' ').  
compute qname = 'xxx'.  
end if.  
compute n = nrow(dat).  
compute p0=-.322232431088.  
compute p1 = -1.  
compute p2 = -.342242088547.
```

```

compute p3 = -.0204231210245.
compute p4 = -.0000453642210148.
compute q0 = .0993484626060.
compute q1 = .588581570495.
compute q2 = .531103462366.
compute q3 = .103537752850.
compute q4 = .0038560700634.
compute badend = 0.
compute priorlo = -9999999.
compute priorhi = 9999999.
compute criterr = 0.
compute cluster = 0.
compute clsdmy = 0.
compute jndich = 0.
compute wvdich=0.
compute zqdich=0.
compute mod74dic=0.
compute warnrep=0.
compute contrast = trunc( 0 ).
do if (contrast > 2 or contrast < 0).
compute contrast = 0.
end if.
compute booterr = 0.
compute effsize = ( 0 = 1).
compute note = make(10,1,0).
compute notes = 1.
compute iterr = 0.
compute clsmth = 0.
compute quantile = ( 0 = 1).
compute mmodval= 999.
compute jn = ( 1 = 1).
compute covcoeff=( 0 =1).
compute center = ( 0 = 1).
compute detail = ( 1 = 1).
compute coeffci = ( 1 = 1).
compute conf = 95.
compute longname = ( 1 =1).
compute bconoff=( 0 <> 1).
compute covmy = trunc( 0 ).
do if (covmy < 0 or covmy > 2).

```

```

compute covmy = 0.
end if.
do if (trunc(conf) >= 100 or (trunc(conf) <=
50)).
compute conf = 95.
compute note(notes,1) = 1.
compute notes = notes + 1.
end if.
do if (n < ninit).
compute nmiss = ninit-n.
compute note(notes,1) = 11.
compute notes = notes + 1.
end if.
compute errs = 0.
compute quantd = {0,0,0,0,0,0}.
compute quantc = {0,0,0,0,0,0}.
compute mcheck = 0.
compute ttt = 0.
compute plot = trunc( 1 ).
compute runerrs = make(50,1,0).
compute model = trunc( 1 ).
compute normal = 0.
compute ws= ( 0 =1).
compute olsdichm= ( 0 =1).
compute olsdichy= ( 0 =1).
compute mcx=trunc( 0 ).
compute mcm=trunc( 0 ).
do if (ws = 1).
do if (effsize = 1).
compute note(notes,1) = 19.
compute notes = notes + 1.
compute effsize=0.
end if.
do if (normal=1).
compute note(notes,1) = 16.
compute notes = notes + 1.
compute normal=0.
end if.
end if.
do if (model <> 4).

```



```

compute ws = 0.
end if.
do if (model > 0 and model < 4 and covmy <>
0).
compute covmy=0.
compute note(notes,1) = 33.
compute notes = notes + 1.
end if.
print/space=0.
do if (jn = 1 and model <> 1 and model <> 3).
compute note(notes,1) = 7.
compute notes = notes + 1.
end if.
do if ((model > 76) or (model < 1)).
compute model = 77.
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 19.
end if.
compute itprob=0.
do if (mcx < 0).
compute mcx=0.
end if.
do if (mcx > 4).
compute mcx=0.
end if.
do if (mcm < 0).
compute mcm=0.
end if.
do if (mcm > 4).
compute mcm=0.
end if.
do if (mcx > 0 and mcm > 0).
compute errs=errs+1.
compute runerrs(errs,1)=34.
compute criterr=1.
end if.
do if ((model <> 1 and model <> 4) and (mcx >
0 or mcm > 0)).
compute errs=errs+1.

```

```

compute runerrs(errs,1)=35.
compute criterr=1.
end if.
do if ((model = 4) and (mcm > 0)).
compute errs=errs+1.
compute runerrs(errs,1)=41.
compute criterr=1.
end if.
do if ((mcx > 0) and jn <> 0).
compute jn=0.
compute note(notes,1) = 23.
compute notes = notes + 1.
end if.
do if (mmodval <> 999 and mcm > 0).
compute note(notes,1) = 24.
compute notes = notes + 1.
end if.
compute toteff = 0.
compute toteff = (( 0 = 1)*( 1 = 4 or 1 = 6)).
compute varorder = 2.
compute hc3 = ( 0 <> 0).
compute cname = 'xxx'.
compute centvar = {'xxx'}.
compute nmeds = ncol(mnames).
do if (longname=0).

```

```
compute criterr = 1.  
compute errs = errs+1.  
compute runerrs(errs,1) = 33.
```

1

2

3

```
end if.  
compute modelm =  
{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,6;  
1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,7;  
1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8;  
1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,9;  
1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,3,10;
```

1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,5,11;
1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,0,5,12;
1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5,13;
0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,2,14;
0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,2,15;
0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,4,16;
0,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,4,17;
0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,6,18;
0,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,6,19;
0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,6,20;
1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,7,21;
1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,7,22;
1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,23;
1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,0,0,24; 1,1,1,1

,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,25;
1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,26;
1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,27;
1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,7,28;
1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,7,29;
1,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,30;
1,1,0,1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,31;
1,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,32;
1,1,1,1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,33;
1,1,1,1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,34;
1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,35;
1,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,36;
1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,37;
1,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,38;
1,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,39;
1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,40;
1,0,0,1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,41;
1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,42;
1,0,0,1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,43;
1,0,0,1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,44;
1,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,45;
1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,46;
1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,47;
1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,48;
1,1,0,1,1,0,1,1,0,0,0,

```

0,0,0,0,0,0,0,0,49;
1,1,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,50;
1,1,1,1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,51;
1,1,1,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,52;
1,1,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,53;
1,1,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,54;
1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,55;
1,1,1,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,56;
1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,57;
1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,58;
1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,59;
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,60;
1,1,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,61;
1,1,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,62;
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,0,0,0,63;
1,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,64;
1,0,0,1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,65;
1,0,0,1,0,0,0,0,0,1,0,0,1,0,0,0,0,0,66;
1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,67;
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,68;
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,69;
1,0,0,1,0,0,0,0,0,0,0,0,1,1,0,0,0,0,70;
1,0,0,1,0,0,1,0,0,1,0,0,1,1,1,0,0,0,71;
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,1,1,0,72;
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,1,1,0,0

,73; 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,74;
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,75;
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,1,0,0,76;
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,77}.
compute wm = modelm(model, 1).
compute zm = modelm(model, 2).
compute wzm = modelm(model, 3).
compute vy = modelm(model, 4).
compute qy = modelm(model, 5).
compute vqy = modelm(model, 6).
compute wy = modelm(model, 7).
compute zy = modelm(model, 8).
compute wzy = modelm(model, 9).

```

```

compute vxy = modelm(model, 10).
compute qxy = modelm(model, 11).
compute vqxy = modelm(model, 12).
compute wmy = modelm(model, 13).
compute wvmy = modelm(model, 14).
compute wvxy = modelm(model, 15).
compute zmy = modelm(model, 16).
compute wzmy = modelm(model, 17).
compute xmy = modelm(model, 18).
compute imm=modelm(model, 19).
do if (model = 74).
compute vdich=0.
end if.
do if (ncol(xname) <> 1).
compute errs = errs+1.
compute runerrs(errs, 1) = 20.
compute criterr = 1.
end if.
do if (ncol(mnames) > 10).
compute errs = errs+1.
compute runerrs(errs, 1) = 39.
compute criterr = 1.
end if.
do if (ncol(yname) <> 1).
compute errs = errs+1.
compute runerrs(errs, 1) = 21.
compute criterr = 1.
end if.
do if (( 999 =0) and ( 999 =0) and (model=3 or
model=2) and (contrast=1)).
compute errs = errs+1.
compute runerrs(errs, 1) = 31.
compute criterr = 1.
end if.
compute xlist = (wm or zm or wzm or wy or zy
or wzy or vxy or qxy or vqxy or wvxy or xmy).
compute mlist = (vy or qy or vqy or zmy or wmy
or wzmy or xmy or (model < 4)).
compute bad=0.
do if (criterr = 0).

```



```

compute werr = 0.
compute verr = 0.
compute qerr = 0.
compute zerr = 0.
compute yerr = 1.
compute xerr = 1.
compute wlist = (wm or wzm or wy or wzy or
wm or wvmy or wvmy or wvxy or wzmy).
do if (wlist = 1 and wname = 'xxx').
compute werr = 1.
compute wlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (wlist = 1 and ((wname = qname) or
(wname = vname) or (wname = zname) or
(wname = xname) or (wname = yname))).
compute werr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 12).
end if.
compute zlist = (zm or wzm or zy or wzy or
zmy or wzmy).
do if (zlist = 1 and zname = 'xxx').
compute zerr = 1.
compute zlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (zlist = 1 and ((zname = qname) or
(zname = vname) or (zname = wname) or
(zname = xname) or (zname = yname))).
compute zerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 13.
end if.
compute qlist = (qy or vqy or qxy or vqxy).
do if (qlist = 1 and qname = 'xxx').
compute qerr = 1.
compute qlist = 0.

```

```

compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (qlist = 1 and ((qname = zname) or
(qname = vname) or (qname = wname) or
(qname = xname) or (qname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 14.
end if.
compute vlist = (vy or vqy or vxy or vqxy or
wvmy or wvxy).
do if (vlist = 1 and vname = 'xxx').
compute verr = 1.
compute vlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (vlist = 1 and ((vname = zname) or
(vname = qname) or (vname = wname) or
(vname = xname) or (vname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 15.
end if.
do if (wlist = 0 and wname <> 'xxx').
compute werr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 8.
end if.
do if (zlist = 0 and zname <> 'xxx').
compute zerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 9.
end if.
do if (qlist = 0 and qname <> 'xxx').
compute qerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 10.
end if.

```

```

do if (vlist = 0 and vname <> 'xxx').
compute verr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 11.
end if.
do if (hc3 = 1).
compute note(notes,1) = 3.
compute notes = notes+1.
end if.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute cons = make(n,1,1).
compute temp = (n*sscp(dat))-
(t(csum(dat))*(csum(dat))).
compute temp = temp/(n*(n-1)).
compute temp = csum(((diag(temp))=0)).
compute temp2=1.
do if (ws = 1).
compute temp2 = (n*sscp(tmp2))-
(t(csum(tmp2))*(csum(tmp2))).
compute temp2 = temp2/(n*(n-1)).
end if.
do if ((temp > 0 and ws = 0) or (temp > 0 and
ws = 1 and temp2 > 0)).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 27.
end if.
compute nmeds = ncol(mnames).
compute mcmats=ident(nmeds*2).
compute mccoef=make(nmeds*2,1,0).
compute sobel = make(nmeds,4,-999).
do if (model = 6 and nmeds > 4).
compute errs = errs+1.
compute runerrs(errs,1)=2.
end if.
do if (model < 4 and nmeds > 1).

```

```

compute errs = errs+1.
compute runerrs(errs,1) = 3.
end if.
compute nmods = (model = 74).
compute bad = 0.
compute intcnt = 1.
compute modvals = 0.
compute modvalsd = 0.
compute yintemp = {'int_1', 'int_2', 'int_3',
'int_4', 'int_5', 'int_6', 'int_7', 'int_8', 'int_9',
'int_10', 'int_11', 'int_12', 'int_13', 'int_14',
'int_15'}.
compute yintemp = {yintemp, 'int_16', 'int_17',
'int_18', 'int_19', 'int_20', 'int_21', 'int_22',
'int_23', 'int_24', 'int_25', 'int_26', 'int_27',
'int_28'}.
compute yintemp = {yintemp, 'int_29', 'int_30',
'int_31', 'int_32', 'int_33', 'int_34', 'int_35',
'int_36', 'int_37', 'int_38', 'int_39', 'int_40'}.
compute yintemp = {yintemp, 'int_41', 'int_42',
'int_43', 'int_44', 'int_45', 'int_46', 'int_47',
'int_48', 'int_49', 'int_50', 'int_51', 'int_52'}.
compute yintemp = {yintemp, 'int_53', 'int_54',
'int_55', 'int_56', 'int_57', 'int_58', 'int_59',
'int_60', 'int_61', 'int_62', 'int_63', 'int_64'}.
compute
cntname={'(C1)';'(C2)';'(C3)';'(C4)';'(C5)';'(C6)';(
C7)';'(C8)';'(C9)';'(C10)';'(C11)';'(C12)';'(C13)';(
C14)';'(C15)';'(C16)';'(C17)'}.
compute
cntname={cntname;'(C18)';'(C19)';'(C20)';'(C21
)';'(C22)';'(C23)';'(C24)';'(C25)';'(C26)';'(C27)';'(
C28)';'(C29)';'(C30)';'(C31)'}.
compute
cntname={cntname;'(C32)';'(C33)';'(C34)';'(C35
)';'(C36)';'(C37)';'(C38)';'(C39)';'(C40)';'(C41)';(
C42)';'(C43)';'(C44)';'(C45)'}.
compute
cntname={cntname;'(C46)';'(C47)';'(C48)';'(C49

```

```

);'(C50)';'(C51)';'(C52)';'(C53)';'(C54)';'(C55)';'(
C56)';'(C57)';'(C58)';'(C59)}.

compute
cntname={cntname;'(C60)';'(C61)';'(C62)';'(C63
)';'(C64)';'(C65)';'(C66)';'(C67)';'(C68)';'(C69)';'(
C70)';'(C71)';'(C72)';'(C73)}.

compute
cntname={cntname;'(C74)';'(C75)';'(C76)';'(C77
)';'(C78)';'(C79)';'(C80)';'(C81)';'(C82)';'(C83)';'(
C84)';'(C85)';'(C86)';'(C87)}.

compute
cntname={cntname;'(C88)';'(C89)';'(C90)';'(C91
)';'(C92)';'(C93)';'(C94)';'(C95)';'(C96)';'(C97)';'(
C98)';'(C99)';'(C100)';'(C101)}.

compute
cntname={cntname;'(C102)';'(C103)';'(C104)';'(
C105)}.

compute apathnam={'a path'; 'a1 path'; 'a2
path'; 'a3 path'; 'a4 path'; 'a5 path'; 'a6 path';
'a7 path'; 'a8 path'; 'a9 path'; 'a10 path'}.

compute bpathnam={'b path'; 'b1 path'; 'b2
path'; 'b3 path'; 'b4path'; 'b5path'; 'b6 path'; 'b7
path'; 'b8 path'; 'b9 path'; 'b10 path'}.

compute modvnm = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.

compute modvnm2 = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.

compute mlab = { ' M1 ='; ' M2 ='; ' M3 ='; '
M4 ='; ' M5 ='; ' M6 ='; ' M7 ='; ' M8 ='; '
M9 ='; ' M10 ='}.

compute m = make(n,nmeds,1).
compute mmat = make(16,nmeds,0).
compute ymat = make(8,nmeds,0).
compute deco = make(10,1,0).
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
compute iterate = abs(trunc( 10000 )).
compute converge = abs( 0.00000001 ).

```

```

compute boot = abs(trunc( 5000 )).
do if (ws=1 and 0 > 0).
compute note(notes,1)=17.
compute notes=notes+1.
end if.
compute mc=(abs(trunc( 0 ))*(1-ws)).
do if (mc > 0 and model > 5).
do if (boot = 0).
compute boot=mc.
end if.
compute mc=0.
compute note(notes,1)=12.
compute notes = notes+1.
end if.
compute adjust = 0.
do if (mc > 0 and (model > 3 and model < 6)).
compute boot=0.
compute bconoff=0.
end if.
do if (boot > 0 and mc > 0).
compute mc = 0.
end if.
compute savboot = 0.
compute saveboot = ( 0 = 1).
do if (ws=1 and saveboot=1).
compute saveboot=0.
compute note(notes,1) = 20.
compute notes = notes + 1.
compute normal=0.
end if.
do if (saveboot =1 and boot > 0 and model >
3).
compute savboot = 1.
end if.
do if (boot <> 0 or mc <> 0).
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
loop.

```

```

compute cilow = rnd(bootesz*(1-(conf/100))/2).
compute cihigh =
trunc((bootesz*(conf/100)+(bootesz*(1-
(conf/100))/2)))+1.
do if (cilow < 1 or cihigh > bootesz).
compute
bootesz=trunc((bootesz+1000)/1000)*1000.
compute adjust = 1.
end if.
end loop if (cilow gt 0 and cihigh le bootesz).
compute boot=bootesz.
do if (mc > 0).
compute mc = bootesz.
end if.
do if (boot > 0 and mc > 0 and (model > 3 and
model < 6)).
compute boot=0.
end if.
do if (adjust = 1 and boot > 0).
compute note(notes,1)=6.
compute notes = notes+1.
end if.
do if (adjust = 1 and mc > 0).
compute note(notes,1)=13.
compute notes = notes+1.
end if.
end if.
do if (model = 6 and nmeds > 1).
compute mmpaths =
make((nmeds+2),(nmeds+2),0).
do if (nmeds = 2).
compute indboot = make(boot+1, 3, 999).
else if (nmeds = 3).
compute indboot = make(boot+1, 7, 999).
else if (nmeds = 4).
compute indboot = make(boot+1, 15, 999).
end if.
compute indlbl = {'Total:','Ind1 :','Ind2 :','Ind3
:','Ind4 :','Ind5 :','Ind6 :','Ind7 :','Ind8 :','Ind9

```

```

:','Ind10:','Ind11:','Ind12:','Ind13:','Ind14:','
'Ind15:'}).
compute indlbl2 = {'Ind1','Ind2','Ind3','Ind4';
'Ind5','Ind6','Ind7','Ind8','Ind9','Ind10';
'Ind11','Ind12','Ind13','Ind14','Ind15'}.
compute indces = make(boot+1, 4, 999).
end if.
do if (model < 4).
compute boot = 0.
compute cmat = make(10,1,0).
compute zmat = make(10,1,0).
end if.
compute nvarch = make(1,ncol(dat),0).
compute wmatch = 0.
compute zmatch = 0.
compute vmatch = 0.
compute qmatch = 0.
compute mmatch=0.
compute minprobe=0.
compute maxprobe=0.
loop i = 1 to ncol(vnames).
do if (vnames(:,i)=yname).
compute y = dat(:,i).
compute nvarch(1,i)=1.
compute yerr = 0.
do if ((yname = xname) or (yname = wname) or
(yname = zname) or (yname = vname) or
(yname = qname)).
compute errs = errs+1.
compute runerrs(errs,1)=17.
end if.
end if.
do if (vnames(:,i)=xname).
compute x = dat(:,i).
compute nvarch(1,i)=1.
compute xdich = 1.
compute xerr=0.
loop jj = 1 to n.
do if ((x(jj,1) <> cmin(x)) and (x(jj,1) <>
cmax(x))).

```



```

compute xdich = 0.
break.
end if.
end loop.
compute xmean = csum(x)/n.
do if (center = 1 and (model < 4 or xlist > 0)
and mcx = 0).
compute meanvec = make(n,1, xmean).
compute x = x-meanvec.
compute centvar = {centvar, xname}.
end if.
compute xmean = csum(x)/n.
compute tmp = x-(cons*xmean).
compute xsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
do if (xdich = 0).
compute quantc(1,6) = 1.
compute matx = {xmean-xsd; xmean;
xmean+xsd}.
do if ( 999 = 999 and quantile = 0).
do if ((matx(1,1) < cmin(x)) and model = 74).
compute matx(1,1)=cmin(x).
compute minprobe=1.
end if.
do if ((matx(3,1) > cmax(x)) and model = 74).
compute matx(3,1)=cmax(x).
compute maxprobe=1.
end if.
end if.
do if (quantile = 1).
compute quantd(1,6) = 1.
compute quantc(1,6) = 0.
compute tmp = x.
compute tmp(GRADE(x)) = x.
compute matx =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (xdich = 1).

```

```

compute matx = {cmin(x); cmax(x)}.
do if (model=74).
compute matx=cmin(x).
compute mod74dic=1.
end if.
end if.
do if ( 999 <> 999).
compute matx = 999.
compute quantd(1,6) = 0.
compute quantc(1,6) = 0.
end if.
end if.
do if (werr = 0 and wlist = 1).
do if (vnames(:,i)=wname).
compute werr = 0.
compute wmatch = 1.
compute w = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(w)/n).
compute w = w-meanvec.
compute centvar = {centvar, wname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute wmean = csum(w)/n.
compute tmp = w-(cons*wmean).
compute wsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute wdich = 1.
loop jj = 1 to n.
do if ((w(jj,1) <> cmin(w)) and (w(jj,1) <>
cmax(w))).
compute wdich = 0.
break.
end if.
end loop.
do if (model = 3).
compute jndich=wdich.
compute jnmin=cmin(w).
compute jnmax=cmax(w).
end if.

```

```

do if (wdich = 0).
compute matw = {wmean-wsd; wmean;
wmean+wsd}.
do if ( 999 = 999 and quantile = 0).
do if (matw(1,1) < cmin(w)).
compute matw(1,1)=cmin(w).
compute minprobe=1.
end if.
do if (matw(3,1) > cmax(w)).
compute matw(3,1)=cmax(w).
compute maxprobe=1.
end if.
end if.
compute quantc(1,1) = 1.
do if (quantile = 1).
compute quantd(1,1) = 1.
compute quantc(1,1) = 0.
compute tmp = w.
compute tmp(GRADE(w)) = w.
compute matw =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (wdich = 1).
compute matw = {cmin(w); cmax(w)}.
compute wvdich=1.
compute cmaxw=cmax(w).
compute cminw=cmin(w).
end if.
do if ( 999 <> 999).
compute matw = 999.
compute quantd(1,1) = 0.
compute quantc(1,1) = 0.
end if.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm(1,1)=wname.
compute modmatp(1,1) = 1.

```

```

end if.
end if.
do if (zerr = 0 and zlist = 1).
do if (vnames(:,i)=zname).
compute zerr = 0.
compute zmatch = 1.
compute z = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(z)/n).
compute z = z-meanvec.
compute centvar = {centvar, zname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute zmean = csum(z)/n.
compute tmp = z-(cons*zmean).
compute zsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute zdich = 1.
loop jj = 1 to n.
do if ((z(jj,1) <> cmin(z)) and (z(jj,1) <>
cmax(z))).
compute zdich = 0.
break.
end if.
end loop.
do if (zdich = 0).
compute matz = {zmean-zsd; zmean;
zmean+zsd}.
do if ( 999 = 999 and quantile = 0).
do if (matz(1,1) < cmin(z)).
compute matz(1,1)=cmin(z).
compute minprobe=1.
end if.
do if (matz(3,1) > cmax(z)).
compute matz(3,1)=cmax(z).
compute maxprobe=1.
end if.
end if.
compute quantc(1,2) = 1.
do if (quantile = 1).

```

```

compute quantd(1,2) = 1.
compute quantc(1,2) = 0.
compute tmp = z.
compute tmp(GRADE(z)) = z.
compute matz = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (zdich = 1).
compute matz = {cmin(z); cmax(z)}.
compute cmaxz=cmax(z).
compute cminz=cmin(z).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matz = 999.
compute quantd(1,2) = 0.
compute quantc(1,2) = 0.
end if.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm(1,2)=zname.
compute modmatp(1,2) = 1.
end if.
end if.
do if (verr = 0 and vlist = 1).
do if (vnames(:,i)=vname).
compute verr = 0.
compute vmatch = 1.
compute v = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(v)/n).
compute v = v-meanvec.
compute centvar = {centvar, vname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute vmean = csum(v)/n.
compute tmp = v-(cons*vmean).

```

```

compute vsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute vdich = 1.
loop jj = 1 to n.
do if ((v(jj,1) <> cmin(v)) and (v(jj,1) <>
cmax(v))).
compute vdich = 0.
break.
end if.
end loop.
do if (vdich = 0).
compute matv = {vmean-vsd; vmean;
vmean+vsd}.
do if ( 999 = 999 and quantile = 0).
do if (matv(1,1) < cmin(v)).
compute matv(1,1)=cmin(v).
compute minprobe=1.
end if.
do if (matv(3,1) > cmax(v)).
compute matv(3,1)=cmax(v).
compute maxprobe=1.
end if.
end if.
compute quantc(1,3) = 1.
do if (quantile = 1).
compute quantd(1,3) = 1.
compute quantc(1,3) = 0.
compute tmp = v.
compute tmp(GRADE(v)) = v.
compute matv = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (vdich = 1).
compute matv = {cmin(v); cmax(v)}.
compute wvdich=1.
compute cmaxv=cmax(v).
compute cminv=cmin(v).
end if.
do if ( 999 <> 999).

```

```

compute matv = 999.
compute quantd(1,3) = 0.
compute quantc(1,3) = 0.
end if.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm(1,3)=vname.
compute modmatp(1,3) = 1.
end if.
end if.
do if (qerr = 0 and qlist = 1).
do if (vnames(:,i)=qname).
compute qerr = 0.
compute qmatch = 1.
compute q = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(q)/n).
compute q = q-meanvec.
compute centvar = {centvar, qname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute qmean = csum(q)/n.
compute tmp = q-(cons*qmean).
compute qsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute qdich = 1.
loop jj = 1 to n.
do if ((q(jj,1) <> cmin(q)) and (q(jj,1) <>
cmax(q))).
compute qdich = 0.
break.
end if.
end loop.
do if (qdich = 0).
compute matq = {qmean-qsd; qmean;
qmean+qsd}.
do if ( 999 = 999 and quantile = 0).
do if (matq(1,1) < cmin(q)).
compute matq(1,1)=cmin(q).
compute minprobe=1.

```

```

end if.
do if (matq(3,1) > cmax(q)).
compute matq(3,1)=cmax(q).
compute maxprobe=1.
end if.
end if.
compute quantc(1,4) = 1.
do if (quantile = 1).
compute quantd(1,4) = 1.
compute quantc(1,4) = 0.
compute tmp = q.
compute tmp(GRADE(q)) = q.
compute matq = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (qdich = 1).
compute matq = {cmin(q); cmax(q)}.
compute cmaxq=cmax(q).
compute cminq=cmin(q).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matq = 999.
compute quantd(1,4) = 0.
compute quantc(1,4) = 0.
end if.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm(1,4)=qname.
compute modmatp(1,4) = 1.
end if.
end if.
do if (vnames(:,i)=cname).
compute cld = dat(:,i).
compute cvname = vnames(:,i).
compute nvarch(1,i)=1.
compute clsmtch = 1.
end if.

```



```

loop j = 1 to ncol(mnames).
do if (vnames(:,i)=mnames(1,j)).
compute mmatch = mmatch + 1.
compute m(:,j)=dat(:,i).
do if (center = 1 and nvarch(1,i) = 0 and mlist >
0 and mcm = 0).
compute meanvec = make(n,1,csum(m(:,j))/n).
compute m(:,j) = m(:,j)-meanvec.
compute centvar = {centvar, mnames(1,j)}.
end if.
compute nvarch(1,i)=1.
compute dichm = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute dichm = 0.
break.
end if.
end loop.
do if ((dichm = 1) and (olsdichm = 1)).
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichm = 1 and model > 3 and mcheck = 0
and oldsdichm = 0).
compute errs= errs+1.
compute runerrs(errs,1) = 1.
compute mcheck = 1.
end if.
do if ((model <= 3) and (ncol(mnames)=1)).
compute nmods = nmods + 1.
compute mmean = csum(m(:,j))/n.
compute tmp = m(:,j)-(cons*mmean).
compute msd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute mdich = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute mdich = 0.
break.

```

```

end if.
end loop.
do if (model = 1).
compute jndich=mdich.
compute jnmin=cmin(m(:,j)).
compute jnmax=cmax(m(:,j)).
end if.
do if (mdich = 0).
compute matm = {mmean-msd; mmean;
mmean+msd}.
do if ( 999 = 999 and quantile = 0).
do if (matm(1,1) < cmin(m(:,j))).
compute matm(1,1)=cmin(m(:,j)).
compute minprobe=1.
end if.
do if (matm(3,1) > cmax(m(:,j))).
compute matm(3,1)=cmax(m(:,j)).
compute maxprobe=1.
end if.
end if.
compute quantc(1,5) = 1.
do if (quantile = 1).
compute quantd(1,5) = 1.
compute quantc(1,5) = 0.
compute tmp = m(:,j).
compute tmp(GRADE(m(:,j))) = m(:,j).
compute matm = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (mdich = 1).
compute matm = {cmin(m); cmax(m)}.
end if.
do if ( 999 <> 999).
compute matm = 999.
compute quantd(1,5) = 0.
compute quantc(1,5) = 0.
end if.
compute modmatv(1,5)=nrow(matm).

```

```

compute modmat((1:nrow(matm)), 5) = matm.
compute modvnm(1,5)=mnames(1,j).
compute modmatp(1,5) = 1.
end if.
end if.
end loop.
end loop.
do if (minprobe=1).
compute note(notes,1) = 14.
compute notes = notes + 1.
end if.
do if (maxprobe=1).
compute note(notes,1) = 15.
compute notes = notes + 1.
end if.
do if (cname <> 'xxx' and clsmatch = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 23.
end if.
do if (cname <> 'xxx').
do if ((cname = zname) or (cname = vname)
or (cname = wname) or (cname = xname) or
(cname = yname) or (cname = qname)).
compute errs = errs+1.
compute runerrs(errs,1) = 24.
end if.
end if.
do if (wlist = 1 and werr = 0 and wmatch = 0).
compute werr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (zlist = 1 and zerr = 0 and zmatch = 0).
compute zerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (qlist = 1 and qerr = 0 and qmatch = 0).
compute qerr = 3.
compute errs = errs+1.

```

```

compute runerrs(errs,1) = 6.
end if.
do if (vlist = 1 and verr = 0 and vmatch = 0).
compute verr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (yerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 16.
end if.
do if (xerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 32.
end if.
do if (model = 6 and nmeds < 2).
compute errs = errs+1.
compute runerrs(errs,1) = 18.
end if.
do if (mmatch < ncol(mnames)).
compute errs=errs+1.
compute runerrs(errs,1) = 25.
end if.
do if (cname <> 'xxx').
compute cld = design(cld).
compute cluster = ncol(cld).
compute cld = cld(:,2:ncol(cld)).
compute clsdmy = ncol(cld).
do if (clsdmy > 19).
compute errs = errs+1.
compute runerrs(errs,1) = 26.
end if.
do if (mcx > 0 or mcm > 0).
compute errs = errs+1.
compute runerrs(errs,1) = 38.
end if.
end if.
compute dichy = 1.
loop jj = 1 to n.

```

```

do if ((y(jj,1) <> cmin(y)) and (y(jj,1) <>
cmax(y))).
compute dichy = 0.
break.
end if.
end loop.
do if ((dichy = 1) and (olstdichy = 1)).
compute dichy=0.
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichy = 1).
do if (toteff=1).
compute toteff=0.
compute note(notes,1) = 31.
compute notes = notes + 1.
end if.
compute jncrit=xp2*xp2.
do if (ws = 1).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 28.
end if.
end if.
compute ncovs = ncol(dat)-rsum(nvarch).
do if (effsize = 1 and covmy <> 0 and ncovs >
0 and model > 3 and model < 7).
compute note(notes,1) = 22.
compute notes = notes + 1.
compute effsize = 0.
end if.
do if (ws = 1 and ncovs < nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 29.
end if.
do if (ws = 1 and ncovs > nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 30.

```

```

end if.
end if.
do if (errs = 0).
do if (rsum(quantd) > 0).
compute note(notes,1) = 4.
compute notes = notes+1.
end if.
do if (rsum(quantc) > 0).
compute note(notes,1) = 5.
compute notes = notes+1.
end if.
do if (ncovs > 0).
compute c = make(n,ncovs,0).
compute cnames = {'x'}.
compute j = 1.
loop i = 1 to ncol(vnames).
do if (nvarch(1,i)) = 0.
compute c(:,j) = dat(:,i).
compute nvarch(1,i)=1.
compute j=j+1.
compute cnames = {cnames, vnames(:,i)}.
end if.
end loop.
compute cnames = cnames(1,2:ncol(cnames)).
do if (ws = 1).
compute covmean=csum(c)/n.
loop i = 1 to ncovs.
compute meanvec = make(n,1, covmean(1,i)).
compute c(:,i) = c(:,i)-meanvec.
end loop.
compute centvar = {centvar, cnames}.
compute x=c(:,1:ncovs).
do if (ncovs=nmeds).
compute ncovs=0.
end if.
end if.
end if.
compute names = {yname, xname, mnames,
wname, zname, vname, qname}.
do if (ncovs > 0).

```

```

compute names = {names, cnames}.
end if.
do if (dichy = 1 and effsize = 1).
compute note(notes,1) = 2.
compute notes = notes+1.
end if.
do if (model > 3 and model < 6).
compute indeff=make(nmeds,1,0).
compute indboot=make(boot+1,nmeds,999).
do if (mc > 0).
compute indboot=make(mc+1,nmeds,999).
end if.
do if (effsize = 1 and dichy = 0).
compute rmeff=make(boot+1,nmeds+1,999).
compute abpseff=make(boot+1,nmeds+1,999).
compute abcseff=make(boot+1,nmeds+1,999).
compute pmeff=make(boot+1,nmeds+1,999).
compute r245 = make(boot+1,1,999).
compute kappa2 = make(boot+1,1,999).
end if.
end if.
do if (model = 6 and effsize = 1 and dichy = 0).
compute
rmeff=make(boot+1,ncol(indboot),999).
compute
abpseff=make(boot+1,ncol(indboot),999).
compute
abcseff=make(boot+1,ncol(indboot),999).
compute
pmeff=make(boot+1,ncol(indboot),999).
end if.
do if (nmods > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm(1,tmp) = modvnm(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.

```

```

end loop.
compute modmat=modmat(:,1:nmods).
compute modvnm=modvnm(:,1:nmods).
compute modmatv=modmatv(:,1:nmods).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvals =
make((modmatv(1,1)*modprod(1,1)),
nmods,0).
loop i = 1 to nmods.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvals)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvals(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
do if (model = 74).
compute modvals=matx.
compute modvnm=xname.
end if.
compute vmat = make(8,nrow(modvals),0).
compute vmat(1,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute vmat(5,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute indeff=make(nrow(modvals),1,0).
do if (model <> 5).
compute indboot = make(((boot+1)*nmeds),
nrow(modvals), -99999999).

```



```

compute indbootp =
make(boot+1,(nmeds*nmods), -99999999).
do if (imm > 4 and imm < 8).
do if (imm = 7).
compute immmv1=matw.
compute immmv2=matv.
end if.
do if (imm = 5).
compute immmv1=matw.
compute immmv2=matz.
end if.
do if (imm = 6).
compute immmv1=matv.
compute immmv2=matq.
end if.
compute
indbootp1=make((boot+1),(nmeds*(nrow(immmv2))),-99999999).
compute
indbootp2=make((boot+1),(nmeds*(nrow(immmv1))),-99999999).
compute onesmv1=make(1,nrow(immmv1),1).
compute onesmv2=make(1,nrow(immmv2),1).
compute
nimvs=nrow(immmv1)+nrow(immmv2).
compute cmmval=0.
compute cmmlbs1={'xx'}.
compute cmmlbs2={'xx'}.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv2}.
compute
cmmlbt=make(nrow(immmv2),1,mnames(1,i)).
compute cmmlbs1={cmmlbs1;cmmlbt}.
end loop.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv1}.
compute
cmmlbt=make(nrow(immmv1),1,mnames(1,i)).
compute cmmlbs2={cmmlbs2;cmmlbt}.
end loop.

```

```

compute cmmval=cmmval(2:nrow(cmmval),:).
end if.
end if.
end if.
do if (nmods> 0).
loop i = 1 to ncol(modvals).
do if (modvnm(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm(1,i)=qname).
compute qcol = i.
end if.
end loop.
end if.
do if (dichy = 1).
compute omx = cmax(y).
compute omn = cmin(y).
compute y = (y = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute data = {cons,y,m,x}.
compute datamed = data.
compute datayed = data.
compute datanm = {'constant'; yname;
t(mnames); xname}.
compute datanmm = {'constant'; yname;
t(mnames); xname}.
compute datanmy = {'constant'; yname;
t(mnames); xname}.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
do if (model < 4 and errs = 0).
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' ', ' '}.
compute datayed = {datayed, x&*m}.

```

```

compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(1,i) = 1.
compute vmat(2,i) = modvals(i,1).
end loop.
compute mmat = make(16,nmeds,1).
end if.
do if (model = 2 or model = 3).
compute int1 = x&*w.
compute datayed = {datayed, w, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,2).
compute vmat(3,i) = modvals(i,1).
compute vmat(4,i) = modvals(i,1)*modvals(i,2).
end loop.
end if.
do if (model = 3).
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*m.
compute int2 = x&*w&*m.
compute datayed = {datayed, int1, int2}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' X', wname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (model = 4 or model = 5).
compute vmat = make(8,1,1).

```

```

end if.
compute yintkey2 = yintkey.
do if (wm = 1).
compute int1 = x&*w.
compute datamed = {datamed, w,int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmm = {datanmm; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,wcol).
end loop.
do if (zm = 1).
compute int1 = x&*z.
compute datamed = {datamed, z, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(3,i) = modvals(i,zcol).
end loop.
end if.
do if (wzm = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*z.
compute int2 = x&*w&*z.
compute datamed = {datamed, int1, int2}.
loop i = 1 to nrow(modvals).

```

```

compute vmat(4,i) =
modvals(i,wcol)*modvals(i,zcol).
end loop.
end if.
end if.
compute mdatacol=ncol(datamed).
compute mintkey = yintkey.
compute yintkey = {' ',' ',' ',' ',' ',' '}.
compute medints = intcnt-1.
do if (vy = 1 or xmy = 1).
compute mp = 1.
do if (xmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,1).
end loop.
end if.
do if (vy = 1).
compute datayed = {datayed, v}.
compute datanmy = {datanmy; vname}.
compute mmods = 1.
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,vcol).
end loop.
do if (qy = 1).
compute mp = 2.
compute datayed = {datayed,q}.
compute datanmy = {datanmy; qname}.
compute mmods = 2.
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,qcol).
end loop.
end if.
do if (vqy = 1).
compute mp = 3.
compute datayed = {datayed, v&*q}.
compute mmods = 3.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,vcol)*modvals(i,qcol).
end loop.

```

```

end if.
end if.
compute mints = make(n,(nmeds*mp),0).
loop i = 0 to (nmeds-1).
do if (i = 0 and vqy = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
vname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (xmy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))&*x.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', xname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (qy = 1).
compute mints(:,((i*mp)+2))=m(:,(i+1))&*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqy = 1).
compute mints(:,((i*mp)+3))=m(:,(i+1))&*v&*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.

```

```

compute intcnt = intcnt+1.
end if.
end if.
end loop.
compute datayed = {datayed, mints}.
end if.
compute mp = 1.
do if (wvmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,wcol)*modvals(i,vcol).
end loop.
end if.
compute mints2 = make(n,(nmeds*mp),0).
do if (wmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,wcol).
end loop.
do if (wy = 0 and model > 3).
compute datayed = {datayed, w}.
compute datanmy = {datanmy; wname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wvmy = 1).
compute datayed = {datayed,w*v}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints2(:,((i*mp)+1))= m(:,(i+1))*w.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wvmy = 1).

```

```

compute mints2(:,((i*mp)+2))=
m(:,(i+1))&*w&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints2}.
end if.
compute mp = 1.
do if (zmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,zcol).
end loop.
do if (wzmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,zcol)&*modvals(i,wcol).
end loop.
end if.
end if.
do if (zmy = 1).
compute mints3 = make(n,(nmeds*mp),0).
do if (zy = 0).
compute datayed = {datayed, z}.
compute datanmy = {datanmy; zname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wzmy = 1 and wzy = 0).
compute datayed = {datayed,w&*z}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints3(:,((i*mp)+1))= m(:,(i+1))&*z.

```



```

compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wzmy = 1).
compute mints3(:,((i*mp)+2))=
m(:,(i+1))&*w&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', zname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints3}.
end if.
compute decoc = 1.
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
do if (wy = 1 and model > 3).
compute datayed = {datayed, w, x&*w}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm2(1,1)=wname.
compute modmatp(1,1) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (zy = 1).
compute datayed = {datayed,z,x&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.

```

```

compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm2(1,2)=zname.
compute modmatp(1,2) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (wzy = 1).
compute datayed = {datayed,w&*z,x&*w&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vxy = 1).
compute datayed = {datayed, x&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm2(1,3)=vname.
compute modmatp(1,3) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (qxy = 1).
compute datayed = {datayed, x&*q}.

```

```

compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm2(1,4)=qname.
compute modmatp(1,4) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqxy = 1).
compute datayed = {datayed, x&*v&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, 'X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end if.
do if (wvxy = 1).
compute datayed = {datayed, x&*w&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, 'X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute modvalsd = 0.
compute ttt = rsum(modmatp).
do if (rsum(modmatp) > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).

```

```

compute modvnm2(1,tmp) = modvnm2(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:ttt).
compute modvnm2=modvnm2(:,1:ttt).
compute modmatv=modmatv(:,1:ttt).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvalsd =
make((modmatv(1,1)*modprod(1,1)), ttt,0).
loop i = 1 to ttt.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvalsd)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvalsd(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
end if.
do if (ttt > 0).
loop i = 1 to ncol(modvalsd).
do if (modvnm2(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm2(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm2(1,i)=vname).

```

```

compute vcol = i.
end if.
do if (modvnm2(1,i)=qname).
compute qcol = i.
end if.
end loop.
compute directv = make(nrow(modvalsd),1,1).
do if (wy = 1).
compute directv ={directv, modvalsd(:,wcol)}.
end if.
do if (zy = 1).
compute directv ={directv, modvalsd(:,zcol)}.
end if.
do if (wzy = 1).
compute directv ={directv,
(modvalsd(:,wcol)&*modvalsd(:,zcol))}.
end if.
do if (vxy = 1).
compute directv ={directv, modvalsd(:,vcol)}.
end if.
do if (qxy = 1).
compute directv ={directv, modvalsd(:,qcol)}.
end if.
do if (vqxy = 1).
compute directv ={directv,
(modvalsd(:,vcol)&*modvalsd(:,qcol))}.
end if.
do if (wvxy = 1).
compute directv ={directv,
(modvalsd(:,vcol)&*modvalsd(:,wcol))}.
end if.
end if.
compute ydatacol=ncol(datayed).
do if (ncovs > 0).
do if (covmy <> 2).
compute datamed = {datamed,c}.
end if.
do if (covmy <> 1).
compute datayed = {datayed,c}.
end if.

```

```

do if (covmy = 1 and model = 4 and mcx > 0).
compute datayed = {datayed,c}.
end if.
compute covmeans = csum(c)/n.
end if.
do if (cluster > 0).
compute datamed = {datamed,cld}.
compute datayed = {datayed, cld}.
compute cldmeans = csum(cld)/n.
end if.
compute mst = 3.
compute mnd = mst+nmeds-1.
compute ydatacol=ncol(datayed).
compute mdatacol=ncol(datamed).
do if (ncovs > 0).
compute datanmy = {datanmy; t(cnames)}.
do if (model > 3).
compute datanmm = {datanmm; t(cnames)}.
end if.
end if.
compute datanmy = {'constant';
datanmy(3:nrow(datanmy),1)}.
do if (model > 3).
compute datanmm = {'constant';
datanmm(3:nrow(datanmm),1)}.
end if.
compute amm = make(2,1,0).
compute abmm = make(2,1,0).
compute mnv = csum(datayed(:,2)/n).
compute mnv = make(n,1,mnv).
compute ssty = csum((datayed(:,2)-mnv)**2).
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
do if (ws=1).
compute stddevm=diag(sigma).
compute
stddevm=sqrt(stddevm(3:(nmeds+2),1)).
end if.
compute stddevy = sqrt(sigma(2,2)).

```

```

compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute r2xy =
(sigma(2,(3+nmeds))/(stddevy*stddevx))&**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))&**2.
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).
do if (model = 4 and nmeds = 1 and cluster = 0
and ncovs = 0).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
compute datatm = datamed.
compute dataty = datayed.
compute mdlms2 = { '1' ; yname; xname}.
compute mdlms = {'Model ='; ' Y ='; ' X ='
}.

```

```

do if (ws = 1).
compute mdlrms2 = { '1' ; yname}.
compute mdlrms = {'Model ='; '  Y ='}.
end if.
loop i = 1 to ncol(mnames).
compute mdlrms2 = {mdlrms2; mnames(1,i)}.
do if (i = 1 and ncol(mnames) = 1).
compute mdlrms = {mdlrms; '  M ='}.
else.
compute mdlrms = {mdlrms; mlab(i,1)}.
end if.
end loop.
do if (wname <> 'xxx').
compute mdlrms2 = {mdlrms2; wname}.
compute mdlrms = {mdlrms; '  W ='}.
end if.
do if (zname <> 'xxx').
compute mdlrms2 = {mdlrms2; zname}.
compute mdlrms = {mdlrms; '  Z ='}.
end if.
do if (vname <> 'xxx').
compute mdlrms2 = {mdlrms2; vname}.
compute mdlrms = {mdlrms; '  V ='}.
end if.
do if (qname <> 'xxx').
compute mdlrms2 = {mdlrms2; qname}.
compute mdlrms = {mdlrms; '  Q ='}.
end if.
do if (jn = 1 and model = 1 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
do if (jn = 1 and model = 3 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
compute mdpbe=0.
compute medte2=0.
do if ( 1 =4 and mcx > 0).
compute medte2=1.

```



```

end if.
do if (((mcx > 0) or (mcm > 0)) AND model =
1).
compute mdpbe=1.
compute dd={datayed(:,4),datayed(:,3)}.
do if (ncovs > 0).
compute dd={datayed(:,6:(5+ncovs)),dd}.
end if.
compute dd={datayed(:,2),dd}.
end if.
compute yes = ((nrow(modvals)=1) and
(contrast = 1) and (model = 3 or model = 2)).
compute yes2 = ((nrow(modvals)=1) and
(contrast = 1) and (model = 2)).
print/title = '***** PROCESS Procedure
for SPSS Release 2.16.3 *****'.
print/title = '      Written by Andrew F. Hayes,
Ph.D.    www.afhayes.com'.
print/title = '  Documentation available in
Hayes (2013).
www.guilford.com/p/hayes3'/space=0.
print mdlnms2/title =
*****
*****'/rnames = mdlnms/format =
a8.
do if (ncovs > 0).
print cnames/title = 'Statistical Controls:'/rlabels
= 'CONTROL='/format a8.
end if.
print n/title = 'Sample size'/format F10.0.
do if (cluster > 0).
print cluster/rnames = cvname/title =
'Clustering Variable and Number of Clusters'.
end if.
do if (model > 3 and ( 'random' <> 'random')).
compute seedt= 'random'.
print seedt/title='Custom'/format=A12/rlabel =
'Seed:'.
end if.
do if (mdpbe=0 and medte2 = 0).

```

```

loop bt = 1 to (boot+1).
do if (bt = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)),-999).
end if.
compute bootcoef = 0.
do if (bt > 1).
loop.
compute v=trunc(uniform(n,1)*n)+1.
compute datayed = dataty(v,:).
do if (ws=1).
compute
bmeans=csum(datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed)))/n.
compute cntmat=make(n,nmeds,1).
compute bmeans2=bmeans.
do if (nmeds > 1).
compute bmeans2=mddiag(bmeans).
end if.
compute cntmat=cntmat*bmeans2.
compute datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed))=datayed(:,(ncol(data
yed)-nmeds+1):ncol(datayed))-cntmat.
end if.
compute detcheck=det(t(datayed)*datayed).
compute rk=(detcheck <> 0).
do if (model > 3).
compute datamed = datatm(v,:).
compute detcheck=det(t(datamed)*datamed).
do if (rk=1).
compute rk=(detcheck <> 0).
end if.
end if.
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
compute temp=diag(sigma).
compute bad = bad+(1-rk).
compute false = 1.
end loop if (rk = 1).

```

```

compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).
do if (model = 4 and nmeds = 1 and ncovs = 0
and cluster = 0).
compute r2xy =
(sigma(2,4)/(stddevy*stddevx))**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))**2.
compute sstot = sigma(2,2)*(n-1).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
end if.
do if (model > 3).
loop im = 1 to nmeds.

```

```

compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant';
datanmm((2+nmeds):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
do if (model = 6).
do if (im = 1).
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
end if.
do if (im > 1).
compute xm={cons, datamed(:,3:(im+1)),
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant'; datanmm(2:im,1);
datanmm((mnd):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
compute
mmpaths((im+1),(2:im))=t(coeff(2:im,1)).
end if.
end if.
do if (ws = 1).
compute
coeff(1,1)=csum(datamed(:,(2+im)))/n.
end if.
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
do if (bt = 1).
compute resid=datamed(:,(2+im))-xm*coeff.
compute sse =cssq(resid).
compute mse = sse/(n-ncol(xm)).
compute mnv = csum(data(:,(2+im)))/n.
compute mnv = make(n,1,mnv).

```

```

compute sstm = csum((data(:,(2+im))-
mnv)**2).
compute k3 = nrow(coeff).
do if (hc3 = 1).
compute h = xm(:,1).
loop i3=1 to n.
compute h(i3,1)= xm(i3,:)*invXtX*t(xm(i3,:)).
end loop.
loop i3=1 to k3.
compute xm(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xm(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xm(:,i3) = sqrt(mse)&*xm(:,i3).
end loop.
end if.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute hccov=invXtX*t(xm)*xm*invXtX.
compute mcmats(im,im)=hccov(2,2).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*hccov*lmat)*((t(lmat)
)*coeff))/dfnum.
compute coeff = coeff(1:(nrow(coeff)-
clsdmy),1).
compute mcoeff(im,1)=coeff(2,1).
compute standerr =
sqrt(diag(invXtX*t(xm)*xm*invXtX)).
compute standerr = standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1).
compute standerr(1,1)=stddevm(im,1)/sqrt(n).
end if.
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-
ncol(xm)))).

```

```

compute temp=(n-ncol(xm)).
do if (ws=1).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute temp=(n-1).
end if.
compute xd = abs(xp2).
compute temp = (temp* (exp((temp-
(5/6))*((xd/(temp-(2/3)+(.11/temp)))*(xd/(temp-
(2/3)+(.11/temp)))))-1)).
compute temp1 = coeff-
sqrt(abs(temp))*standerr.
compute temp2 =
coeff+sqrt(abs(temp))*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (ws=1).
compute op=op(1,:).
compute xnmn =
apathnam(((nmeds>1)+1+(im-1)),1).
end if.
compute sobel(im,1) = coeff(2,1).
compute sobel(im,2)=standerr(2,1).
compute temp = mnames(1,im).
compute r2full = 1-(sse/sstm).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute summ = {sqrt(r2full), r2full, mse,
Fratio, dfnum, dfden, pfr}.
do if (detail = 1).
print temp/title =
*****
*****'/rlabels = 'Outcome:'/format
= A8.
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels 'R',
'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.

```

```

print op/title = 'Model'/rnames = xnmn/clabels
'coeff', 'se', 't', 'p', 'LLCI', 'ULCI'/format = F10.4.
do if (covcoeff=1 and ws <> 1).
compute
hccovtmp=hccov(1:nrow(op),1:nrow(op)).
compute cnamestp=t(xnmn).
print hccovtmp/title='Covariance matrix of
regression parameter
estimates'/rnames=xnmn/cnames=cnamestp/f
ormat= F10.4.
end if.
do if (nmods > 0 and nrow(mintkey) > 1).
print mintkey/title = 'Product terms key:.'/format
= A8.
end if.
do if (ws = 1).
compute wsd f=n-1.
print wsd f/title=' '/rlabels = 'df = '/space=0.
end if.
end if.
end if.
compute ymat(1,im) = coeff(2,1).
do if (ws = 1).
compute ymat(1,im)=coeff(1,1).
end if.
do if (wm = 1).
compute ymat(2,im) = coeff(4,1).
do if (zm = 1).
compute ymat(3,im) = coeff(6,1).
do if (wzm = 1).
compute ymat(4, im) = coeff(8,1).
end if.
end if.
end if.
do if (model = 6).
compute mmpaths((im+1),1)=coeff((im+1),1).
end if.
end loop.
end if.
loop totlp = 1 to (1+(toteff*(bt = 1))).

```

```

do if (toteff = 1 and totlp =2).
compute xy =
{cons,datayed(:,(3+nmeds):ydatacol)}.
else.
compute xy={cons, datayed(:,3:ydatacol)}.
end if.
do if (dichy = 1).
compute pt2 =
make(nrow(datayed(:,2)),1,(csum(datayed(:,2))
/n)).
compute LL3 = datayed(:,2)*ln(pt2)+(1-
datayed(:,2))*ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xy),1,0).
compute LL1 = 0.
compute xy22=xy.
loop jjj = 1 to iterate.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.
compute coeff =
bt1+inv(t(xy22)*xy)*t(xy)*(datayed(:,2)-pt1).
compute pt1 = 1/(1+exp(-(xy*coeff))).
compute itprob = csum((pt1 <
.00000000000001) or (pt1 >
.99999999999999)).
do if (itprob = 0).
compute LL = datayed(:,2)*ln(pt1)+(1-
datayed(:,2))*ln(1-pt1).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
break.
end if.
compute bt1 = coeff.
compute LL1 = LL2.
end loop.
do if (jjj >= iterate and iterr = 0).
compute errs = errs+1.

```



```

compute runerrs(errs,1) = 22.
computer iterr = 1.
end if.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)&*pt1&*(1-pt1).
end loop.
compute covmat = inv(t(xy22)*xy).
release xy22.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
end if.
do if (dichy = 0).
compute invXtX = inv(t(xy)*xy).
compute coeff =invXtX*t(xy)*datayed(:,2).
do if (effsize=1 and model > 3 and model < 7
and (ncovs > 0 or clsdmy > 0)).
compute xysdy =
{cons,datayed(:,(4+nmeds):ydatacol)}.
compute coeffsd
=inv(t(xysdy)*xysdy)*t(xysdy)*datayed(:,2).
compute resid=datayed(:,2)-(xysdy*coeffsd).
compute sdycov =sqrt(csum(resid&*resid)/(n-
ncol(xysdy))).
compute xvaron=datayed(:,(3+nmeds)).
compute
coeffx2=inv(t(xysdy)*xysdy)*t(xysdy)*xvaron.
compute residx2=xvaron-(xysdy*coeffx2).
compute sdxcov
=sqrt(csum(residx2&*residx2)/(n-ncol(xysdy))).
end if.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4 and bt > 1).

```

```

compute resid=datayed(:,2)-xy*coeff.
compute sse = cssq(resid).
compute r2full = 1-(sse/sstot).
end if.
do if (bt =1).
compute resid=data(:,2)-xy*coeff.
compute k3 = nrow(coeff).
compute sse = cssq(resid).
compute mse = sse/(n-ncol(xy)).
compute
hcnot=coeff&/(sqrt(diag(mse*invXtX))).
do if (hc3 = 1).
compute h = xy(:,1).
loop i3=1 to n.
compute h(i3,1)= xy(i3,:)*invXtX*t(xy(i3,:)).
end loop.
loop i3=1 to k3.
compute xy(:,i3) = (resid(:,ncol(resid))&/(1-
h))*xy(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xy(:,i3) = sqrt(mse)*xy(:,i3).
end loop.
end if.
compute covmat = (invXtX*t(xy)*xy*invXtX).
end if.
end if.
do if (bt = 1).
do if (model = 2).
compute xy2={cons, datayed(:,3:ydatacol)}.
compute temp = ncol(xy2).
do if (temp > 6).
compute xy3=xy2(:,7:temp)}.
end if.
compute xy2={xy2(:,1:3), xy2(:,5)}.
do if (temp > 6).
compute xy2={xy2, xy3}.
release xy3.

```

```

end if.
compute invXtX = inv(t(xy2)*xy2).
compute coeff2 = invXtX*t(xy2)*datayed(:,2).
compute ssem2 = cssq(datayed(:,2)-xy2*coeff2).
release xy2.
end if.
compute standerr = sqrt(diag(covmat)).
do if (totlp = 1).
compute
mcmats((nmeds+1):ncol(mcmats),(nmeds+1):n
col(mcmats))=covmat(2:(1+nmeds),2:(1+nmed
s)).
end if.
compute standerr=standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1 and totp=2).
compute standerr(1,1)=stddevy/sqrt(n).
end if.
compute coeffplt = coeff.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*coeff)))/dfnum).
compute coeff=coeff(1:(nrow(coeff)-clsdmy),1).
do if (totlp=1).
compute
mccoeff((nmeds+1):nrow(mccoeff))=coeff(2:(1+
nmeds),1).
end if.
compute bbbb=coeff(2,1).
do if totp = 1.
compute deco(1,1)=2+nmeds.
compute deco = deco(1:decoc,1).
compute covdirt = make((nrow(covmat)-
clsdmy),(ncol(covmat)-clsdmy),0).
compute covdirt = covmat(deco,:).

```

```

compute covdir =
make(nrow(covdirt),ncol(covdirt),0).
compute covdir = covdirt(:,t(deco)).
compute deco=coeff(deco,1).
do if (ttt > 0).
compute sedir =
sqrt(diag(directv*covdir*t(directv))).
compute directv = directv*deco.
end if.
compute sobel(:,3)=coeff(2:(1+nmeds),1).
compute sobel(:,4) = standerr(2:(1+nmeds),1).
compute sobel2 = sobel*sobel.
do if (varorder <> 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)*sobel2(:,4)+sobel2(:,3)*sobel2(:,2)).
end if.
do if (varorder = 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)*sobel2(:,4)+sobel2(:,3)*sobel2(:,2)+sobel2(:,2)*sobel2(:,4)).
end if.
compute sobel(:,1) = sobel(:,1)*sobel(:,3).
compute sobel(:,3) = sobel(:,1)/sobel(:,2).
compute sobel(:,4) = 2*(1-
cdfnorm(abs(sobel(:,3)))).
end if.
do if (dichy = 0).
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
do if (ws= 1 and totlp=2).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute dfden=n-1.
end if.
compute cnms = {'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute op = {coeff, standerr, tratio, p}.
end if.
do if (dichy = 1).
compute tratio = (coeff&/standerr).

```

```

compute p = 2*(1-cdfnorm(abs(tratio))).
compute wald = tratio&*tratio.
compute cnms = {'coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute temp=coeff-abs(xp2)*standerr.
compute op = {coeff, standerr, tratio, p, temp}.
compute temp=coeff+abs(xp2)*standerr.
compute op = {op, temp}.
end if.
do if (detail = 1).
do if (totlp = 2).
print yname/title = '*****
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:'/format = A8.
end if.
do if (totlp <> 2).
print yname/title =
'*****
*****'/rlabels = 'Outcome:'/format
= A8.
end if.
end if.
do if (dichy = 1 and bt = 1 and totp = 1).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:'/cnames = nmsd/format = F9.2.
end if.
do if (dichy = 0).
compute r2full = 1-(sse/ssty).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute jndf=dfden.
compute wsdf=dfden.
compute xd = abs(xp2).
compute jncrit = (dfden* (exp((dfden-
(5/6))*((xd/(dfden-
(2/3)+(.11/dfden)))*(xd/(dfden-
(2/3)+(.11/dfden))))-1)).
compute summ = {sqrt(r2full), r2full, mse,
fratio, dfnum, dfden, pfr}.

```

```

compute temp1=coeff-sqrt(jncrit)*standerr.
compute temp2=coeff+sqrt(jncrit)*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (detail = 1).
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels =
'R', 'R-sq', 'MSE' 'F', 'df1', 'df2', 'p'/format =
F10.4.
end if.
end if.
end if.
do if (dichy = 1).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(coeff)-
1)).
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
do if (detail = 1).
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format =
F10.4.
end if.
end if.
do if (totlp = 2).
compute datanmy={'constant';
datanmy((nmeds+2):nrow(datanmy),1)}.
end if.
do if (detail = 1).
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
do if (ws=1).
compute op2=op(1:(1+nmeds),:).

```

```

compute datanmy2 = {'c" path'; bpathnam(
(1+(nmeds>1)): ((1+(nmeds>1))+(nmeds-
1)),1)}.
do if (totlp=2).
compute datanmy2={'c path'}.
compute op2=op(1,:).
compute wsdf=n-1.
end if.
print op2/title = 'Model'/rnames =
datanmy2/cnames = cnms/format = F10.4.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
do if (ws <> 1).
print op/title = 'Model'/rnames =
datanmy/cnames = cnms/format = F10.4.
do if (covcoeff=1).
compute
covmattp=covmat(1:nrow(op),1:nrow(op)).
compute cnamestp=t(datanmy).
print covmattp/title='Covariance matrix of
regression parameter
estimates'/rnames=datanmy/cnames=cnamest
p/format= F10.4.
end if.
end if.
end if.
do if (ttt = 0 and totp = 1).
compute deco = op((nmeds+2),:).
do if (ws = 1).
compute deco=op(1,:).
end if.
end if.
do if (ttt = 0 and totp = 2).
compute decotot = op(2,:).
do if (ws = 1).
compute decotot=op(1,:).
end if.
end if.
do if (nmods > 0 and model > 4 and detail = 1
and nrow(yintkey) > 1)).

```

```

print yintkey/title = 'Product terms key: '/format
= A8.
end if.
do if (nmods > 0 and model < 4 and detail = 1).
print yintkey2/title = 'Product terms key: '/format
= A8.
do if ((model = 1 or model = 2) and dichy = 0).
compute temp={((op(4,3)**2)*(1-r2full))/dfden,
op(4,3)**2, 1, dfden, op(4,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(4,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(2,1).
do if (model = 2).
compute temp={temp;((op(6,3)**2)*(1-
r2full))/dfden, op(6,3)**2, 1, dfden, op(6,4)}.
do if (hc3=1).
compute temp(2,1)=((hcnot(6,1)**2)*(1-
r2full))/dfden.
end if.
compute frat2=(dfden*(r2full-(1-
(ssem2/ssty))))/(2*(1-r2full)).
do if (hc3=1).
compute lmat=make(nrow(coeffplt),2,0).
compute lmat(4,1)=1.
compute lmat(6,2)=1.
compute frat2 =
(t(t(lmat)*coeffplt)*inv(t(lmat)*covmat*lmat)*((t(l
mat)*coeffplt)))/2).
end if.
compute temp={temp;(r2full-(1-
(ssem2/ssty))),frat2,2,dfden,1-
fcd(frat2,2,dfden)}.
compute rnms={rnms; yintkey2(3,1),'Both'}.
end if.
print temp/title = 'R-square increase due to
interaction(s): '/rnames=rnms/clabels = 'R2-
chng', 'F', 'df1', 'df2', 'p'/format = F10.4.
end if.

```



```

do if (model = 3 and dichy = 0).
compute temp={((op(8,3)**2)*(1-r2full))/dfden,
op(8,3)**2, dfden, op(8,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(8,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(5,1).
print temp/title = 'R-square increase due to
three-way interaction:.'/rnames=rnms/clabels =
'R2-chng', 'F(1,df2)', 'df2', 'p'/format = F10.4.
end if.
end if.
end if.
do if (model = 6 and totlp = 1).
compute
mmpaths(nrow(mmpaths),1)=coeff(nrow(mmpa
ths),1).
compute
mmpaths(nrow(mmpaths),(2:(nmeds+1)))=t(co
eff(2:(nmeds+1),1)).
end if.
do if (totlp = 1).
loop im = 1 to nmeds.
do if (model < 4).
compute ymat(1,im) = coeff(3,1)*(1-yes).
compute ymat(2,im) = coeff(4,1).
compute cmat(1,im) = covmat(3,3).
compute cmat(2,im) = covmat(4,4).
compute cmat(5,im) = covmat(3,4).
compute jnb1=coeff(3,1).
compute jnb3=coeff(4,1).
compute jnsb1=covmat(3,3).
compute jnsb3=covmat(4,4).
compute jnsb1b3=covmat(3,4).
do if (model = 2 or model = 3).
compute ymat(3,im) = coeff(6,1).
compute cmat(3,im) = covmat(6,6).
compute cmat(6,im) = covmat(3,6).
compute cmat(8,im) = covmat(4,6).

```

```

end if.
do if (model = 3).
compute ymat(4,im) = coeff(8,1).
compute cmat(4,im) = covmat(8,8).
compute cmat(7,im) = covmat(3,8).
compute cmat(9,im) = covmat(4,8).
compute cmat(10,im) = covmat(6,8).
compute jnb1=coeff(4,1).
compute jnb3=coeff(8,1).
compute jnsb1=covmat(4,4).
compute jnsb3=covmat(8,8).
compute jnsb1b3=covmat(4,8).
end if.
end if.
do if (model > 3).
compute ymat(5,im) = coeff((1+im),1).
end if.
do if (xmy = 1).
compute ymat(6,im) = coeff((2+nmeds+im),1).
end if.
do if (vy = 1).
compute ymat(6,im) = coeff((3+nmeds+im),1).
end if.
do if (qy = 1 and vy = 1).
compute ymat(6,im) = coeff((5+nmeds+((im-1)*2)),1).
compute ymat(7,im) = coeff((6+nmeds+((im-1)*2)),1).
end if.
do if (vqy = 1).
compute ymat(6,im) = coeff((6+nmeds+((im-1)*3)),1).
compute ymat(7,im) = coeff((7+nmeds+((im-1)*3)),1).
compute ymat(8,im) = coeff((8+nmeds+((im-1)*3)),1).
end if.
do if (wmy = 1).
compute ymat(7,im) = coeff((3+nmeds+im-wy),1).

```

```

end if.
do if (wmy = 1 and vy = 1).
compute ymat(7,im) =coeff((4+(nmeds*2)+im-
wy),1).
end if.
do if (wmy = 1 and vy = 1 and wvmy = 1).
compute ymat(7,im) =
coeff((6+(nmeds*2)+((im-1)*2)-wy),1).
compute ymat(8,im) =
coeff((7+(nmeds*2)+((im-1)*2)-wy),1).
end if.
do if (wmy = 1 and zmy = 1).
compute ymat(6,im) = coeff((6-(wzy*3)+(wzm-
1)+(nmeds*2)+((im-1)*2)*wzm)+((im-1)*(1-
wzm))-((zy-wzy)*2),1).
compute ymat(7,im) = coeff((3-
zy+im+nmeds),1).
do if (wzmy = 1).
compute ymat(8,im) = coeff((7-
(wzy*3)+(nmeds*2)+((im-1)*2)),1).
end if.
end if.
do if (nmods > 0 and model <> 5).
loop indlp = 1 to nrow(modvals).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp)).
do if (model > 6).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp))*csum(ym
at(5:8,im)*vmat(5:8,indlp)).
end if.
end loop.
compute indboot((bt+(im-
1)*(boot+1)),:)=t(indeff).
do if (imm =1 or imm = 3).
compute
indbootp(bt,im)=ymat(2,im)*ymat(5,im).
do if (imm=3).

```

```

compute
indbootp(bt,(im+nmeds))=ymat(3,im)*ymat(5,i
m).
do if (zdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxz-cminz).
end if.
end if.
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
end if.
end if.
do if ((imm= 2 or imm = 4) or model = 74).
compute
indbootp(bt,im)=ymat(1,im)*ymat(6,im).
do if (imm=4).
compute
indbootp(bt,(im+nmeds))=ymat(1,im)*ymat(7,i
m).
do if (qdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxq-cminq).
end if.
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
end if.
end if.
do if (imm=7).
compute
indbootp(bt,im)=ymat(2,im)*ymat(6,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y

```

```

mat(2,im)*ymat(5,im)*onesmv2)+(ymat(2,im)*y
mat(6,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(6,im)*onesmv1)+(ymat(2,im)*y
mat(6,im)*t(immmv1)).
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxv-cminv).
end if.
end if.
do if (imm=5).
compute
indbootp(bt,im)=ymat(4,im)*ymat(5,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(4,im)*y
mat(5,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(3,im)*ymat(5,im)*onesmv1)+(ymat(4,im)*y
mat(5,im)*t(immmv1)).
do if (wdich=1).

```

```

compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).
end if.
do if (zdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxz-
cminz).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxz-cminz).
end if.
end if.
do if (imm=6).
compute
indbootp(bt,im)=ymat(1,im)*ymat(8,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(1,im)*ymat(6,im)*onesmv2)+(ymat(1,im)*y
mat(8,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(7,im)*onesmv1)+(ymat(1,im)*y
mat(8,im)*t(immmv1)).
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxv-cminv).

```

```

end if.
do if (qdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxq-
cminq).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxq-cminq).
end if.
end if.
do if ((model = 58 or model = 59) and
wvdich=1).
compute
indbootp(bt,im)=(ymat(1,im)*ymat(7,im)*(cmax
w-cminw)+(ymat(2,im)*ymat(5,im)*(cmaxw-
cminw)+(ymat(2,im)*ymat(7,im)*((cmaxw*cma
xw)-(cminw*cminw))).
end if.
end if.
do if (model = 4 or model = 5).
compute
indboot(bt,im)=csum(ymat(1:4,im)*vmat(1:4,1
))*csum(ymat(5:8,im)*vmat(5:8,1)).
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.000000000000001.
end if.
compute sumind=rsum(indboot(bt,:)).
do if (im=nmeds).
compute
pmeff(bt,2:(im+1))=indboot(bt,:)/(sumind+coeff(
(2+nmeds),1)).
end if.
compute
rmeff(bt,(im+1))=indboot(bt,im)/coeff((2+nmeds
),1).
compute
abpseff(bt,(im+1))=indboot(bt,im)/stddevy.

```

```

compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*stddevx.
do if (ncovs > 0 or clsdmy > 0).
compute
abpseff(bt,(im+1))=indboot(bt,im)/sdycov.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*sdxcov.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,1) = r2my-(r2full-r2xy).
compute abmmr = 1.
do if (indboot(bt,im) < 0).
compute abmmr = cmin(abmm).
end if.
do if (indboot(bt,im) > 0).
compute abmmr = cmax(abmm).
end if.
compute kappa2(bt,1)=indboot(bt,im)/abmmr.
compute tmp = indboot(bt,im)/abmmr.
end if.
end if.
do if (model = 6).
do if (nmeds = 2).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(4,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3).
compute
indboot(bt,3)=mmpaths(3,1)*mmpaths(4,3).
end if.
do if (nmeds = 3).
compute indboot(bt,1) =
mmpaths(2,1)*mmpaths(5,2).
compute indboot(bt,2) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3).
compute indboot(bt,3) =
mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4).

```



```

compute indboot(bt,4) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*m
mpaths(5,4).
compute indboot(bt,5) =
mmpaths(3,1)*mmpaths(5,3).
compute indboot(bt,6) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4).
compute indboot(bt,7) =
mmpaths(4,1)*mmpaths(5,4).
end if.
do if (nmeds = 4).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(6,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mm
paths(6,3).
compute
indboot(bt,3)=mmpaths(2,1)*mmpaths(4,2)*mm
paths(6,4).
compute
indboot(bt,4)=mmpaths(2,1)*mmpaths(5,2)*mm
paths(6,5).
compute
indboot(bt,5)=mmpaths(2,1)*mmpaths(3,2)*mm
paths(4,3)*mmpaths(6,4).
compute
indboot(bt,6)=mmpaths(2,1)*mmpaths(3,2)*mm
paths(5,3)*mmpaths(6,5).
compute
indboot(bt,7)=mmpaths(2,1)*mmpaths(4,2)*mm
paths(5,4)*mmpaths(6,5).
compute
indboot(bt,8)=mmpaths(2,1)*mmpaths(3,2)*mm
paths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,9)=mmpaths(3,1)*mmpaths(6,3).
compute
indboot(bt,10)=mmpaths(3,1)*mmpaths(4,3)*m
mpaths(6,4).

```

```

compute indboot(bt,11)
=mmpaths(3,1)*mmpaths(5,3)*mmpaths(6,5).
compute indboot(bt,12) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4)*m
mpaths(6,5).
compute indboot(bt,13) =
mmpaths(4,1)*mmpaths(6,4).
compute indboot(bt,14) =
mmpaths(4,1)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,15) =
mmpaths(5,1)*mmpaths(6,5).
end if.
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.00000000000001.
end if.
compute
pmeff(bt,:)=indboot(bt,:)/(rsum(indboot(bt,:))+m
mpaths(nrow(mmpaths),1)).
compute rmeff(bt,:) =
indboot(bt,:)/mmpaths(nrow(mmpaths),1).
compute abpseff(bt,:)=indboot(bt,:)/stddevy.
compute
abcseff(bt,:)=(stddevx*indboot(bt,:))/stddevy.
do if (ncovs > 0 or clsdmy > 0).
compute abpseff(bt,:)=indboot(bt,:)/sdycov.
compute
abcseff(bt,:)=(sdxcov*indboot(bt,:))/sdycov.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,:) = r2my-(r2full-r2xy).
end if.
end if.
end if.
end loop.
end if.
end loop.
do if (savboot = 1 and bt > 1).
compute bootstrp((bt-1),:)=bootcoef.

```

```

end if.
end loop.
do if (savboot = 1).
compute bootstrp = bootstrap(:,2:ncol(bootstrap)).
save bootstrp/outfile = *.
end if.
release datayed, dat, datamed, data, y, x,
datatm, dataty, mints2, cons, mnv, tmp.
do if (mc > 0).
compute x1 = sqrt(-
2*ln(uniform(mc,nrow(mcmats))))&*cos((2*3.14
159265358979)*uniform(mc,nrow(mcmats))).
compute x1=x1*chol(mcmats).
loop i=1 to nrow(x1).
compute x1(i,:)=x1(i,:)+t(mccoeff).
end loop.
loop i = 1 to nmeds.
compute
indboot(2:nrow(indboot),i)=x1(:,i)&*x1(:,(i+nme
ds)).
end loop.
end if.
do if (ttt = 0 and model > 3).
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
end if.
do if (model <> 74).
do if (dichy = 0).
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 't', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 't', 'p', 'LLCI', 'ULCI'/format =
F10.4.

```

```

else.
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 'Z', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 'Z', 'p', 'LLCI', 'ULCI' /format =
F10.4.
end if.
end if.
end if.
do if (ttt > 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
compute clbs = {modvnm2, 'Effect', 'SE', 't', 'p',
'LLCI', 'ULCI'}.
compute tratio = (directv&/sedir).
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
compute outp = {modvalsd, directv, sedir,
tratio, p}.
do if (dichy = 0).
compute temp1=directv-sqrt(jncrit)*sedir.
compute temp2=directv+sqrt(jncrit)*sedir.
compute outp = {outp, temp1, temp2}.
end if.
do if (dichy = 1).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute temp = directv-abs(xp2)*sedir.
compute outp = {outp, temp}.
compute temp = directv+abs(xp2)*sedir.
compute outp = {outp, temp}.
compute clbs = {modvnm2, 'Effect', 'SE', 'Z', 'p',
'LLCI', 'ULCI'}.
end if.
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.

```

```

print outp/title = 'Conditional direct effect(s) of
X on Y at values of the moderator(s)://format =
F10.4 /cnames = clbs.
end if.
do if (nmods > 0 and model <> 5).
do if (model < 4).
print/title =
*****
*****!
compute zmat(1,1) = 1*(1-yes).
compute cfse = make(nrow(modvals),1,0).
loop #m = 1 to nrow(modvals).
do if (model = 1).
compute zmat(2,1)=modvals(#m,1)**2.
compute zmat(5,1)=2*modvals(#m,1).
end if.
do if (model = 2 or model = 3).
compute zmat(2,1)=modvals(#m,2)**2.
compute zmat(3,1)=modvals(#m,1)**2.
compute
zmat(4,1)=(modvals(#m,1)**2)*(modvals(#m,2)
**2)*(1-yes2).
compute zmat(5,1)=2*modvals(#m,2)*(1-yes).
compute zmat(6,1)=2*modvals(#m,1)*(1-yes).
compute
zmat(7,1)=2*modvals(#m,1)*modvals(#m,2)*(1-
yes).
compute
zmat(8,1)=2*modvals(#m,1)*modvals(#m,2).
compute
zmat(9,1)=2*modvals(#m,1)*(modvals(#m,2)**
2)*(1-yes2).
compute
zmat(10,1)=2*(modvals(#m,1)**2)*modvals(#m
,2)*(1-yes2).
end if.
compute cfse(#m,1)=sqrt(csum(zmat&*cmat)).
end loop.
end if.
do if (nmods > 0).

```

```

compute clbs = {modvnm, 'Effect'}.
loop im = 1 to nmeds.
compute obs = t(indboot(1+(im-1)*(boot+1),:)).
compute outp = {modvals, obs}.
do if (model < 4).
compute tstat = obs&/cfse.
do if (dichy = 0).
compute pval = 2*(1-tcdf(abs(tstat), (n-
ncol(xy)))).
compute temp=obs-sqrt(jncrit)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp=obs+sqrt(jncrit)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 't', 'p', 'LLCI', 'ULCI'}.
compute jnclbs=clbs.
end if.
do if (dichy = 1).
compute pval = 2*(1-cdfnorm(abs(tstat))).
compute temp = obs-abs(xp2)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp = obs+abs(xp2)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute jnclbs=clbs.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indboot((1+(im-
1)*(boot+1)),:).
compute indboot2 = indboot((2+(im-
1)*(boot+1)):(1+(im-1)*(boot+1)+boot),:).
compute mnind = t(csum(indboot2)/boot).
compute stdind=t(sqrt((cssq(indboot2)-
((csum(indboot2)**2)/boot))/(boot-1))).
compute llci=make(1,ncol(indboot2),-999).
compute ulci=make(1,ncol(indboot2),-999).
loop #e = 1 to ncol(indboot2).

```

```

.
compute temp = indboot2(:,#e).
compute temp(GRADE( indboot2(:,#e) )) =
indboot2(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).

```

```

compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {modvals, obs, stdind, t(llci),
t(ulci)}.
compute clbs = {modvnm, 'Effect', 'Boot SE',
'BootLLCI', 'BootULCI'}.
end if.
compute mtemp = mnames(1, im).
compute rlbs =
make(nrow(modvals),1,mnames(1,im)).
do if (model < 4).
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
do if (yes=0).
print outp/title = 'Conditional effect of X on Y at
values of the moderator(s):'/cnames =
clbs/format = F10.4.
else.
compute outp=outp(:,3:ncol(outp)).
compute clbs=clbs(:,3:ncol(clbs)).
compute clbs(1,1)='Contrast'.
print outp/title = 'Contrast of conditional effects
of X on Y'/cnames = clbs/format = F10.4.
end if.
end if.
do if (model > 5 and mod74dic <> 1).

```



```

do if (im = 1).
print/title = 'Conditional indirect effect(s) of X on
Y at values of the moderator(s):'.
end if.
print outp/title = 'Mediator'/rnames =
rlbs/cnames = clbs/format = F10.4.
else if (mod74dic = 1 and model = 74).
do if (im = 1).
print/title = 'Indirect effect(s) of X on Y:'.
compute clbs3=clbs(1,2:ncol(clbs)).
end if.
compute outp3=outp(1,2:ncol(outp)).
print outp3/title = 'Mediator'/rnames =
rlbs/cnames = clbs3/format = F10.4.
end if.
end loop.
loop i = notes to 1 by -1.
do if (note(i,1) = 4 and yes=0).
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 5 and yes=0).
print/title = 'Values for quantitative moderators
are the mean and plus/minus one SD from
mean.'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 14 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    below the mean was replaced
with the minimum because one SD below the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.

```

```

do if (note(i,1) = 15 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    above the mean was replaced
with the maximum because one SD above the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.
end loop.
do if (model = 3 and yes=0).
compute jnvals=make(nrow(matw),7,0).
compute jnvals(:,1)=matw.
compute jnvals(:,2)=jnb1+jnb3*jnvals(:,1).
compute
jnvals(:,3)=sqrt(jnsb1+2*jnvals(:,1)*jnsb1b3+(jn
vals(:,1)*jnvals(:,1))*jnsb3).
compute jnvals(:,4)=jnvals(:,2)/jnvals(:,3).
do if (dichy = 0).
compute jnvals(:,5)=2*(1-tcdf(abs(jnvals(:,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(:,5)=2*(1-
cdfnorm(abs(jnvals(:,4)))).
end if.
compute jnvals(:,6)=jnvals(:,2)-
sqrt(jncrit)*jnvals(:,3).
compute
jnvals(:,7)=jnvals(:,2)+sqrt(jncrit)*jnvals(:,3).
compute clbs={clbs(:,1),clbs(:,3:ncol(clbs))}.
do if (coeffci = 0).
compute jnvals =jnvals(:,1:(ncol(jnvals)-2)).
end if.
print jnvals/title = 'Conditional effect of X*M
interaction at values of W: '/cnames
=clbs/format = F10.4.
end if.
do if (jn = 1 and (model = 1 or model = 3) and
jndich = 0 and yes=0).

```

```

print/title = '***** JOHNSON-
NEYMAN TECHNIQUE
*****'

compute ajn =(jncrit*jnsb3)-(jnb3*jnb3).
compute bjn = 2*((jncrit*jnsb1b3)-(jnb1*jnb3)).
compute cjn = (jncrit*jnsb1)-(jnb1*jnb1).
compute radarg = (bjn*bjn)-(4*ajn*cjn).
compute den = 2*ajn.
compute nrts = 0.
do if (radarg >= 0 and den <> 0).
compute x21 = (-bjn+sqrt(radarg))/den.
compute x22 = (-bjn-sqrt(radarg))/den.
compute roots = 0.
do if (x21 >= jnmin and x21 <= jnmax).
compute nrts = 1.
compute roots = {roots; x21}.
end if.
do if (x22 >= jnmin and x22 <= jnmax).
compute nrts = nrts + 1.
compute roots = {roots; x22}.
end if.
compute roots={roots,make(nrow(roots),2,0)}.
compute modtemp=m.
do if (model=3).
compute modtemp=w.
end if.
do if (nrts > 0).
compute roots = roots(2:nrow(roots),1:3).
compute roots(1,2)=(csum(modtemp <
roots(1,1))/n)*100.
compute roots(1,3)=(csum(modtemp >
roots(1,1))/n)*100.
do if (nrow(roots)=2).
compute roots(2,2)=(csum(modtemp <
roots(2,1))/n)*100.
compute roots(2,3)=(csum(modtemp >
roots(2,1))/n)*100.
end if.
print roots/title = 'Moderator value(s) defining
Johnson-Neyman significance

```

```

region(s)/clabels = 'Value', '% below', '%
above'/format = F10.4.
compute jnvals=make((21+nrts),7,0).
loop i= 0 to 20.
compute jnvals((i+1),1)=jnmin+(i*((jnmax-
jnmin)/20)).
end loop.
loop i = 1 to nrts.
loop j = 2 to nrow(jnvals).
do if ((roots(i,1) > jnvals((j-1),1)) and (roots(i,1)
< jnvals(j,1))).
compute
jnvals((j+1):(21+i),1)=jnvals(j:(20+i),1).
compute jnvals(j,1)=roots(i,1).
end if.
end loop.
end loop.
loop i = 1 to nrow(jnvals).
compute jnvals(i,2)=jnb1+jnb3*jnvals(i,1).
compute
jnvals(i,3)=sqrt(jnsb1+2*jnvals(i,1)*jnsb1b3+(jn
vals(i,1)*jnvals(i,1))*jnsb3).
compute jnvals(i,4)=jnvals(i,2)/jnvals(i,3).
do if (dichy = 0).
compute jnvals(i,5)=2*(1-tcdf(abs(jnvals(i,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(i,5)=2*(1-
cdfnorm(abs(jnvals(i,4)))).
end if.
compute jnvals(i,6)=jnvals(i,2)-
sqrt(jncrit)*jnvals(i,3).
compute
jnvals(i,7)=jnvals(i,2)+sqrt(jncrit)*jnvals(i,3).
end loop.
do if (model = 1).
print jnvals/title = 'Conditional effect of X on Y
at values of the moderator (M)'/cnames
=jnclbs/format = F10.4.

```

```

end if.
do if (model = 3).
compute
jnclbs={jnclbs(:,1),jnclbs(:,3:ncol(jnclbs))}.
print jnclbs/title = 'Conditional effect of X*M on
Y at values of the moderator (W)'/cnames
=jnclbs/format = F10.4.
end if.
end if.
do if (nrts = 0).
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
else.
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
end if.
end if.
do if (model < 4 and (plot = 1)).
compute dataplot =
make((nrow(modvals)*nrow(matx)),(ncol(modv
als)+1),0).
compute tmp = 1.
loop i = 1 to nrow(modvals).
loop j = 1 to nrow(matx).
compute dataplot(tmp,:)=matx(j,1),
modvals(i,:)}.
compute tmp=tmp+1.
end loop.
end loop.
compute dataplot = {dataplot,
make(nrow(dataplot),(1+dichy),0)}.
compute dataplo2 = make(nrow(dataplot),1,1).
do if (model = 1).
compute dataplo2 = {dataplo2, dataplot(:,2),
dataplot(:,1), (dataplot(:,1)*dataplot(:,2))}.
end if.

```

```

do if (model = 2 or model = 3).
compute dataplo2 = {dataplo2, dataplot(:,3),
dataplot(:,1), (dataplot(:,1)*dataplot(:,3)),
dataplot(:,2), (dataplot(:,1)*dataplot(:,2))}.
do if (model = 3).
compute dataplo2 = {dataplo2,
(dataplot(:,2)*dataplot(:,3)),
(dataplot(:,1)*dataplot(:,2)*dataplot(:,3))}.
end if.
end if.
loop i = 1 to nrow(dataplot).
compute tmp=dataplo2(i,:).
do if (ncovs > 0).
compute tmp = {tmp, covmeans}.
end if.
do if (cluster > 0).
compute tmp = {tmp, cldmeans}.
end if.
compute dataplot(i,(ncol(dataplot)-
(dichy)))=tmp*coeffplt.
do if (dichy = 1).
compute
dataplot(i,(ncol(dataplot)))=exp(tmp*coeffplt)/(1
+exp(tmp*coeffplt)).
end if.
end loop.
compute clbs = {xname, modvnm, 'yhat'}.
do if (dichy = 1).
compute clbs = {xname, modvnm, 'ln(odds)',
'prob'}.
end if.
print/title =
*****
*****!

```

```

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (model = 1).
do if (dichy = 0).
print/title = 'DATA LIST FREE/Pro_Env_Strgy
Technological_Advan
Operational_Performance.' /space=1.
else if (dichy = 1).
print/title = 'DATA LIST FREE/Pro_Env_Strgy
Technological_Advan logodds prob.'
/space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH Operational_Performance BY
Technological_Advan.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva

```

```

n WITH Operational_Performance BY
Pro_Env_Strgy.' /space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH logodds BY Technological_Advan.'
/space=0.
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH prob BY Technological_Advan.'
/space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Pro_Env_Strgy.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Pro_Env_Strgy.' /space=0.
end if.
end if.
end if.
do if (model = 2 or model = 3).
do if (dichy = 0).
print/title = 'DATA LIST FREE/Pro_Env_Strgy
Technological_Advan
Operational_Performance.' /space=1.
end if.
do if (dichy = 1).
print/title = 'DATA LIST FREE/Pro_Env_Strgy
Technological_Advan logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' /format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich=0).

```



```

print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH Operational_Performance BY
Technological_Advan/PANEL ROWVAR='
/space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH Operational_Performance BY
Pro_Env_Strgy/PANEL ROWVAR=' /space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH logodds BY
Technological_Advan/PANEL ROWVAR='
/space=0.
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH prob BY Technological_Advan/PANEL
ROWVAR=' /space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Pro_Env_Strgy/PANEL
ROWVAR=' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Pro_Env_Strgy/PANEL
ROWVAR=' /space=0.
end if.
end if.
end if.
do if (ncovs > 0).
print/title = '* Estimates are based on setting
covariates to their sample means.'.

```

```

end if.
end if.
end if.
do if ((imm > 0) or (model = 74 and
mod74dic=0) or ((model = 58 or model = 59)
and wvdich=1))).
do if (imm > 4 and imm < 8).
compute
indbootp={indbootp(:,1:nmeds),indboop1,indbo
op2}.
end if.
compute obsprod = t(indbootp(1,:)).
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indbootp(1,:).
compute indbootp = indbootp(2:(boot+1),:).
compute mnindp = t(csum(indbootp)/boot).
compute stdindp=t(sqrt((cssq(indbootp)-
((csum(indbootp)**2)/boot))/(boot-1))).
compute llcip=make(1,ncol(indbootp),-999).
compute ulcip=make(1,ncol(indbootp),-999).
loop #e = 1 to ncol(indbootp).

.
compute temp = indbootp(:,#e).
compute temp(GRADE( indbootp(:,#e) )) =
indbootp(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).

```

```

compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcip(1,#e)=llcit.
compute ulcip(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.

```

```

compute outp = {obsprod, stdindp, t(llcip),
t(ulcip)}.
compute clbs = {'Effect', 'SE(Boot)', 'BootLLCI',
'BootULCI'}.
do if (model = 8 or model = 12).
print/title='-----'.
compute outpimm=outp(1:nmeds,:).
print/title = 'Indirect effect of highest order
product: '/space=0.
print outpimm/title = 'Mediator'/cnames =
clbs/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
compute clbs3 = {'Index', 'SE(Boot)',
'BootLLCI', 'BootULCI'}.
do if (imm < 5).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator: '/format=A8.
end if.
compute outpimm=outp((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.

```

```

print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'.
compute outpimm=outp(1:nmeds,:).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
compute
outpimm=outp((nmeds+1):nrow(outp),:).
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),clbs3}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3(1,1)=modvnm(1,1).
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator: '/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.

```

```

end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
end if.
do if (boot = 0).
do if (model = 8 or model = 12).
print/title='-----'.
compute obsimm=obsprod(1:nmeds,:).
print/title = 'Indirect effect of highest order
product:.'/space=0.
print obsimm/title = 'Mediator'/clabels =
'Effect'/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
do if ((imm > 0 and imm < 5) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=obsprod((((k4-
1)*nmeds)+1):(nmeds*k4),:).

```

```

print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
end loop.
end if.
do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'.
compute outpimm=obsprod(1:nmeds,1).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
compute
outpimm=obsprod((nmeds+1):nrow(obsprod),:)
.
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),'Index'}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3={modvnm(1,1),'Index'}.
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.

```

```

print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
end if.
end if.
end if.
compute conmake=0.
compute concols = 0.
do if ((model > 3 and model < 7) and (contrast
> 0) and nmods = 0 and nmeds > 1 ).
compute concols =
((ncol(indboot)*(ncol(indboot)-1))/2).
compute indcon=make(nrow(indboot),concols,-
999).
compute conkey = { ' ', ' ', ' ', ' ' }.
compute temp=1.
compute conmake=1.
loop i = 1 to (ncol(indboot)-1).
loop j = (i+1) to (ncol(indboot)).
compute indcon(:,temp)=indboot(:,i)-
indboot(:,j).
do if (contrast = 2).
compute indcon(:,temp)=abs(indboot(:,i))-
abs(indboot(:,j)).
end if.
do if (model <> 6).
compute conkey={conkey; mnames(1,i),
'minus', mnames(1,j)}.
end if.
do if (model = 6).
compute conkey={conkey; indlbl2(i,1), 'minus',
indlbl2(j,1)}.
end if.
compute temp=temp+1.
end loop.
end loop.
end if.
do if (model = 4 or model = 5).
compute clbs = {'Effect'}.

```



```

compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obs = {csum(obs); obs}.
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute rlbs = {rlbs;
cntname(1:ncol(indcon),1)}.
end if.
compute outp = obs.
compute outp2=outp.
do if (effsize = 1 and dichy = 0).
compute
pmeff(:,1)=rsum(pmeff(:,2:ncol(pmeff))).
compute
rmeff(:,1)=rsum(rmeff(:,2:ncol(rmeff))).
compute
abpseff(:,1)=rsum(abpseff(:,2:ncol(abpseff))).
compute
abcseff(:,1)=rsum(abcseff(:,2:ncol(abcseff))).
compute eff = {pmeff, rmeff, abpseff, abcseff}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute eff = {eff, r245}.
compute r245obs = {r245(1,1);r245(1,1)}.
compute kappa2ob = kappa2(1,1).
end if.
compute pmobs = t(pmeff(1,1:(nmeds+1))).
compute rmobs = t(rmeff(1,1:(nmeds+1))).
compute psobs = t(abpseff(1,1:(nmeds+1))).
compute csobs = t(abcseff(1,1:(nmeds+1))).
do if (contrast = 0).
compute outp2 = {obs, psobs, csobs, pmobs,
rmobs}.
end if.
do if (contrast > 0).
compute obs2=obs(1:nrow(psobs),:).
compute outp2 = {obs2, psobs, csobs, pmobs,
rmobs}.
end if.

```

```

compute clbs = {'ab', 'ab_ps', 'ab_cs', 'ab/c',
'ab/c'}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute outp2 = {outp2, r245obs}.
compute clbs = {clbs, 'R-sq_med'}.
do if (boot = 0 and mc = 0).
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
do if (boot = 0 and mc = 0).
do if (nmeds = 1).
compute outp2 = outp2(2,:).
compute rlbs = rlbs(2,1).
end if.
do if (model=5).
compute outp2=outp2(:,1:3).
end if.
print outp2/title = 'Indirect effect(s) of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1 and nmeds >
1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0 or mc > 0).
compute indboot = {rsum(indboot), indboot}.
compute bootasz=boot.
do if (mc > 0).
compute bootasz=mc.
end if.
compute ones = make(bootasz,1,1).
do if (conmake = 1).

```

```

compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(bootsz+1),:).
compute mnind = t(csum(indboot)/bootsz).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/bootsz))/(bootsz-1))).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).

.

compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.

```

```

end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
do if (effsize = 1 and dichy = 0 and boot > 0).
compute estmte=eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).
.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.

```

```

do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.

```

```

do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
end if.
end if.
do if (boot > 0 or mc > 0).
compute outp = {obs, stdind, t(llci), t(ulci)}.
do if (nmeds = 1).
compute outp = outp(2,:).
compute rlbs = rlbs(2,1).
end if.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
do if (mc > 0).
compute clbs = {'Effect', 'MC SE', 'MC LLCI',
'MC ULCI'}.
end if.
print outp/title = 'Indirect effect of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (dichy = 0 and effsize = 1 and boot > 0).
compute outp = {psobs,
stdindf((2*(nmeds+1)+1):(3*(nmeds+1)),1),
t(llcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))),
t(ulcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Partially standardized indirect
effect of X on Y'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute outp = {csobs,
stdindf((3*(nmeds+1)+1):(4*(nmeds+1)),1),

```

```

t(llcif(1,(3*(nmeds+1)+1):(4*(nmeds+1)))),
t(ulcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))}.
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Completely standardized
indirect effect of X on Y'/rnames = rlbs/cnames
= clbs/format = F10.4.
compute outp = {pmobs,
stdindf(1:(nmeds+1),1),t(llcif(1,1:(nmeds+1))),
t(ulcif(1,1:(nmeds+1))))}.
do if (model = 4).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
compute outp = {rmobs, stdindf(((nmeds+1)+1)
:(2*(nmeds+1)) ,1),
t(llcif(1,((nmeds+1)+1):(2*(nmeds+1)))),
t(ulcif(1,((nmeds+1)+1):(2*(nmeds+1))))}.
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
do if (nmeds = 1 and cluster = 0 and ncovs = 0
and model = 4).
compute r245obs = r245obs(1,1).
compute outp = {r245obs,
stdindf((4*(nmeds+1)+1):(4*(nmeds+1)+1),1),
t(llcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1))),
t(ulcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1))))}.
print outp/title = 'R-squared mediation effect
size (R-sq_med)'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute note(notes,1) = 32.
compute notes = notes + 1.

```

```

end if.
end if.
end if.
end if.
do if (normal = 1).
do if (nmeds = 1).
print sobel/title = 'Normal theory tests for
indirect effect'/clabels = 'Effect', 'se', 'Z',
'p'/format = F10.4.
end if.
do if (nmeds > 1).
compute rlbs2 = rlbs(2:nrow(rlbs),1).
print sobel/title = 'Normal theory tests for
specific indirect effects'/rnames = rlbs2/clabels
= 'Effect', 'se', 'Z', 'p'/format = F10.4.
end if.
end if.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
do if (model = 6).
compute clbs = {'eff'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obsfsum=csum(obs).
compute obs = {obsfsum; obs}.
compute indlbl = indlbl(1:nrow(obs),1).
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute indlbl = {indlbl;
cntname(1:ncol(indcon),1)}.
end if.
compute obs2=obs.
do if (boot = 0).
do if (dichy = 0 and effsize = 1).

```



```

compute clbs = {'eff', 'eff_ps', 'eff_cs', 'eff/c',
'eff/c'}.
do if (ncovs = 0).
compute obs = {obs, obs/stddevy,
(obs*stddevx/stddevy), obs/ctot,
obs/mmpaths(nrow(mmpaths),1)}.
end if.
do if (ncovs > 0 or clsdmy > 0).
compute obs = {obs, obs/sdy cov,
(obs*sdx cov/sdy cov),
obs/(obsfsum+mmpaths(nrow(mmpaths),1)),
obs/mmpaths(nrow(mmpaths),1)}.
end if.
compute obs2=obs.
do if (contrast > 0).
compute obs2 = obs(1:(nrow(obs)-concols),:).
end if.
end if.
print obs2/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute indboot = {rsum(indboot), indboot}.
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(boot+1),:).
compute mnind = t(csum(indboot)/boot).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/boot))/(boot-1))).

```

```

compute temp = nrow(indboot).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).

.

compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.

```

```

compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute obs = {obs, stdind, t(llci), t(ulci)}.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
print obs/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (effsize = 1 and dichy = 0).
compute indboot=indboot(:,1:(ncol(indboot)-
concols)).
compute eff = {rsum(abpseff), abpseff,
rsum(abcseff), abcseff, rsum(pmeff), pmeff,
rsum(rmeff), rmeff}.
compute effobs = eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).

```

```

.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(effobs(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).

```

```

compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute temp2 = stdindf(1:ncol(indboot),1).
compute temp3 = effobs(:,1:ncol(indboot)).
compute templow = llcif(1,1:ncol(indboot)).
compute temphi = ulcif(1,1:ncol(indboot)).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Partially standardized indirect
effect of X on Y'/cnames = clbs/rnames =
indlbl/format = F10.4.
compute temp2 =
stdindf((ncol(indboot)+1):(2*ncol(indboot)),1).
compute temp3 =
effobs(:,(ncol(indboot)+1):(2*ncol(indboot))).
compute templow =
llcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute temphi =
ulcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Completely standardized
indirect effect of X on Y'/cnames = clbs/rnames
= indlbl/format = F10.4.
compute temp2 =
stdindf((2*(ncol(indboot))+1):(3*ncol(indboot)),1
).

```

```

compute temp3 =
effobs(:,(2*(ncol(indboot))+1):(3*ncol(indboot)))
.
compute templow =
llcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute temphi =
ulcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
compute temp =
eff(:,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temp2 =
stdindf((3*(ncol(indboot))+1):(4*ncol(indboot)),1
).
compute temp3 =
effobs(:,(3*(ncol(indboot))+1):(4*ncol(indboot)))
.
compute templow =
llcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temphi =
ulcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
end if.
end if.
do if (nmeds = 2).
compute effkey = {xname, '->', mnames(1,1), '-
>', yname, ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, ' ', ' '}.
compute effkey = {indlbl(2:4,1), effkey}.
end if.

```

```

do if (nmeds = 3).
compute effkey = {xname, '->' , mnames(1,1), '->', yname, '','','',' '}.
compute effkey = {effkey; xname, '->' , mnames(1,1), '->', mnames(1,2), '->', yname, '','',' '}.
compute effkey = {effkey; xname, '->' , mnames(1,1), '->', mnames(1,3), '->', yname, '','',' '}.
compute effkey = {effkey; xname, '->' , mnames(1,1), '->', mnames(1,2), '->', mnames(1,3), '->', yname}.
compute effkey = {effkey; xname, '->' , mnames(1,2), '->', yname, '','','',' '}.
compute effkey = {effkey; xname, '->' , mnames(1,2), '->', mnames(1,3), '->', yname, '','',' '}.
compute effkey = {effkey; xname, '->' , mnames(1,3), '->', yname, '','','',' '}.
compute effkey = {indlbl(2:8,1), effkey}.
end if.

do if (nmeds = 4).
compute effkey = {xname, '->' , mnames(1,1), '->', yname, '','','',' ',' ',' '}.
compute effkey = {effkey; xname, '->' , mnames(1,1), '->', mnames(1,2), '->', yname, '','','',' '}.
compute effkey = {effkey; xname, '->' , mnames(1,1), '->', mnames(1,3), '->', yname, '','','',' '}.
compute effkey = {effkey; xname, '->' , mnames(1,1), '->', mnames(1,4), '->', yname, '','','',' '}.
compute effkey = {effkey; xname, '->' , mnames(1,1), '->', mnames(1,2), '->', mnames(1,3), '->', yname, '',' '}.
compute effkey = {effkey; xname, '->' , mnames(1,1), '->', mnames(1,2), '->', mnames(1,4), '->', yname, '',' '}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, ' ', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', mnames(1,4), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, ' ', ' ', ' ', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,4), '->', yname, '
', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, ' ', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, ' ', ' ', ' ', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', mnames(1,4), '->', yname, '
', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,4), '->', yname, ' ', ' ', ' ', ' ', ' ', ' '}.
compute effkey = {indlbl(2:16,1), effkey}.
end if.
print effkey/title = 'Indirect effect key'/format =
A8.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
end if.
DO IF (medte2=1).
compute
name=t(mdlnms2(2:nrow(mdlnms2),1)).

```



```

compute tmpdat=datayed(:,2:ncol(datayed)).
compute dd=tmpdat(:,1).
compute
dd={dd,tmpdat(:,(2+nmeds)),tmpdat(:,2:(1+nmeds))}.
compute stratify=0.
do if (ncovs > 0).
compute cname = cnames.
compute
dd={dd,tmpdat(:,(3+nmeds):(2+nmeds+ncovs))}.
end if.
compute method = 1.
compute samples = boot.
do if (mc > 0).
compute samples = mc.
compute method=2.
end if.
compute omnibus = 1.
compute booterr = 0.
do if (mcx < 0 or mcx > 4).
compute mcx = 1.
end if.
compute ovals = ncol(design(dd(:,1))).
compute nte = notes.
compute criterr = 0.
compute xskip = 0.
compute dumok = 0.
compute badend = 0.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute priorlo = -9999999.
compute priorhi = 9999999.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute reps = samples.
do if (contrast = 1).

```

```

compute note(nte,1) = 26.
compute nte=nte+1.
end if.
do if (effsize = 1).
compute note(nte,1) = 27.
compute nte=nte+1.
end if.
do if (normal = 1).
compute note(nte,1) = 28.
compute nte=nte+1.
end if.
do if (method = 2).
do if (omnibus = 1).
compute note(nte,1) = 25.
compute nte=nte+1.
end if.
compute samples = 0.
do if (reps < 1000).
compute reps = 1000.
end if.
end if.
do if (method = 1).
compute samples = reps.
end if.
compute temp = ncol(dd).
compute n = nrow(dd).
compute ones = make(n,1,1).
compute temp=dd(:,2).
compute nxd = ncol(temp).
compute nx = nxd.
do if (nx = 1 and mcx > 0).
compute temp = dd.
compute temp(GRADE(dd(:,2)),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,2)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute mnvls = cmin(t(nnvls)).
do if (mnvls < 2).
compute errs = errs + 1.

```

```

compute runerrs(errs,1) = 37.
compute criterr = 1.
end if.
do if (nvls > 9).
compute errs = errs+1.
compute runerrs(errs,1) = 36.
compute criterr = 1.
end if.
do if (criterr = 0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,2).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,2) <> nnvls((temp-1),1)).
compute nnvls(temp,1)=dd(i,2).
compute temp = temp+1.
end if.
end loop.
do if (mcx > 0).
compute x = dummy(:,2:ncol(dummy)).
compute nx = ncol(x).
compute minus1 = make(1,ncol(x),-1).
compute xdes=make((nx+1),3,0).
compute xdes(1,1)=dd(1,2).
compute xdes(1,2)=1.
compute temp = 2.
loop k = 2 to n.
do if (dd(k,2) <> dd((k-1),2)).
compute xdes(temp,2) = k.
compute xdes(temp,1) = dd(k,2).
compute xdes((temp-1),3) = k-1.
compute temp=temp+1.
end if.
end loop.
compute xdes((temp-1),3)=n.
compute xdes = {xdes, (xdes(:,3)-xdes(:,2)+1)}.
do if (stratify <> 0).
compute note(nte,1) = 7.
compute nte=nte+1.

```

```

end if.
do if (mcx = 4).
loop k = 1 to n.
do if (rsum(x(k,:)) = 0).
compute x(k,:) = minus1.
end if.
end loop.
end if.
do if (mcx = 2 or mcx = 3).
compute note(nte,1)=4.
compute nte = nte+1.
loop k = 1 to n.
do if (rsum(x(k,:)) > 0).
loop i = 1 to ncol(x).
do if (x(k,i) = 0).
compute x(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcx = 3).
compute conmat1={-8,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1; 0,0,0,0,0,-
3,1,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.
loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,:)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute x(k,:)=conmat1((rsum(x(k,:))+1),:).
end loop.
end if.
end if.
compute xdata = x.
compute xskip = 1.
compute xname = ddd(1,1:nx).

```

```

compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcx = 4).
compute dummat(1,:) = minus1.
end if.
do if (mcx = 2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcx = 3).
compute dummat=conmat1.
end if.
compute dummat={nnvls, dummat}.
compute xname2={name(1,2), xname}.
end if.
end if.
end if.
compute nm=nmeds.
compute r2b = make(nm,1,-999).
compute indeff = make((nx*nm),1,-999).
compute cov =
make((nx*nm+nm),(nx*nm+nm),0).
compute mns=make(nrow(cov),1,0).
compute ny = ncol(dd)-nm-nxd-ncovs.
compute y = dd(:,1).
compute ydata = y.
compute yname = name(1,1).
do if (xskip = 0).
compute x = dd(:,2:(1+nxd)).
compute xdata = x.
compute xname = name(1,2:(1+nxd)).
compute indlbs = t(xname).
end if.
compute m = dd(:,(2+nxd):(1+nxd+nm)).
compute mdata = m.

```

```

compute mlab = { ' M1 = ';' ' M2 = ';' ' M3 = ';' '
M4 = ';' ' M5 = ';' ' M6 = ';' ' M7 = ';' ' M8 = ';' '
M9 = ';' ' M10 = '}.
compute mlab = {mlab; ' M11 = ';' ' M12 = ';' '
M13 = ';' ' M14 = ';' ' M15 = '}.
compute mname =
name(1,(2+nxd):(1+nxd+nm)).
do if (ncovs > 0).
compute c =
dd(:,(2+nxd+nm):(1+nxd+nm+ncovs)).
compute x = {x, c}.
compute xdata = x.
compute cdata = c.
compute xname = {xname, cname}.
end if.
compute nms={yname; t(mname);
t(xname(1,1:(ncol(xname)-ncovs)))}.
do if (dumok = 1).
compute nms={yname; t(mname); name(1,2)}.
end if.
do if (nm < 16).
do if (dumok = 1).
print dummat/title = 'Coding of categorical X
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
compute
boots=make((samples+1),((nx+1)*nm),-999).
do if (errs = 0 and criterr = 0).
loop k = 1 to (samples+1).
do if (k = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef=0.
do if (k > 1).
compute u=trunc(uniform(n,1)*n)+1.
compute xdata = x(u,:).
compute mdata = m(u,:).

```

```

compute ydata = y(u,:).
do if (ncovs > 0).
compute cdata = c(u,:).
end if.
end if.
loop i = 1 to nm.
compute start = 1+(i-1)*nx.
compute xmat = {ones, xdata}.
do if (covmy=2).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = mdata(:,i).
compute amat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*amat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-(((1-r2)*(n-1))/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute r2cha = r2.
compute adjr2cha = adjr2.
do if (ncovs > 0 and covmy <> 2).
compute xmat = {ones, cdata}.
compute atmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*atmat)**2).
compute r2step1 = (sstotal-ssresid2)/sstotal.
compute r2cha = r2-r2step1.
compute adjr2cha = adjr2- (1-(((1-r2step1)*(n-
1))/(n-ncol(xmat)))).
compute xmat = {ones, xdata}.
end if.
compute bootcoef={bootcoef,t(amat)}.
compute r2b(i,1)=adjr2cha.
do if (k = 1).

```

```

compute hcinvtX=inv(t( xmat )' xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,1)*hcinvtX*t( xmat
(i3,1)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvtX*t( xmat )' xmat
*hcinvtX
.
compute tratio = amat&/sqrt(diag(covmat)).
compute rnms = mname(1,i).
print rnms/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.
compute lmat = ident(nrow(amat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*amat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*amat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/'format = F10.4
/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'.

```



```

compute op = {amat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*(xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute cov(start:(start+(nx-
1)),start:(start+(nx-
1)))=covmat(2:(ncol(covmat)-(ncovs*(covmy <>
2))),2:(ncol(covmat)-(ncovs*(covmy <> 2)))).
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns(start:(start+(nx-
1)),1)=amat(2:(1+nx)).
end loop.
compute xmat = {ones,mdata,xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (k = 1).
print yname/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.

```

```

do if (ovals = 2).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:'/cnames = nmsd/format = F9.2.
end if.
end if.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(yamat)/n)).
compute LL3 = yamat*&ln(pt2)+(1-yamat)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= yamat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).

```

```

compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute bootcoef={bootcoef,t(bmat)}.
compute LL1 = LL2.
compute covmat=varb.
end if.
do if (ovals <> 2).
compute bmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*bmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-(((1-r2)*(n-1))/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute bootcoef={bootcoef,t(bmat)}.
end if.
do if (k = 1).
do if (ovals = 2).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.

```

```

compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).
compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms2/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute
opdirsum=outp((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
end if.
do if (ovals <> 2).
.
compute hcinvtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).

```

```

end loop.
loop i3=1 to k3.
compute xmat(:,i3) = ( resid(:,ncol( resid
))&/(1-h))&* xmat(:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat(:,i3) = sqrt( mse )&* xmat(:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(bmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/(ncol(xmat)-1).
compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat((2+nmeds):(2+nmeds+(nx-1)),:)=
lmat2.
compute fratiodr =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/nx.
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute df2=(n-ncol(xmat)).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), df2, pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = bmat&/sqrt(diag(covmat)).
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute op = {bmat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat)))) }.

```

```

compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp))))*(xp2/(temp-
(2/3)+(.11/temp)))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms2/format =
F10.4.
compute
opdirsum=op((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
compute xmat = {ones,mdata}.
do if (ncovs > 0).
compute xmat = {xmat, cdata}.
end if.
compute btmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*btmat)**2).
compute r2direct = r2-((sstotal-
ssresid2)/sstotal).
compute pfr = 1-fcdf(fratiodr,(nx),(n-1-nx-nm-
ncovs)).
compute r2direct = {r2direct, fratiodr, nx, (n-1-
nx-nm-ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms2/cnames=rnms2/form
at= F10.4.
end if.
end if.

```

```

do if (k = 1 and toteff = 1).
print yname/title = '*****'
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:.'/format = A8.
compute xmat = {ones, xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(ymat)/n)).
compute LL3 = ymat*&ln(pt2)+(1-ymat)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= ymat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.

```

```

compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute LL1 = LL2.
compute covmat=varb.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(bmat)-
1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms = {'Constant'; t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).

```



```

compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute optotsum=outp(2:(1+nx),:).
end if.
do if (ovals <> 2).
compute cmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*cmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).

.

compute hcinvtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).

```

```

loop i3=1 to k3.
compute xmat(:,i3) = sqrt( mse )&* xmat(:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(cmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = cmatrix/sqrt(diag(covmat)).
compute op = {cmatrix, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/names = rnms/format =
F10.4.
compute optotsum=op(2:(1+nx),:).
compute ssresid2 = ssresid.
compute r2total = r2.

```

```

compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat(2:(1+nx),:) = lmat2.
compute fratio3 =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/nx.
do if (ncovs > 0).
compute xmat = {ones, cdata}.
compute ctmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*ctmat)**2).
compute r2total = r2-((sstotal-ssresid2)/sstotal).
end if.
compute pfr = 1-fcdf(fratio3,(nx),(n-1-nx-
ncovs)).
compute r2tot2 = {r2total, fratio3, nx, (n-1-nx-
ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns((nrow(mns)-
nm+1):nrow(mns),1)=bmat(2:(1+nm),1).
compute tmp = 1.
loop i = 1 to (nm*nx).
compute
boots(k,i)=mns(i,1)*mns(((nm*nx)+tmp),1).
do if ((i/nx)=trunc(i/nx)).
compute
boots(k,((nm*nx)+tmp))=r2b(tmp,1)*mns(((nm*
nx)+tmp),1).
compute tmp = tmp+1.
end if.
end loop.
do if (savboot = 1 and k > 1).

```

```

compute bootstrp((k-1),:)=bootcoef.
end if.
end loop.
do if (method = 1).
compute llci=make(1,ncol(boots),-999).
compute ulci=make(1,ncol(boots),-999).
compute indeff = t(boots(1,1:(nm*nx))).
compute indeff22=t(boots(1,:)).
compute omni = t(boots(1,((nm*nx)+1) :
ncol(boots))).
do if (nrow(boots) > 1).
compute boots = boots(2:nrow(boots),:).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
loop i = 1 to ncol(boots).
.
compute temp = boots(:,i).
compute temp(GRADE( boots(:,i) )) = boots(:,i).
compute badlo = 0.
compute badhi = 0.
do if ( indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(indeff22(i,1)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.

```

```

do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,i)=llcit.
compute ulci(1,i)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute llci = t(llci).
compute ulci= t(ulci).
compute llci={llci,ulci}.
end if.
end if.
do if (method = 2).
compute tmp=1.
loop i = 1 to (nm*nx).
compute
indeff(i,1)=mns(i,1)&*mns(((nm*nx)+tmp),1).

```

```

do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = sqrt(-
2*ln(uniform(reps,nrow(cov))))&*cos((2*3.1415
9265358979)*uniform(reps,nrow(cov))).
compute boots=boots*chol(cov).
loop i = 1 to ncol(boots).
compute boots(:,i)=boots(:,i)+mns(i,1).
end loop.
compute tmp=1.
loop i = 1 to (nm*nx).
compute
boots(:,i)=boots(:,i)&*boots(:,((nm*nx)+tmp)).
compute temp = boots(:,i).
compute temp(GRADE(boots(:,i))) = boots(:,i).
compute boots(:,i) = temp.
do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = boots(:,1:(nm*nx)).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
compute llci =
t(((boots(cilow,:));(boots(cihigh,:)))).
end if.
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
compute rnmssum=rnms(2:(1+nx),1).
do if (ovals <> 2).
print optotsum/title = 'Relative total effects of X
of Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.

```

```

print r2tot2/title = 'Omnibus test of total effect of
X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print optotsum/title = 'Relative total effects of X
on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
end if.
compute
rnmssum=rnms2((2+nmeds):(1+nmeds+nx),1).
do if (ovals <> 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.
print r2direct/title = 'Omnibus test of direct
effect of X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
do if (omnibus = 1 and (method = 1 or method
= 3)).
compute indlbs = {indlbs; 'Omnibus'}.
end if.
loop i = 1 to nm.
compute op = mname(1,i).
print op/title = 'Relative indirect effect(s) of X on
Y through:.'/format = A8/space=0.
do if (method = 2).
compute clbs = {'Effect', 'SE MC', 'MC LLCI',
'MC ULCI'}.
compute op = {indeff, bootse(1:(nm*nx)), lici}.
end if.

```

```

do if (method = 1).
compute clbs = {'Effect', 'SE(boot)', 'LLCI',
'ULCI'}.
compute op = indeff.
do if (samples > 0).
compute op = {indeff, bootse(1:(nm*nx),:),
llci(1:(nm*nx),:)}).
end if.
end if.
compute temp = op((((i-1)*nx)+1):(nx*i),:).
do if ((method = 1) and omnibus = 1).
do if (samples = 0).
compute temp={temp;omni(i,1)}.
end if.
do if (samples > 0).
compute temp = {temp; omni(i,1),
bootse(((nm*nx)+i),:), llci(((nm*nx)+i),:)}).
end if.
end if.
print temp/title = ' '/rnames = indlbs/cnames =
clbs/format = F10.4 /space=0.
print/title = '-----'.
end loop.
end if.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
END IF.
DO IF (mdpbe=1).
compute nm={yname,xname,mnames}.
do if (ncovs > 0).

```



```

compute
nm={yname,cnames,xname,mnames}.
end if.

compute tpx=xname.
compute x = dd(:,2:ncol(dd)).
compute tpy = yname.
compute tempy= dd(:,1).
compute n = nrow(dd).
compute nx=1.
compute criterr=0.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute ddd1 = {'int_1', 'int_2', 'int_3', 'int_4',
'int_5', 'int_6', 'int_7', 'int_8', 'int_9'}.
compute dumok = 0.
compute mcfoc=mcx.
compute mcmmod=mcm.
compute mcloc=(mcfoc>0).
compute con = make(n,1,1).
compute ncovs=ncol(x)-2.
do if (mcfoc > 0 or mcmmod > 0).
compute temp = dd.
compute temp(GRADE(dd(:,(ncol(dd)-
mcloc))),:)= dd.
compute dd = temp.
compute dummy = design(dd(:,ncol(dd)-
mcloc)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute toosmall=rsum(nnvls < 2).
compute mnvls = cmin(t(nnvls)).
do if ((rsum(nnvls < 2)) > 0).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=37.
end if.
do if (nvls > 10).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=36.

```

```

end if.
do if (criterr=0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,ncol(dd)-mcloc).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,ncol(dd)-mcloc) <> nnvls((temp-
1),1)).
compute nnvls(temp,1)=dd(i,ncol(dd)-mcloc).
compute temp = temp+1.
end if.
end loop.
compute dummy = dummy(:,2:ncol(dummy)).
do if (mcfoc=4 or mcmmod=4).
compute minus1=make(1,ncol(dummy),-1).
loop k = 1 to n.
do if (rsum(dummy(k,:)) = 0).
compute dummy(k,:)=minus1.
end if.
end loop.
end if.
do if (mcfoc = 2 or mcfoc = 3) or (mcmmod = 2 or
mcmmod = 3)).
loop k = 1 to n.
do if (rsum(dummy(k,:)) > 0).
loop i = 1 to ncol(dummy).
do if (dummy(k,i) = 0).
compute dummy(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcfoc = 3 or mcmmod=3).
compute conmat1={-8,1,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1,1; 0,0,0,0,0,-
3,1,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.

```

```

loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,:)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute
dummy(k,:)=conmat1((rsum(dummy(k,:))+1),:).
end loop.
end if.
end if.
compute nx = ncol(dummy).
compute xname = ddd(1,1:nx).
compute xdata = dummy.
compute xname = ddd(1,1:nx).
compute xname2={nm(1,(ncol(nm)-mcloc)),
xname}.
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcfoc = 2 or mcmmod=2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcfoc = 3 or mcmmod=3).
compute dummat=conmat1.
end if.
do if (mcfoc = 4 or mcmmod = 4).
compute dummat(1,:)=minus1.
end if.
compute dummat={nnvls, dummat}.
end if.
end if.
do if (criterr=0).
compute y = dd(:,1).
compute ovals = ncol(design(dd(:,1))).

```

```

compute itprob = 0.
do if (ovals = 2).
compute omx = cmax(y(:,1)).
compute omn = cmin(y(:,1)).
compute y(:,1) = (y(:,1) = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute sstotal = csum((y-(csum(y)/n))&**2).
compute outv = t(nm(1,1)).
compute mdtr = nm(1,ncol(dd)).
compute fciv = nm(1,(ncol(dd)-1)).
do if (mcfoc <> 0 or mcmod <> 0).
compute inter=make(n,nx,0).
loop i = 1 to nx.
compute inter(:,i)=dummy(:,i)&*dd(:,ncol(dd)-
(1-mcloc)).
end loop.
do if (ncovs = 0).
compute x={con,dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
do if (ncovs > 0).
compute
x={con,dd(:,2:(1+ncovs)),dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={t(nm(1,2:(1+ncovs)));
nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
end if.
compute xmns = csum(x)/n.
compute focvals = {1,0,0}.
compute highwarn = 0.
compute lowwarn = 0.

```

```

do if (ncol(x) > (2*nx+2)).
compute covmns = {1, xmns(1,2:(ncol(x)-1-
(2*nx)))}.
compute focvals = {covmns,0,0}.
end if.
compute dfres = n-ncol(x).
do if (ovals = 2).
compute pt2 = make(n,1,(csum(y)/n)).
compute LL3 = y*&ln(pt2)+(1-y)*&ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(x),1,0).
compute LL1 = 0.

.

compute pt1lp= pt1.
compute xlp= x.
compute ylp= y.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)*&ln(1-
pt1lp).

```

```

compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlx).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
else if (ovals <> 2).
compute invXtX=inv(t(x)*x).
compute b = invXtX*t(x)*y.
compute resid = y-(x*b).
compute ssresid=csum(resid*resid).
compute msresid=ssresid/dfres.
compute varb = msresid*invXtX.
compute
seint=sqrt(varb(nrow(varb),nrow(varb))).
compute k3 = nrow(b).
do if (hc3 = 1).
compute xhc=x.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)= xhc(i3,:)*invXtX*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).
end loop.
compute varb=(invXtX*t(xhc)*xhc*invXtX).
end if.
compute seb = sqrt(diag(varb)).
compute r2 = 1-(ssresid/sstotal).

```

```

do if (mcfoc <> 0 or mcmod <> 0).
compute ytp=y.
compute xtp=x(:,1:(ncol(x)-nx)).
compute btp = inv(t(xtp)*xtp)*t(xtp)*ytp.
compute residt = ytp-(xtp*btp).
compute ssresidt=csum(residt*&residt).
compute r2noint = 1-(ssresidt/sstotal).
end if.
compute pr = ncol(x)-1.
compute lmat = ident(nrow(b)).
compute lmat = lmat(:,2:ncol(lmat)).
compute f =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
pr).
compute pf = 1-fcdf(f,pr,dfres).
compute pf = {sqrt(r2),r2,msresid,f,pr,dfres,pf}.
compute xd=abs(xp2).
end if.
do if (itprob = 0).
compute tstat = b&/seb.
do if (ovals <> 2).
compute p = 2*(1-tcdf(abs(tstat), dfres)).
else if (ovals = 2).
compute p = 2*(1-cdfnorm(abs(tstat))).
end if.
compute outp = {b,seb,tstat,p}.
compute nms = {'constant'; nms; 'interact'}.
do if (ovals <> 2).
compute tval = (dfres* (exp((dfres-
(5/6))*((xd/(dfres-(2/3)+(.11/dfres)))*(xd/(dfres-
(2/3)+(.11/dfres)))))-1)).
compute outp={outp,(b-
sqrt(tval)&*seb),(b+sqrt(tval)&*seb)}.
end if.
compute bb = tval.
do if (dumok = 1).
do if (mcfoc > 0).
print dummat/title = 'Coding of categorical X
variable for analysis:'/cnames = xname2/format
= F5.2.

```

```

else if (mcmod > 0).
print dummat/title = 'Coding of categorical M
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
print yname/title =
*****
*****'/rlabels = 'Outcome: '/format
= A8.
do if (ovals = 2).
compute nmsd = {outv, 'Analysis'}.
print rcd/title = 'Coding of binary Y for
analysis: '/cnames = nmsd/format = F10.4.
compute outp = {outp,(outp(:,1)-
sqrt(tval)&*outp(:,2)),(outp(:,1)+sqrt(tval)&*outp
(:,2))}.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
print outp/title = 'Logistic Regression
Model'/rnames = nms/clabels 'Coeff' 'se' 'Z' 'p'
'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
compute varbtmp=varb.
compute btmp=b.

```



```

compute LL2f=LL2.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=x(:,1:(ncol(x)-nx)).
compute bt1 = make(ncol(xlp),1,0).

.

compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(yp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-yp)&ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.

```

```

compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL2f.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(LLdiff,nx).
compute rcha={LLdiff,nx,pchi}.
end if.
do if (ovals <> 2).
print pf/title = 'Model Summary'/clabels = 'R' 'R-
sq' 'MSE' 'F' 'df1' 'df2' 'p'/format F10.4.
compute lmat=make(nrow(b),1,0).
compute lmat(nrow(lmat),1)=1.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
1).
compute
rcha=((b(nrow(b),1)/seint)*(b(nrow(b),1)/seint))*
(1-r2)/dfres.
compute rcha = {rcha, fcha, 1, dfres,
outp((pr+1),4)}.
print outp/title = 'Model'/rnames = nms/clabels
'coeff' 'se' 't' 'p' 'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
do if (mcfoc > 0 or mcmod > 0).
compute rcha=r2-r2noint.
compute lmat=make((nrow(b)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
nx).
compute pvalr2c=1-fcdf(fcha,nx,dfres).

```

```

compute rcha = {rcha, fcha, nx, dfres,pvalr2c}.
end if.
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute intkey = {'a', 'b', 'c', 'd', 'e'}.
loop i = 1 to nx.
compute intkey={intkey; ddd1(1,i), ' : ', ddd(1,i),
' X ', nm(1,ncol(dd)-(1-mcloc))}.
end loop.
compute intkey=intkey(2:nrow(intkey),:).
print intkey/title='Product terms key:.'/format =
a8.
end if.
do if (ovals <> 2).
print rcha/title = 'R-square increase due to
interaction:.'/clabels 'R2-chng' 'F' 'df1' 'df2'
'p'/format F10.4.
else if (ovals = 2).
print rcha/title = 'Likelihood ratio test for
interaction:.'/clabels 'Chi-sq' 'df' 'p'/format F10.4.
end if.
print/title =
*****
*****'.
compute mdvar = x(:,(ncol(x)-1)).
do if (mcfoc = 0 and mcmmod = 0).
compute g1 = b((ncol(x)-2),1).
compute g3 = b(ncol(x),1).
compute vg1 = varb((ncol(x)-2),(ncol(x)-2)).
compute vg3 = varb(ncol(x),ncol(x)).
compute covg1g3 = varb((ncol(x)-2), ncol(x)).
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute mdvar = x(:,(ncol(x)-(2*nx))).
end if.
compute mdmin = cmin(mdvar).
compute mdmax = cmax(mdvar).
compute fvar = x(:,(ncol(x)-2)).
compute nval = ncol(design(mdvar)).
compute fvmin = cmin(fvar).

```

```

compute fvmax = cmax(fvar).
do if (mmodval = 999).
compute mnmd = csum(mdvar)/n.
compute tmp = make(n,1,mnmd).
compute sdmd = sqrt(csum(((mdvar-
tmp)**2))/(n-1)).
compute probeval = {mnmd-sdmd; mnmd;
mnmd+sdmd}.
do if (probeval(1,1) < mdmin).
compute lowwarn = 1.
compute probeval(1,1)=mdmin.
end if.
do if (probeval(nrow(probeval),1) > mdmax).
compute highwarn = 1.
compute probeval(nrow(probeval),1)=mdmax.
end if.
do if (quantile = 1).
compute tmp = mdvar.
compute tmp(GRADE(mdvar(:,1)),:) = mdvar.
compute mdvar = tmp.
compute
probeval={mdvar(trunc(0.1*n),1);mdvar(trunc(0
.25*n),1);mdvar(trunc(0.5*n),1);mdvar(trunc(0.7
5*n),1);mdvar(trunc(0.9*n),1)}.
end if.
end if.
do if (nval = 2).
compute probeval = make(2,1,0).
compute probeval(1,1)=cmin(mdvar).
loop i = 1 to n.
do if (mdvar(i,1) <> probeval(1,1)).
compute probeval(2,1) = mdvar(i,1).
BREAK.
end if.
end loop.
end if.
do if (mmodval <> 999).
compute probeval = mmodval.
end if.
compute outp = make(nrow(probeval),7,0).

```

```

do if (mcfoc > 0 or mcmmod > 0).
compute focvals=make(1,ncol(x)+1,1).
do if (mcfoc > 0).
print/title = 'Conditional effect of X on Y at
values of the moderator:'.
print/title = ' '/space=0.
compute rnn2=mdtr.
compute matt=make(nx,6,0).
end if.
loop jj=1 to nrow(probeval).
do if (mcfoc > 0).
loop ii=1 to nx.
compute g1=b((ncol(x)-(2*nx)+ii),1).
compute g3=b((ncol(x)-nx+ii),1).
compute vg1=varb((ncol(x)-(2*nx)+ii),(ncol(x)-
(2*nx)+ii)).
compute vg3=varb((ncol(x)-nx+ii),(ncol(x)-
nx+ii)).
compute covg1g3=varb((ncol(x)-
(2*nx)+ii),(ncol(x)-nx+ii)).
compute x2 = probeval(jj,1).
compute w1 = g1+g3*x2.
compute varw1 =
vg1+(2*x2*covg1g3)+((x2*x2)*vg3).
compute sew1 = sqrt(varw1).
compute t1 = w1/sew1.
do if (ovals <> 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-tcdf(abs(t1), dfres)).
compute cnms = {'Coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)= {w1,sew1,t1,p,llci,ulci}.
else if (ovals = 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-cdfnorm(abs(t1))).
compute cnms = {'Coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)= {w1,sew1,t1,p,llci,ulci}.

```

```

end if.
end loop.
compute rnms=t(xname).
compute mdvalpr=probeval(jj,1).
print mdvalpr/title = 'Moderator
value: '/rnames=rnn2/format= F10.4 /space=0.
print matt/title='
'/cnames=cnms/rnames=rnms/format= F10.4
/space=0.
compute xprob=x(:,(ncol(x)-(2*nx)))-mdvalpr.
loop kk = 1 to nx.
compute xprob={xprob,
(xprob(:,1)&*x(:,(ncol(x)-(2*nx)+kk)))}.
end loop.
compute xprob={x(:,1:((ncol(x)-(2*nx))-
1)),xprob}.
do if (ovals <> 2).
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute residc = y-(xprob*bmultc).
compute ssresidc=csum(residc&*residc).
compute r2c = r2-(1-(ssresidc/sstotal)).
compute fcha2=(dfres*r2c)/(nx*(1-r2)).
do if (hc3 = 1).
loop kk = 1 to nx.
compute xprob={xprob, x(:,(ncol(x)-
(2*nx)+kk))}.
end loop.
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute k3 = nrow(bmultc).
compute xhc=xprob.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)=
xhc(i3,:)*inv(t(xprob)*xprob)*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).

```

```

end loop.
compute
varbc=(inv(t(xprob)*xprob)*t(xhc)*xhc*inv(t(xprob)*xprob)).
compute lmat=make((nrow(bmultc)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha2 =
(t(t(lmat)*bmultc)*inv(t(lmat)*varbc*lmat)*((t(lmat)*bmultc)))/nx).
end if.
compute pvalr2cc=1-fcdf(fcha2,nx,dfres).
compute rcha2 = {r2c, fcha2, nx,
dfres,pvalr2cc}.
print rcha2/title = 'Test of equality of conditional
means at this value of the moderator'/clabels
'R2-chng' 'F' 'df1' 'df2' 'p'/format F10.4.
else if (ovals = 2).
compute btmp=b.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=xprob.
compute bt1 = make(ncol(xlp),1,0).

.
compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.

```

```

end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&*ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*ulp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL4.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(lldiff,nx).
compute rcha2 = {LLdiff, nx,pchi}.
print rcha2/title = 'Test of equality of log odds
conditioned on this moderator value'/clabels
'Chi-sq' 'df' 'p'/format F10.4.
end if.
end if.
compute
tttt=make(nrow(dummat),1,probeval(jj,1)).
compute tttt={tttt,dummat(:,2:ncol(dummat))}.
loop kkk=1 to nx.
compute tttt={tttt,tttt(:,1)&*tttt(:,1+kkk)}.
end loop.
compute ones=make(nrow(dummat),1,1).
do if (ncol(x) > (2*nx+2)).

```



```

compute
covmnmat=make(nrow(tttt),ncol(covmns),0).
loop kkk=1 to nrow(tttt).
compute covmnmat(kkk,:)=covmns.
end loop.
compute tttt={covmnmat,tttt}.
else.
compute tttt={ones,tttt}.
end if.
compute focvals={focvals;dummat(:,1),tttt}.
do if (mcfoc > 0).
compute yhat={dummat(:,1),(tttt*b)}.
compute cmnms={fciv, 'yhat'}.
do if (ovals <> 2).
print yhat/title = 'Estimated conditional means
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (ovals = 2).
print yhat/title = 'Estimated conditional log odds
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (jj <> nrow(probeval)).
print/title='-----'/space=0.
end if.
end if.
end loop.
compute focvals=focvals(2:nrow(focvals),:).
compute yhat=focvals(:,2:ncol(focvals))*b.
compute
focvals={focvals(:,1),focvals(:,(ncol(focvals)-
(2*nx))),yhat}.
compute cnms={fciv,mdtr,'yhat'}.
do if (mcmod > 0).
compute cnms={mdtr,fciv,'yhat'}.
end if.
end if.
do if (mcmod > 0).
compute outp=make((nx+1),7,0).

```

```

compute bcatm={b((ncol(x)-
(2*nx)),1);b((ncol(x)-nx+1):ncol(x),1)}.
compute outp(:,1)=dummat(:,1).
compute bcatcov=varb((ncol(x)-
nx):ncol(x),(ncol(x)-nx):ncol(x)).
compute bcatcov(1,1)=varb((ncol(x)-
(2*nx)),(ncol(x)-(2*nx))).
compute
bcatcov(2:nrow(bcatcov),1)=varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx)).
compute
bcatcov(1,2:nrow(bcatcov))=t(varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx))).
loop i = 1 to nrow(dummat).
compute
catmval={1,dummat(i,2:ncol(dummat))}.
compute condeff=catmval*bcatm.
compute
condse=sqrt(catmval*bcatcov*t(catmval)).
compute outp(i,2:3)={condeff,condse}.
end loop.
compute outp(:,4)=outp(:,2)&/outp(:,3).
compute outp(:,5) = 2*(1-tcdf(abs(outp(:,4)),
dfres)).
compute outp(:,6) = (outp(:,2)-
sqrt(tval)&*outp(:,3)).
compute outp(:,7) =
(outp(:,2)+sqrt(tval)&*outp(:,3)).
do if (ovals <> 2).
compute cnmms = {xname2(1,1), 'coeff', 'se',
't', 'p', 'LLCI', 'ULCI'}.
end if.
do if (ovals = 2).
compute outp(:,5) = 2*(1-
cdfnorm(abs(outp(:,4)))).
compute cnmms = {xname2(1,1), 'coeff', 'se',
'Z', 'p', 'LLCI', 'ULCI'}.
end if.

```

```

print outp/title = 'Conditional Effect of Focal
Predictor in Groups Defined by the Moderator
Variable:.'/cnames=cnmms/format = F10.4.
end if.
do if (nval > 2 and (mmodval = 999) and
mcmmod = 0).
do if (quantile <> 1).
print/title = 'Moderator values are the sample
mean and plus/minus one SD from mean'.
else.
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
end if.
do if (highwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD above the mean was'.
print/title = 'replaced with the maximum
because one SD above the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
do if (lowwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD below the mean was'.
print/title = 'replaced with the minimum
because one SD below the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
end if.
do if (nval = 2 and (mmodval = 999)).
print/title = 'The moderator variable is
dichotomous'.
end if.
do if (mmodval <> 999 and mcmmod = 0 and
((mmodval < mdmin) or (mmodval > mdmax))).
print/title = 'Warning: Moderator value specified
is outside of the range of the data'.
end if.

```

```

do if (plot = 1).
do if (mcfoc = 0 and mcmmod=0).
compute cnms = {t(nms((ncol(x)-2):(ncol(x)-
1),1)), 'yhat'}.
end if.
do if (ovals = 2).
compute prob =
exp(focvals(:,3))/(1+exp(focvals(:,3))).
compute focvals = {focvals, prob}.
compute cnms = {cnms, 'prob'}.
end if.
print/title =
*****
*****!

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (ovals = 2).
do if (mcfoc=0).
print/title = 'DATA LIST
FREE/Technological_Advan Pro_Env_Strgy
logodds prob.' /space=1.
end if.
do if (mcmmod=0).

```

```

print/title = 'DATA LIST FREE/Pro_Env_Strgy
Technological_Advan logodds prob.'
/space=1.
end if.
else if (ovals <> 2).
do if (mcfoc=0).
print/title = 'DATA LIST
FREE/Technological_Advan Pro_Env_Strgy
Operational_Performance.' /space=1.
end if.
do if (mcmmod=0).
print/title = 'DATA LIST FREE/Pro_Env_Strgy
Technological_Advan
Operational_Performance.' /space=1.
end if.
end if.
print/title = 'BEGIN DATA.' /space=0.
print focvals/title = ' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (ovals <> 2 and mcmmod=0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH Operational_Performance BY
Pro_Env_Strgy.' /space=0.
end if.
do if (ovals <> 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH Operational_Performance BY
Technological_Advan.' /space=0.
end if.
do if (ovals = 2 and mcmmod=0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Pro_Env_Strgy.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Pro_Env_Strgy.' /space=0.
end if.
do if (ovals = 2 and mcfoc=0).

```

```

print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH logodds BY Technological_Advan.'
/space=0.
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH prob BY Technological_Advan.'
/space=0.
end if.
do if (ncovs > 0).
print/title = 'NOTE: For data above, covariates
are set to their sample means.'.
end if.
end if.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
end if.
END IF.
end if.
do if (bad > 0).
compute note(notes,1) = 9.
compute notes = notes + 1.
end if.
print/title = '***** ANALYSIS
NOTES AND WARNINGS
*****'.
loop i = 1 to errs.
do if (runerrs(i,1) = 1).
print/title = 'ERROR: One of your declared
mediators is dichotomous. This procedure
can"t be used.'.
end if.
do if (runerrs(i,1) = 2).

```

```
print/title = 'ERROR: For model 6, this
procedure limits the number of mediators to
four.'.
end if.
do if (runerrs(i,1) = 3).
print/title = 'ERROR: For models 1, 2, and 3,
only a single variable can be listed in the M
list.'.
end if.
do if (runerrs(i,1) = 4).
print/title = 'ERROR: You requested a model
involving W but did not provide a valid W
variable name.'.
end if.
do if (runerrs(i,1) = 5).
print/title = 'ERROR: You requested a model
involving Z but did not provide a valid Z
variable name.'.
end if.
do if (runerrs(i,1) = 6).
print/title = 'ERROR: You requested a model
involving Q but did not provide a valid Q
variable name.'.
end if.
do if (runerrs(i,1) = 7).
print/title = 'ERROR: You requested a model
involving V but did not provide a valid V
variable name.'.
end if.
do if (runerrs(i,1) = 8).
print/title = 'ERROR: You specified a W
variable for a model that does not need it'.
end if.
do if (runerrs(i,1) = 9).
print/title = 'ERROR: You specified a Z variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 10).
print/title = 'ERROR: You specified a Q variable
for a model that does not need it'.
```

```
end if.  
do if (runerrs(i,1) = 11).  
print/title = 'ERROR: You specified a V variable  
for a model that does not need it.'  
end if.  
do if (runerrs(i,1) = 12).  
print/title = 'ERROR: The variable specified for  
W has already been assigned.'  
end if.  
do if (runerrs(i,1) = 13).  
print/title = 'ERROR: The variable specified for  
Z has already been assigned.'  
end if.  
do if (runerrs(i,1) = 14).  
print/title = 'ERROR: The variable specified for  
Q has already been assigned.'  
end if.  
do if (runerrs(i,1) = 15).  
print/title = 'ERROR: The variable specified for  
V has already been assigned.'  
end if.  
do if (runerrs(i,1) = 16).  
print/title = 'ERROR: You did not provide a  
valid Y variable name.'  
end if.  
do if (runerrs(i,1) = 17).  
print/title = 'ERROR: The variable specified for  
Y has already been assigned.'  
end if.  
do if (runerrs(i,1) = 18).  
print/title = 'ERROR: Model 6 requires more  
than one mediator.'  
end if.  
do if (runerrs(i,1) = 19).  
print/title = 'ERROR: You have not specified a  
valid model number.'  
end if.  
do if (runerrs(i,1) = 20).  
print/title = 'ERROR: At least one and only one  
variable must be listed for X.'
```



```
end if.  
do if (runerrs(i,1) = 21).  
print/title = 'ERROR: At least one and only one  
variable must be listed for Y.'.  
end if.  
do if (runerrs(i,1) = 22).  
print/title = 'ERROR: Iteration didn"t converge  
to a solution. Interpret results with caution.'.  
end if.  
do if (runerrs(i,1) = 23).  
print/title = 'ERROR: You specified a clustering  
variable that does not exist in your variable  
list.'.  
end if.  
do if (runerrs(i,1) = 24).  
print/title = 'ERROR: You specified a clustering  
variable that has already been assigned.'.  
end if.  
do if (runerrs(i,1) = 25).  
print/title = 'ERROR: One or more of your M  
variables is not listed in the variables list.'.  
end if.  
do if (runerrs(i,1) = 26).  
print/title = 'ERROR: A maximum of 20 cluster  
units is allowed. Use multilevel modeling  
instead.'.  
end if.  
do if (runerrs(i,1) = 27).  
print/title = 'ERROR: One of the variables in  
your model is a constant.'.  
end if.  
do if (runerrs(i,1) = 28).  
print/title = 'ERROR: Dichotomous Y is not  
permitted with WS option.'.  
end if.  
do if (runerrs(i,1) = 29).  
print/title = 'ERROR: Insufficient number of  
variables in vars= list when using WS option.'.  
end if.  
do if (runerrs(i,1) = 30).
```

```

print/title = 'ERROR: Too many variables in
vars= list when using WS option. Covariates
not allowed.'.
end if.
do if (runerrs(i,1) = 31).
print/title = 'ERROR: mmodval and wmodval
can"t both be set to zero with contrast option.'.
end if.
do if (runerrs(i,1) = 32).
print/title = 'ERROR: You did not provide a
valid X variable name.'.
end if.
do if (runerrs(i,1) = 33).
print/title = 'ERROR: PROCESS requires all
variable names to be eight characters or
fewer.'.
print/title = '      Please shorten variable names
and reexecute.'/space=0.
end if.
do if (runerrs(i,1)=34).
print/title = 'ERROR: X and M cannot both be
specified as multicategorical.'.
end if.
do if (runerrs(i,1)=35).
print/title = 'ERROR: Multicategorical variable
specification available only for models 1 and
4.'.
end if.
do if (runerrs(i,1)=36).
print/title = 'ERROR: Categorical variables
cannot have more than 10 categories.'.
end if.
do if (runerrs(i,1)=37).
print/title = 'ERROR: Each group must have at
least two cases.'.
end if.
do if (runerrs(i,1)=38).
print/title = 'ERROR: Cluster option not
available with multicategorical variables.'.
end if.

```

```

do if (runerrs(i,1)=39).
print/title = 'ERROR: A maximum of 10
variables is allowed in the M= list.'.
end if.
do if (runerrs(i,1) = 41).
print/title = 'ERROR: MCM option not permitted
in model 4.'.
end if.
end loop.
do if (errs = 0).
do if (boot > 1 or mc > 0).
do if (bconoff = 1 and boot > 1).
print boot/title = 'Number of bootstrap samples
for bias corrected bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (bconoff = 0 and boot > 1).
print boot/title = 'Number of bootstrap samples
for percentile bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (mc > 1).
print mc/title = 'Number of samples for Monte
Carlo confidence intervals:'/format = F8.0.
end if.
do if (booterr = 1).
compute badend = badend(1,2:ncol(badend)).
print badend/title = 'WARNING: Bootstrap CI
endpoints below not trustworthy. Decrease
confidence or increase bootstraps'/format =
F10.4.
end if.
end if.
print conf/title = 'Level of confidence for all
confidence intervals in output:'/format = F8.2.
do if ((center = 1 or ws=1) and (ncol(centvar) >
1)).
compute centvar = centvar(1,2:ncol(centvar)).

```

```

print centvar/title = 'NOTE: The following
variables were mean centered prior to
analysis:.'/format = a8.
end if.
loop i = 1 to notes.
do if (note(i,1) = 1).
print/title = 'NOTE: Confidence level restricted
to between 50 and 99.9999%. 95% confidence
is provided in output'.
end if.
do if (note(i,1) = 2).
print/title = 'NOTE: Effect size measures for
indirect effects are not available for models
with dichotomous outcomes'.
end if.
do if (note(i,1) = 3).
print/title = 'NOTE: All standard errors for
continuous outcome models are based on the
HC3 estimator'.
end if.
do if (note(i,1) = 6).
print/title = 'NOTE: The number of bootstrap
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 7).
print/title = 'NOTE: The Johnson-Neyman
method is available only for Models 1 and 3'.
end if.
do if (note(i,1) = 8).
print/title = 'NOTE: The Johnson-Neyman
method cannot be used with a dichotomous
moderator'.
end if.
do if (note(i,1)=9).
print bad/title = 'NOTE: Some bootstrap
samples had to be replaced. The number of
such replacements was:'.
end if.
do if (note(i,1) = 11).

```

```

print nmiss/title = 'NOTE: Some cases were
deleted due to missing data. The number of
such cases was:'.
end if.
do if (note(i,1) = 12).
print /title = 'NOTE: Monte Carlo method
available only for models 4 and 5.
Bootstrapping was used instead.'.
end if.
do if (note(i,1) = 13).
print/title = 'NOTE: The number of Monte Carlo
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 19).
print/title = 'NOTE: Effect sizes not available for
within-subject analyses.'.
end if.
do if (note(i,1) = 16).
print/title = 'NOTE: Normal theory tests not
available for within-subject analyses.'.
end if.
do if (note(i,1) = 17).
print/title = 'NOTE: Monte Carlo confidence
intervals not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 18 and warnrep=0).
print/title = 'WARNING: You have requested
OLS estimation with a dichotomous criterion.'.
print/title = 'Interpret model coefficients and
inferential statistics with caution.'/space=0.
compute warnrep=1.
end if.
do if (note(i,1) = 20).
print/title = 'NOTE: Saving of bootstrap
estimates not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 22).

```

```
print/title = 'NOTE: Effect size option with
covariates requires covariates in models of M
and Y.'.
end if.
do if (note(i,1) = 23).
print/title = 'NOTE: Johnson-Neyman method
not available with multicategorical X'.
end if.
do if (note(i,1) = 24).
print/title = 'NOTE: MMODVAL option not
available with a multicategorical moderator'.
end if.
do if (note(i,1) = 25).
print/title = 'NOTE: Omnibus test for relative
indirect effects is not available with the MC
option.'.
end if.
do if (note(i,1) = 26).
print/title = 'NOTE: CONTRAST option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 27).
print/title = 'NOTE: EFFSIZE option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 28).
print/title = 'NOTE: NORMAL option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 31).
print/title = 'NOTE: The TOTAL option is not
available when Y is dichotomous.'.
end if.
do if (note(i,1) = 32).
print/title = 'NOTE: Kappa-squared is disabled
from output as of version 2.16.'.
end if.
do if (note(i,1) = 33).
print/title = 'NOTE: COVMY option ignored in
models 1, 2, and 3.'.
```

| | | | |
|-----------|----------------|-------------|-------------|
| | | end if. | |
| | | end loop. | |
| | | end if. | |
| | | end matrix. | |
| Resources | Processor Time | | 00:00:00.89 |
| | Elapsed Time | | 00:00:00.92 |

Run MATRIX procedure:

***** PROCESS Procedure for SPSS Release 2.16.3 *****

Written by Andrew F. Hayes, Ph.D. www.afhayes.com
Documentation available in Hayes (2013). www.guilford.com/p/hayes3

Model = 1
Y = Operatio
X = Pro_Env_
M = Technolo

Sample size
856

Outcome: Operatio

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|-----------|--------|----------|-------|
| .9192 | .8449 | .1863 | 1547.3053 | 3.0000 | 852.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|--------|-------|---------|-------|--------|--------|
| constant | -.3956 | .1221 | -3.2411 | .0012 | -.6352 | -.1560 |
| Technolo | .4223 | .0459 | 9.1946 | .0000 | .3322 | .5125 |
| Pro_Env_ | .8990 | .0458 | 19.6345 | .0000 | .8091 | .9888 |
| int_1 | -.0603 | .0126 | -4.8032 | .0000 | -.0849 | -.0357 |

Product terms key:

int_1 Pro_Env_ X Technolo

R-square increase due to interaction(s):

| | R2-chng | F | df1 | df2 | p |
|-------|---------|---------|--------|----------|-------|
| int_1 | .0042 | 23.0703 | 1.0000 | 852.0000 | .0000 |

Conditional effect of X on Y at values of the moderator(s):

| Technolo | Effect | se | t | p | LLCI | ULCI |
|----------|--------|-------|---------|-------|-------|-------|
| 2.7372 | .7339 | .0230 | 31.8919 | .0000 | .6888 | .7791 |
| 3.8107 | .6692 | .0237 | 28.2843 | .0000 | .6228 | .7156 |
| 4.8843 | .6045 | .0309 | 19.5792 | .0000 | .5439 | .6651 |

Values for quantitative moderators are the mean and plus/minus one SD from mean.

Values for dichotomous moderators are the two values of the moderator.

***** JOHNSON-NEYMAN TECHNIQUE *****

There are no statistical significance transition points within the observed range of the moderator.

Data for visualizing conditional effect of X on Y

Paste text below into a SPSS syntax window and execute to produce plot.

DATA LIST FREE/Pro_Env_Strgy Technological_Advan Operational_Performance.
BEGIN DATA.

| | | |
|--------|--------|--------|
| 2.7685 | 2.7372 | 2.7922 |
| 3.8879 | 2.7372 | 3.6137 |
| 5.0072 | 2.7372 | 4.4352 |
| 2.7685 | 3.8107 | 3.0663 |
| 3.8879 | 3.8107 | 3.8154 |
| 5.0072 | 3.8107 | 4.5644 |
| 2.7685 | 4.8843 | 3.3405 |
| 3.8879 | 4.8843 | 4.0171 |
| 5.0072 | 4.8843 | 4.6937 |

END DATA.

GRAPH/SCATTERPLOT=Pro_Env_Strgy WITH Operational_Performance BY
Technological_Advan.

***** ANALYSIS NOTES AND WARNINGS *****

Level of confidence for all confidence intervals in output:
95.00

----- END MATRIX -----

restore.

Matrix

Notes

| | | | |
|----------------|--------------------------------|----------------------|--|
| Output Created | | 20-OCT-2020 16:03:36 | |
| Comments | | | |
| Input | Active Dataset | DataSet1 | |
| | Filter | <none> | |
| | Weight | <none> | |
| | Split File | <none> | |
| | N of Rows in Working Data File | 856 | |

Syntax

```
matrix.  
get dat/file = */variables =  
Operational_Performance Plan_Orga_Pract  
Technological_Advan /names =  
vnames/missing = 9999.  
compute ninit = nrow(dat).  
get dat/file = */variables =  
Operational_Performance Plan_Orga_Pract  
Technological_Advan /names =  
vnames/missing = omit.  
get tmp/file = */variables =  
Operational_Performance /names =  
ynames/missing = omit.  
get tmp2/file = */variables = Plan_Orga_Pract  
/names = xnames/missing = omit.  
get tmp/file = */variables =  
Technological_Advan /names =  
mnames/missing = omit.  
get tmp/file = */variables = w999999t z999999t  
v999999t q999999t.  
compute wname=tmp(1,1).  
do if (wname = ' ').  
compute wname = 'xxx'.  
end if.  
compute zname=tmp(1,2).  
do if (zname = ' ').  
compute zname = 'xxx'.  
end if.  
compute vname=tmp(1,3).  
do if (vname = ' ').  
compute vname = 'xxx'.  
end if.  
compute qname=tmp(1,4).  
do if (qname = ' ').  
compute qname = 'xxx'.  
end if.  
compute n = nrow(dat).  
compute p0=-.322232431088.  
compute p1 = -1.  
compute p2 = -.342242088547.
```

```

compute p3 = -.0204231210245.
compute p4 = -.0000453642210148.
compute q0 = .0993484626060.
compute q1 = .588581570495.
compute q2 = .531103462366.
compute q3 = .103537752850.
compute q4 = .0038560700634.
compute badend = 0.
compute priorlo = -9999999.
compute priorhi = 9999999.
compute criterr = 0.
compute cluster = 0.
compute clsdmy = 0.
compute jndich = 0.
compute wvdich=0.
compute zqdich=0.
compute mod74dic=0.
compute warnrep=0.
compute contrast = trunc( 0 ).
do if (contrast > 2 or contrast < 0).
compute contrast = 0.
end if.
compute booterr = 0.
compute effsize = ( 0 = 1).
compute note = make(10,1,0).
compute notes = 1.
compute iterr = 0.
compute clsmth = 0.
compute quantile = ( 0 = 1).
compute mmodval= 999.
compute jn = ( 1 = 1).
compute covcoeff=( 0 =1).
compute center = ( 0 = 1).
compute detail = ( 1 = 1).
compute coeffci = ( 1 = 1).
compute conf = 95.
compute longname = ( 1 =1).
compute bconoff=( 0 <> 1).
compute covmy = trunc( 0 ).
do if (covmy < 0 or covmy > 2).

```

```

compute covmy = 0.
end if.
do if (trunc(conf) >= 100 or (trunc(conf) <=
50)).
compute conf = 95.
compute note(notes,1) = 1.
compute notes = notes + 1.
end if.
do if (n < ninit).
compute nmiss = ninit-n.
compute note(notes,1) = 11.
compute notes = notes + 1.
end if.
compute errs = 0.
compute quantd = {0,0,0,0,0,0}.
compute quantc = {0,0,0,0,0,0}.
compute mcheck = 0.
compute ttt = 0.
compute plot = trunc( 1 ).
compute runerrs = make(50,1,0).
compute model = trunc( 1 ).
compute normal = 0.
compute ws= ( 0 =1).
compute olsdichm= ( 0 =1).
compute olsdichy= ( 0 =1).
compute mcx=trunc( 0 ).
compute mcm=trunc( 0 ).
do if (ws = 1).
do if (effsize = 1).
compute note(notes,1) = 19.
compute notes = notes + 1.
compute effsize=0.
end if.
do if (normal=1).
compute note(notes,1) = 16.
compute notes = notes + 1.
compute normal=0.
end if.
end if.
do if (model <> 4).

```

```

compute ws = 0.
end if.
do if (model > 0 and model < 4 and covmy <>
0).
compute covmy=0.
compute note(notes,1) = 33.
compute notes = notes + 1.
end if.
print/space=0.
do if (jn = 1 and model <> 1 and model <> 3).
compute note(notes,1) = 7.
compute notes = notes + 1.
end if.
do if ((model > 76) or (model < 1)).
compute model = 77.
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 19.
end if.
compute itprob=0.
do if (mcx < 0).
compute mcx=0.
end if.
do if (mcx > 4).
compute mcx=0.
end if.
do if (mcm < 0).
compute mcm=0.
end if.
do if (mcm > 4).
compute mcm=0.
end if.
do if (mcx > 0 and mcm > 0).
compute errs=errs+1.
compute runerrs(errs,1)=34.
compute criterr=1.
end if.
do if ((model <> 1 and model <> 4) and (mcx >
0 or mcm > 0)).
compute errs=errs+1.

```

```

compute runerrs(errs,1)=35.
compute criterr=1.
end if.
do if ((model = 4) and (mcm > 0)).
compute errs=errs+1.
compute runerrs(errs,1)=41.
compute criterr=1.
end if.
do if ((mcx > 0) and jn <> 0).
compute jn=0.
compute note(notes,1) = 23.
compute notes = notes + 1.
end if.
do if (mmodval <> 999 and mcm > 0).
compute note(notes,1) = 24.
compute notes = notes + 1.
end if.
compute toteff = 0.
compute toteff = (( 0 = 1)*( 1 = 4 or 1 = 6)).
compute varorder = 2.
compute hc3 = ( 0 <> 0).
compute cname = 'xxx'.
compute centvar = {'xxx'}.
compute nmeds = ncol(mnames).
do if (longname=0).

```

```
compute criterr = 1.  
compute errs = errs+1.  
compute runerrs(errs,1) = 33.
```

1

2

3

end if.

compute modelm =

```
{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,6;
```

1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,7;
1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,8;
1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,9;
1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,3,10;
1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,5,11;
1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,5,12;
1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,5,13;
0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,2,14;
0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,2,15;
0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,4,16;
0,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,4,17;
0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,6,18;
0,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,6,19;
0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,6,20;
1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,7,21;
1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,7,22;
1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,23;
1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,0,24; 1,1,1,1

,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,25;
1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,0,0,26;
1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,0,0,27;
1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,7,28;
1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,0,7,29;
1,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,30;
1,1,0,1,0,0,1,1,0,1,0,0,0,0,0,0,0,31;
1,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,0,32;
1,1,1,1,0,0,1,1,1,1,0,0,0,0,0,0,0,33;
1,1,1,1,0,0,1,0,0,1,0,0,0,0,0,0,0,34;
1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,35;
1,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,36;
1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,37;
1,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,38;
1,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,39;
1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,0,40;
1,0,0,1,1,0,1,0,0,1,1,0,0,0,0,0,0,41;
1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,42;
1,0,0,1,1,1,1,0,0,1,1,1,0,0,0,0,0,43;
1,0,0,1,1,1,1,0,0,1,0,0,0,0,0,0,0,44;
1,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,45;

1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,46;
1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,47;
1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,48;
1,1,0,1,1,0,1,1,0,0,0,

0,0,0,0,0,0,0,0,49;
1,1,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,50;
1,1,1,1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,51;
1,1,1,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,52;
1,1,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,53;
1,1,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,54;
1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,55;
1,1,1,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,56;
1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,57;
1,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,58;
1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,59;
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,60;
1,1,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,61;
1,1,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,62;
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,0,0,0,0,63;
1,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,64;
1,0,0,1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,65;
1,0,0,1,0,0,0,0,0,1,0,0,1,0,0,0,0,0,0,66;
1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,67;
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,68;
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,69;
1,0,0,1,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,70;
1,0,0,1,0,0,1,0,0,1,0,0,1,1,1,0,0,0,0,71;
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,1,1,0,0,72;
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,1,1,0,0

,73; 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,74;

1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,75;

1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,1,0,0,0,76;

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,77).

compute wm = modelm(model, 1).

compute zm = modelm(model, 2).

compute wzm = modelm(model, 3).

compute vy = modelm(model, 4).

compute qy = modelm(model, 5).

```

compute vqy = modelm(model, 6).
compute wy = modelm(model, 7).
compute zy = modelm(model, 8).
compute wzy = modelm(model, 9).
compute vxy = modelm(model, 10).
compute qxy = modelm(model, 11).
compute vqxy = modelm(model, 12).
compute wmy = modelm(model, 13).
compute wvmy = modelm(model, 14).
compute wvxy = modelm(model, 15).
compute zmy = modelm(model, 16).
compute wzmy = modelm(model, 17).
compute xmy = modelm(model, 18).
compute imm = modelm(model, 19).
do if (model = 74).
compute vdich=0.
end if.
do if (ncol(xname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 20.
compute criterr = 1.
end if.
do if (ncol(mnames) > 10).
compute errs = errs+1.
compute runerrs(errs,1) = 39.
compute criterr = 1.
end if.
do if (ncol(yname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 21.
compute criterr = 1.
end if.
do if (( 999 =0) and ( 999 =0) and (model=3 or
model=2) and (contrast=1)).
compute errs = errs+1.
compute runerrs(errs,1) = 31.
compute criterr = 1.
end if.
compute xlist = (wm or zm or wzm or wy or zy
or wzy or vxy or qxy or vqxy or wvxy or xmy).

```

```

compute mlist = (vy or qy or vqy or zmy or wmy
or wzmy or xmy or (model < 4)).
compute bad=0.
do if (criterr = 0).
compute werr = 0.
compute verr = 0.
compute qerr = 0.
compute zerr = 0.
compute yerr = 1.
compute xerr = 1.
compute wlist = (wm or wzm or wy or wzy or
wm or wvmy or wvmy or wvxy or wzmy).
do if (wlist = 1 and wname = 'xxx').
compute werr = 1.
compute wlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (wlist = 1 and ((wname = qname) or
(wname = vname) or (wname = zname) or
(wname = xname) or (wname = yname))).
compute werr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 12).
end if.
compute zlist = (zm or wzm or zy or wzy or
zmy or wzmy).
do if (zlist = 1 and zname = 'xxx').
compute zerr = 1.
compute zlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (zlist = 1 and ((zname = qname) or
(zname = vname) or (zname = wname) or
(zname = xname) or (zname = yname))).
compute zerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 13.
end if.

```

```

compute qlist = (qy or vqy or qxy or vqxy).
do if (qlist = 1 and qname = 'xxx').
compute qerr = 1.
compute qlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (qlist = 1 and ((qname = zname) or
(qname = vname) or (qname = wname) or
(qname = xname) or (qname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 14.
end if.
compute vlist = (vy or vqy or vxy or vqxy or
wvmy or wvxy).
do if (vlist = 1 and vname = 'xxx').
compute verr = 1.
compute vlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (vlist = 1 and ((vname = zname) or
(vname = qname) or (vname = wname) or
(vname = xname) or (vname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 15.
end if.
do if (wlist = 0 and wname <> 'xxx').
compute werr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 8.
end if.
do if (zlist = 0 and zname <> 'xxx').
compute zerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 9.
end if.
do if (qlist = 0 and qname <> 'xxx').

```

```

compute qerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 10.
end if.
do if (vlist = 0 and vname <> 'xxx').
compute verr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 11.
end if.
do if (hc3 = 1).
compute note(notes,1) = 3.
compute notes = notes+1.
end if.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+(((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute cons = make(n,1,1).
compute temp = (n*sscp(dat))-
(t(csum(dat))*(csum(dat))).
compute temp = temp/(n*(n-1)).
compute temp = csum(((diag(temp))=0)).
compute temp2=1.
do if (ws = 1).
compute temp2 = (n*sscp(tmp2))-
(t(csum(tmp2))*(csum(tmp2))).
compute temp2 = temp2/(n*(n-1)).
end if.
do if ((temp > 0 and ws = 0) or (temp > 0 and
ws = 1 and temp2 > 0)).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 27.
end if.
compute nmeds = ncol(mnames).
compute mcmats=ident(nmeds*2).
compute mccoef=make(nmeds*2,1,0).
compute sobel = make(nmeds,4,-999).
do if (model = 6 and nmeds > 4).

```

```

compute errs = errs+1.
compute runerrs(errs,1)=2.
end if.
do if (model < 4 and nmeds > 1).
compute errs = errs+1.
compute runerrs(errs,1) = 3.
end if.
compute nmods = (model = 74).
compute bad = 0.
compute intcnt = 1.
compute modvals = 0.
compute modvalsd = 0.
compute yintemp = {'int_1', 'int_2', 'int_3',
'int_4', 'int_5', 'int_6', 'int_7', 'int_8', 'int_9',
'int_10', 'int_11', 'int_12', 'int_13', 'int_14',
'int_15'}.
compute yintemp = {yintemp, 'int_16', 'int_17',
'int_18', 'int_19', 'int_20', 'int_21', 'int_22',
'int_23', 'int_24', 'int_25', 'int_26', 'int_27',
'int_28'}.
compute yintemp = {yintemp, 'int_29', 'int_30',
'int_31', 'int_32', 'int_33', 'int_34', 'int_35',
'int_36', 'int_37', 'int_38', 'int_39', 'int_40'}.
compute yintemp = {yintemp, 'int_41', 'int_42',
'int_43', 'int_44', 'int_45', 'int_46', 'int_47',
'int_48', 'int_49', 'int_50', 'int_51', 'int_52'}.
compute yintemp = {yintemp, 'int_53', 'int_54',
'int_55', 'int_56', 'int_57', 'int_58', 'int_59',
'int_60', 'int_61', 'int_62', 'int_63', 'int_64'}.
compute
cntname={'(C1)';'(C2)';'(C3)';'(C4)';'(C5)';'(C6)';'(
C7)';'(C8)';'(C9)';'(C10)';'(C11)';'(C12)';'(C13)';(
C14)';'(C15)';'(C16)';'(C17)'}.
compute
cntname={cntname;'(C18)';'(C19)';'(C20)';'(C21
)';'(C22)';'(C23)';'(C24)';'(C25)';'(C26)';'(C27)';(
C28)';'(C29)';'(C30)';'(C31)'}.
compute
cntname={cntname;'(C32)';'(C33)';'(C34)';'(C35

```



```

);'(C36)';'(C37)';'(C38)';'(C39)';'(C40)';'(C41)';'(
C42)';'(C43)';'(C44)';'(C45)}.

compute
cntname={cntname;'(C46)';'(C47)';'(C48)';'(C49
)';'(C50)';'(C51)';'(C52)';'(C53)';'(C54)';'(C55)';'(
C56)';'(C57)';'(C58)';'(C59)}.

compute
cntname={cntname;'(C60)';'(C61)';'(C62)';'(C63
)';'(C64)';'(C65)';'(C66)';'(C67)';'(C68)';'(C69)';'(
C70)';'(C71)';'(C72)';'(C73)}.

compute
cntname={cntname;'(C74)';'(C75)';'(C76)';'(C77
)';'(C78)';'(C79)';'(C80)';'(C81)';'(C82)';'(C83)';'(
C84)';'(C85)';'(C86)';'(C87)}.

compute
cntname={cntname;'(C88)';'(C89)';'(C90)';'(C91
)';'(C92)';'(C93)';'(C94)';'(C95)';'(C96)';'(C97)';'(
C98)';'(C99)';'(C100)';'(C101)}.

compute
cntname={cntname;'(C102)';'(C103)';'(C104)';'(
C105)}.

compute apathnam={'a path'; 'a1 path'; 'a2
path'; 'a3 path'; 'a4 path'; 'a5 path'; 'a6 path';
'a7 path'; 'a8 path'; 'a9 path'; 'a10 path'}.

compute bpathnam={'b path'; 'b1 path'; 'b2
path'; 'b3 path'; 'b4path'; 'b5path'; 'b6 path'; 'b7
path'; 'b8 path'; 'b9 path'; 'b10 path'}.

compute modvnm = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.

compute modvnm2 = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.

compute mlab = { ' M1 ='; ' M2 ='; ' M3 ='; '
M4 ='; ' M5 ='; ' M6 ='; ' M7 ='; ' M8 ='; '
M9 ='; ' M10 ='}.

compute m = make(n,nmeds,1).
compute mmat = make(16,nmeds,0).
compute ymat = make(8,nmeds,0).
compute deco = make(10,1,0).
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).

```

```

compute modmatp = make(1,5,0).
compute modprod = modmatv.
compute iterate = abs(trunc( 10000 )).
compute converge = abs( 0.00000001 ).
compute boot = abs(trunc( 5000 )).
do if (ws=1 and 0 > 0).
compute note(notes,1)=17.
compute notes=notes+1.
end if.
compute mc=(abs(trunc( 0 ))*(1-ws)).
do if (mc > 0 and model > 5).
do if (boot = 0).
compute boot=mc.
end if.
compute mc=0.
compute note(notes,1)=12.
compute notes = notes+1.
end if.
compute adjust = 0.
do if (mc > 0 and (model > 3 and model < 6)).
compute boot=0.
compute bconoff=0.
end if.
do if (boot > 0 and mc > 0).
compute mc = 0.
end if.
compute savboot = 0.
compute saveboot = ( 0 = 1).
do if (ws=1 and saveboot=1).
compute saveboot=0.
compute note(notes,1) = 20.
compute notes = notes + 1.
compute normal=0.
end if.
do if (saveboot =1 and boot > 0 and model >
3).
compute savboot = 1.
end if.
do if (boot <> 0 or mc <> 0).
compute bootsz=boot.

```

```

do if (mc > 0).
compute bootsz=mc.
end if.
loop.
compute cilow = rnd(bootSZ*(1-(conf/100))/2).
compute cihigh =
trunc((bootSZ*(conf/100)+(bootSZ*(1-
(conf/100))/2)))+1.
do if (cilow < 1 or cihigh > bootSZ).
compute
bootSZ=trunc((bootSZ+1000)/1000)*1000.
compute adjust = 1.
end if.
end loop if (cilow gt 0 and cihigh le bootSZ).
compute boot=bootSZ.
do if (mc > 0).
compute mc = bootSZ.
end if.
do if (boot > 0 and mc > 0 and (model > 3 and
model < 6)).
compute boot=0.
end if.
do if (adjust = 1 and boot > 0).
compute note(notes,1)=6.
compute notes = notes+1.
end if.
do if (adjust = 1 and mc > 0).
compute note(notes,1)=13.
compute notes = notes+1.
end if.
end if.
do if (model = 6 and nmeds > 1).
compute mmpaths =
make((nmeds+2),(nmeds+2),0).
do if (nmeds = 2).
compute indboot = make(boot+1, 3, 999).
else if (nmeds = 3).
compute indboot = make(boot+1, 7, 999).
else if (nmeds = 4).
compute indboot = make(boot+1, 15, 999).

```

```

end if.
compute indlbl = {'Total:','Ind1 :','Ind2 :','Ind3
:','Ind4 :','Ind5 :','Ind6 :','Ind7 :','Ind8 :','Ind9
:','Ind10:','Ind11:','Ind12:','Ind13:','Ind14:','
Ind15:'}.
compute indlbl2 = {'Ind1','Ind2','Ind3','Ind4';
'Ind5','Ind6','Ind7','Ind8','Ind9','Ind10';
'Ind11','Ind12','Ind13','Ind14','Ind15'}.
compute indces = make(boot+1, 4, 999).
end if.
do if (model < 4).
compute boot = 0.
compute cmat = make(10,1,0).
compute zmat = make(10,1,0).
end if.
compute nvarch = make(1,ncol(dat),0).
compute wmatch = 0.
compute zmatch = 0.
compute vmatch = 0.
compute qmatch = 0.
compute mmatch=0.
compute minprobe=0.
compute maxprobe=0.
loop i = 1 to ncol(vnames).
do if (vnames(:,i)=yname).
compute y = dat(:,i).
compute nvarch(1,i)=1.
compute yerr = 0.
do if ((yname = xname) or (yname = wname) or
(yname = zname) or (yname = vname) or
(yname = qname)).
compute errs = errs+1.
compute runerrs(errs,1)=17.
end if.
end if.
do if (vnames(:,i)=xname).
compute x = dat(:,i).
compute nvarch(1,i)=1.
compute xdich = 1.
compute xerr=0.

```

```

loop jj = 1 to n.
do if ((x(jj,1) <> cmin(x)) and (x(jj,1) <>
cmax(x))).
compute xdich = 0.
break.
end if.
end loop.
compute xmean = csum(x)/n.
do if (center = 1 and (model < 4 or xlist > 0)
and mcx = 0).
compute meanvec = make(n,1, xmean).
compute x = x-meanvec.
compute centvar = {centvar, xname}.
end if.
compute xmean = csum(x)/n.
compute tmp = x-(cons*xmean).
compute xsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
do if (xdich = 0).
compute quantc(1,6) = 1.
compute matx = {xmean-xsd; xmean;
xmean+xsd}.
do if ( 999 = 999 and quantile = 0).
do if ((matx(1,1) < cmin(x)) and model = 74).
compute matx(1,1)=cmin(x).
compute minprobe=1.
end if.
do if ((matx(3,1) > cmax(x)) and model = 74).
compute matx(3,1)=cmax(x).
compute maxprobe=1.
end if.
end if.
do if (quantile = 1).
compute quantd(1,6) = 1.
compute quantc(1,6) = 0.
compute tmp = x.
compute tmp(GRADE(x)) = x.
compute matx =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.

```

```

end if.
end if.
do if (xdich = 1).
compute matx = {cmin(x); cmax(x)}.
do if (model=74).
compute matx=cmin(x).
compute mod74dic=1.
end if.
end if.
do if ( 999 <> 999).
compute matx = 999.
compute quantd(1,6) = 0.
compute quantc(1,6) = 0.
end if.
end if.
do if (werr = 0 and wlist = 1).
do if (vnames(:,i)=wname).
compute werr = 0.
compute wmatch = 1.
compute w = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(w)/n).
compute w = w-meanvec.
compute centvar = {centvar, wname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute wmean = csum(w)/n.
compute tmp = w-(cons*wmean).
compute wsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute wdich = 1.
loop jj = 1 to n.
do if ((w(jj,1) <> cmin(w)) and (w(jj,1) <>
cmax(w))).
compute wdich = 0.
break.
end if.
end loop.
do if (model = 3).
compute jndich=wdich.

```

```

compute jnmin=cmin(w).
compute jnmax=cmax(w).
end if.
do if (wdich = 0).
compute matw = {wmean-wsd; wmean;
wmean+wsd}.
do if ( 999 = 999 and quantile = 0).
do if (matw(1,1) < cmin(w)).
compute matw(1,1)=cmin(w).
compute minprobe=1.
end if.
do if (matw(3,1) > cmax(w)).
compute matw(3,1)=cmax(w).
compute maxprobe=1.
end if.
end if.
compute quantc(1,1) = 1.
do if (quantile = 1).
compute quantd(1,1) = 1.
compute quantc(1,1) = 0.
compute tmp = w.
compute tmp(GRADE(w)) = w.
compute matw =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (wdich = 1).
compute matw = {cmin(w); cmax(w)}.
compute wvdich=1.
compute cmaxw=cmax(w).
compute cminw=cmin(w).
end if.
do if ( 999 <> 999).
compute matw = 999.
compute quantd(1,1) = 0.
compute quantc(1,1) = 0.
end if.
compute modmatv(1,1)=nrow(matw).

```

```

compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm(1,1)=wname.
compute modmatp(1,1) = 1.
end if.
end if.
do if (zerr = 0 and zlist = 1).
do if (vnames(:,i)=zname).
compute zerr = 0.
compute zmatch = 1.
compute z = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(z)/n).
compute z = z-meanvec.
compute centvar = {centvar, zname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute zmean = csum(z)/n.
compute tmp = z-(cons*zmean).
compute zsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute zdich = 1.
loop jj = 1 to n.
do if ((z(jj,1) <> cmin(z)) and (z(jj,1) <>
cmax(z))).
compute zdich = 0.
break.
end if.
end loop.
do if (zdich = 0).
compute matz = {zmean-zsd; zmean;
zmean+zsd}.
do if ( 999 = 999 and quantile = 0).
do if (matz(1,1) < cmin(z)).
compute matz(1,1)=cmin(z).
compute minprobe=1.
end if.
do if (matz(3,1) > cmax(z)).
compute matz(3,1)=cmax(z).
compute maxprobe=1.
end if.

```



```

end if.
compute quantc(1,2) = 1.
do if (quantile = 1).
compute quantd(1,2) = 1.
compute quantc(1,2) = 0.
compute tmp = z.
compute tmp(GRADE(z)) = z.
compute matz = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (zdich = 1).
compute matz = {cmin(z); cmax(z)}.
compute cmaxz=cmax(z).
compute cminz=cmin(z).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matz = 999.
compute quantd(1,2) = 0.
compute quantc(1,2) = 0.
end if.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm(1,2)=zname.
compute modmatp(1,2) = 1.
end if.
end if.
do if (verr = 0 and vlist = 1).
do if (vnames(:,i)=vname).
compute verr = 0.
compute vmatch = 1.
compute v = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(v)/n).
compute v = v-meanvec.
compute centvar = {centvar, vname}.
end if.
compute nvarch(1,i)=1.

```

```

compute nmods = nmods + 1.
compute vmean = csum(v)/n.
compute tmp = v-(cons*vmean).
compute vsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute vdich = 1.
loop jj = 1 to n.
do if ((v(jj,1) <> cmin(v)) and (v(jj,1) <>
cmax(v))).
compute vdich = 0.
break.
end if.
end loop.
do if (vdich = 0).
compute matv = {vmean-vsd; vmean;
vmean+vsd}.
do if ( 999 = 999 and quantile = 0).
do if (matv(1,1) < cmin(v)).
compute matv(1,1)=cmin(v).
compute minprobe=1.
end if.
do if (matv(3,1) > cmax(v)).
compute matv(3,1)=cmax(v).
compute maxprobe=1.
end if.
end if.
compute quantc(1,3) = 1.
do if (quantile = 1).
compute quantd(1,3) = 1.
compute quantc(1,3) = 0.
compute tmp = v.
compute tmp(GRADE(v)) = v.
compute matv = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (vdich = 1).
compute matv = {cmin(v); cmax(v)}.
compute wvdich=1.
compute cmaxv=cmax(v).

```

```

compute cminv=cmin(v).
end if.
do if ( 999 <> 999).
compute matv = 999.
compute quantd(1,3) = 0.
compute quantc(1,3) = 0.
end if.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm(1,3)=vname.
compute modmatp(1,3) = 1.
end if.
end if.
do if (qerr = 0 and qlist = 1).
do if (vnames(:,i)=qname).
compute qerr = 0.
compute qmatch = 1.
compute q = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(q)/n).
compute q = q-meanvec.
compute centvar = {centvar, qname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute qmean = csum(q)/n.
compute tmp = q-(cons*qmean).
compute qsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute qdich = 1.
loop jj = 1 to n.
do if ((q(jj,1) <> cmin(q)) and (q(jj,1) <>
cmax(q))).
compute qdich = 0.
break.
end if.
end loop.
do if (qdich = 0).
compute matq = {qmean-qsd; qmean;
qmean+qsd}.
do if ( 999 = 999 and quantile = 0).

```

```

do if (matq(1,1) < cmin(q)).
compute matq(1,1)=cmin(q).
compute minprobe=1.
end if.
do if (matq(3,1) > cmax(q)).
compute matq(3,1)=cmax(q).
compute maxprobe=1.
end if.
end if.
compute quantc(1,4) = 1.
do if (quantile = 1).
compute quantd(1,4) = 1.
compute quantc(1,4) = 0.
compute tmp = q.
compute tmp(GRADE(q)) = q.
compute matq = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (qdich = 1).
compute matq = {cmin(q); cmax(q)}.
compute cmaxq=cmax(q).
compute cminq=cmin(q).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matq = 999.
compute quantd(1,4) = 0.
compute quantc(1,4) = 0.
end if.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm(1,4)=qname.
compute modmatp(1,4) = 1.
end if.
end if.
do if (vnames(:,i)=cname).
compute cld = dat(:,i).
compute cvname = vnames(:,i).

```

```

compute nvarch(1,i)=1.
compute clsmatch = 1.
end if.
loop j = 1 to ncol(mnames).
do if (vnames(:,i)=mnames(1,j)).
compute mmatch = mmatch + 1.
compute m(:,j)=dat(:,i).
do if (center = 1 and nvarch(1,i) = 0 and mlist >
0 and mcm = 0).
compute meanvec = make(n,1,csum(m(:,j))/n).
compute m(:,j) = m(:,j)-meanvec.
compute centvar = {centvar, mnames(1,j)}.
end if.
compute nvarch(1,i)=1.
compute dichm = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute dichm = 0.
break.
end if.
end loop.
do if ((dichm = 1) and (olsdichm = 1)).
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichm = 1 and model > 3 and mcheck = 0
and olsdichm = 0).
compute errs= errs+1.
compute runerrs(errs,1) = 1.
compute mcheck = 1.
end if.
do if ((model <= 3) and (ncol(mnames)=1)).
compute nmods = nmods + 1.
compute mmean = csum(m(:,j))/n.
compute tmp = m(:,j)-(cons*mmean).
compute msd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute mdich = 1.
loop jj = 1 to n.

```

```

do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute mdich = 0.
break.
end if.
end loop.
do if (model = 1).
compute jndich=mdich.
compute jnmin=cmin(m(:,j)).
compute jnmax=cmax(m(:,j)).
end if.
do if (mdich = 0).
compute matm = {mmean-msd; mmean;
mmean+msd}.
do if ( 999 = 999 and quantile = 0).
do if (matm(1,1) < cmin(m(:,j))).
compute matm(1,1)=cmin(m(:,j)).
compute minprobe=1.
end if.
do if (matm(3,1) > cmax(m(:,j))).
compute matm(3,1)=cmax(m(:,j)).
compute maxprobe=1.
end if.
end if.
compute quantc(1,5) = 1.
do if (quantile = 1).
compute quantd(1,5) = 1.
compute quantc(1,5) = 0.
compute tmp = m(:,j).
compute tmp(GRADE(m(:,j))) = m(:,j).
compute matm = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (mdich = 1).
compute matm = {cmin(m); cmax(m)}.
end if.
do if ( 999 <> 999).
compute matm = 999.

```

```

compute quantd(1,5) = 0.
compute quantc(1,5) = 0.
end if.
compute modmatv(1,5)=nrow(matm).
compute modmat((1:nrow(matm)), 5) = matm.
compute modvnm(1,5)=mnames(1,j).
compute modmatp(1,5) = 1.
end if.
end if.
end loop.
end loop.
do if (minprobe=1).
compute note(notes,1) = 14.
compute notes = notes + 1.
end if.
do if (maxprobe=1).
compute note(notes,1) = 15.
compute notes = notes + 1.
end if.
do if (cname <> 'xxx' and clsmatch = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 23.
end if.
do if (cname <> 'xxx').
do if ((cname = zname) or (cname = vname)
or (cname = wname) or (cname = xname) or
(cname = yname) or (cname = qname)).
compute errs = errs+1.
compute runerrs(errs,1) = 24.
end if.
end if.
do if (wlist = 1 and werr = 0 and wmatch = 0).
compute werr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (zlist = 1 and zerr = 0 and zmatch = 0).
compute zerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 5.

```

```

end if.
do if (qlist = 1 and qerr = 0 and qmatch = 0).
compute qerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (vlist = 1 and verr = 0 and vmatch = 0).
compute verr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (yerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 16.
end if.
do if (xerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 32.
end if.
do if (model = 6 and nmeds < 2).
compute errs = errs+1.
compute runerrs(errs,1) = 18.
end if.
do if (mmatch < ncol(mnames)).
compute errs=errs+1.
compute runerrs(errs,1) = 25.
end if.
do if (cname <> 'xxx').
compute cld = design(cld).
compute cluster = ncol(cld).
compute cld = cld(:,2:ncol(cld)).
compute clsdmy = ncol(cld).
do if (clsdmy > 19).
compute errs = errs+1.
compute runerrs(errs,1) = 26.
end if.
do if (mcx > 0 or mcm > 0).
compute errs = errs+1.
compute runerrs(errs,1) = 38.
end if.

```



```

end if.
compute dichy = 1.
loop jj = 1 to n.
do if ((y(jj,1) <> cmin(y)) and (y(jj,1) <>
cmax(y))).
compute dichy = 0.
break.
end if.
end loop.
do if ((dichy = 1) and (olstdichy = 1)).
compute dichy=0.
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichy = 1).
do if (toteff=1).
compute toteff=0.
compute note(notes,1) = 31.
compute notes = notes + 1.
end if.
compute jncrit=xp2*xp2.
do if (ws = 1).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 28.
end if.
end if.
compute ncovs = ncol(dat)-rsum(nvarch).
do if (effsize = 1 and covmy <> 0 and ncovs >
0 and model > 3 and model < 7).
compute note(notes,1) = 22.
compute notes = notes + 1.
compute effsize = 0.
end if.
do if (ws = 1 and ncovs < nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 29.
end if.
do if (ws = 1 and ncovs > nmeds).

```

```

compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 30.
end if.
end if.
do if (errs = 0).
do if (rsum(quantd) > 0).
compute note(notes,1) = 4.
compute notes = notes+1.
end if.
do if (rsum(quantc) > 0).
compute note(notes,1) = 5.
compute notes = notes+1.
end if.
do if (ncovs > 0).
compute c = make(n,ncovs,0).
compute cnames = {'x'}.
compute j = 1.
loop i = 1 to ncol(vnames).
do if (nvarch(1,i)) = 0.
compute c(:,j) = dat(:,i).
compute nvarch(1,i)=1.
compute j=j+1.
compute cnames = {cnames, vnames(:,i)}.
end if.
end loop.
compute cnames = cnames(1,2:ncol(cnames)).
do if (ws = 1).
compute covmean=csum(c)/n.
loop i = 1 to ncovs.
compute meanvec = make(n,1, covmean(1,i)).
compute c(:,i) = c(:,i)-meanvec.
end loop.
compute centvar = {centvar, cnames}.
compute x=c(:,1:ncovs).
do if (ncovs=nmeds).
compute ncovs=0.
end if.
end if.
end if.

```

```

compute names = {yname, xname, mnames,
wname, zname, vname, qname}.
do if (ncovs > 0).
compute names = {names, cnames}.
end if.
do if (dichy = 1 and effsize = 1).
compute note(notes,1) = 2.
compute notes = notes+1.
end if.
do if (model > 3 and model < 6).
compute indeff=make(nmeds,1,0).
compute indboot=make(boot+1,nmeds,999).
do if (mc > 0).
compute indboot=make(mc+1,nmeds,999).
end if.
do if (effsize = 1 and dichy = 0).
compute rmeff=make(boot+1,nmeds+1,999).
compute abpseff=make(boot+1,nmeds+1,999).
compute abcseff=make(boot+1,nmeds+1,999).
compute pmeff=make(boot+1,nmeds+1,999).
compute r245 = make(boot+1,1,999).
compute kappa2 = make(boot+1,1,999).
end if.
end if.
do if (model = 6 and effsize = 1 and dichy = 0).
compute
rmeff=make(boot+1,ncol(indboot),999).
compute
abpseff=make(boot+1,ncol(indboot),999).
compute
abcseff=make(boot+1,ncol(indboot),999).
compute
pmeff=make(boot+1,ncol(indboot),999).
end if.
do if (nmods > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm(1,tmp) = modvnm(1,i).

```

```

compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:nmods).
compute modvnm=modvnm(:,1:nmods).
compute modmatv=modmatv(:,1:nmods).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvals =
make((modmatv(1,1)*modprod(1,1)),
nmods,0).
loop i = 1 to nmods.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvals)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvals(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
do if (model = 74).
compute modvals=matx.
compute modvnm=xname.
end if.
compute vmat = make(8,nrow(modvals),0).
compute vmat(1,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute vmat(5,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute indeff=make(nrow(modvals),1,0).

```

```

do if (model <> 5).
compute indboot = make(((boot+1)*nmeds),
nrow(modvals), -99999999).
compute indbootp =
make(boot+1,(nmeds*nmods), -99999999).
do if (imm > 4 and imm < 8).
do if (imm = 7).
compute immmv1=matw.
compute immmv2=matv.
end if.
do if (imm = 5).
compute immmv1=matw.
compute immmv2=matz.
end if.
do if (imm = 6).
compute immmv1=matv.
compute immmv2=matq.
end if.
compute
indboop1=make((boot+1),(nmeds*(nrow(immmv2))),-99999999).
compute
indboop2=make((boot+1),(nmeds*(nrow(immmv1))),-99999999).
compute onesmv1=make(1,nrow(immmv1),1).
compute onesmv2=make(1,nrow(immmv2),1).
compute
nimvs=nrow(immmv1)+nrow(immmv2).
compute cmmval=0.
compute cmmlbs1={'xx'}.
compute cmmlbs2={'xx'}.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv2}.
compute
cmmlbt=make(nrow(immmv2),1,mnames(1,i)).
compute cmmlbs1={cmmlbs1;cmmlbt}.
end loop.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv1}.

```

```

compute
cmm1bt=make(nrow(immmv1),1,mnames(1,i)).
compute cmm1bs2={cmm1bs2;cmm1bt}.
end loop.
compute cmmval=cmmval(2:nrow(cmmval),:).
end if.
end if.
end if.
do if (nmods> 0).
loop i = 1 to ncol(modvals).
do if (modvnm(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm(1,i)=qname).
compute qcol = i.
end if.
end loop.
end if.
do if (dichy = 1).
compute omx = cmax(y).
compute omn = cmin(y).
compute y = (y = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute data = {cons,y,m,x}.
compute datamed = data.
compute datayed = data.
compute datanm = {'constant'; yname;
t(mnames); xname}.
compute datanmm = {'constant'; yname;
t(mnames); xname}.
compute datanmy = {'constant'; yname;
t(mnames); xname}.
compute yintkey = {' ',' ',' ',' ',' ',' '}.

```

```

do if (model < 4 and errs = 0).
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' ', ' '}.
compute datayed = {datayed, x&*m}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(1,i) = 1.
compute vmat(2,i) = modvals(i,1).
end loop.
compute mmat = make(16,nmeds,1).
end if.
do if (model = 2 or model = 3).
compute int1 = x&*w.
compute datayed = {datayed, w, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,2).
compute vmat(3,i) = modvals(i,1).
compute vmat(4,i) = modvals(i,1)*modvals(i,2).
end loop.
end if.
do if (model = 3).
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*m.
compute int2 = x&*w&*m.
compute datayed = {datayed, int1, int2}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' X', wname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.

```

```

compute intcnt = intcnt+1.
end if.
do if (model = 4 or model = 5).
compute vmat = make(8,1,1).
end if.
compute yintkey2 = yintkey.
do if (wm = 1).
compute int1 = x&*w.
compute datamed = {datamed, w,int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmm = {datanmm; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,wcol).
end loop.
do if (zm = 1).
compute int1 = x&*z.
compute datamed = {datamed, z, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(3,i) = modvals(i,zcol).
end loop.
end if.
do if (wzm = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.

```



```

compute int1 = w&*z.
compute int2 = x&*w&*z.
compute datamed = {datamed, int1, int2}.
loop i = 1 to nrow(modvals).
compute vmat(4,i) =
modvals(i,wcol)*modvals(i,zcol).
end loop.
end if.
end if.
compute mdatacol=ncol(datamed).
compute mintkey = yintkey.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
compute medints = intcnt-1.
do if (vy = 1 or xmy = 1).
compute mp = 1.
do if (xmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,1).
end loop.
end if.
do if (vy = 1).
compute datayed = {datayed, v}.
compute datanmy = {datanmy; vname}.
compute mmods = 1.
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,vcol).
end loop.
do if (qy = 1).
compute mp = 2.
compute datayed = {datayed,q}.
compute datanmy = {datanmy; qname}.
compute mmods = 2.
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,qcol).
end loop.
end if.
do if (vqy = 1).
compute mp = 3.
compute datayed = {datayed, v&*q}.
compute mmods = 3.

```

```

loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,vcol)*modvals(i,qcol).
end loop.
end if.
end if.
compute mints = make(n,(nmeds*mp),0).
loop i = 0 to (nmeds-1).
do if (i = 0 and vqy = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
vname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (xmy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))*x.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', xname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (qy = 1).
compute mints(:,((i*mp)+2))=m(:,(i+1))*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqy = 1).
compute mints(:,((i*mp)+3))=m(:,(i+1))*v&*q.

```

```

compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end loop.
compute datayed = {datayed, mints}.
end if.
compute mp = 1.
do if (wvmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,wcol)*modvals(i,vcol).
end loop.
end if.
compute mints2 = make(n,(nmeds*mp),0).
do if (wmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,wcol).
end loop.
do if (wy = 0 and model > 3).
compute datayed = {datayed, w}.
compute datanmy = {datanmy; wname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wvmy = 1).
compute datayed = {datayed,w&*v}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints2(:,(i*mp)+1))= m(:,(i+1))&*w.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' ', ' '}.

```

```

compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wvmy = 1).
compute mints2(:,(i*mp)+2))=
m(:,(i+1))&*w&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints2}.
end if.
compute mp = 1.
do if (zmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,zcol).
end loop.
do if (wzmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,zcol)&*modvals(i,wcol).
end loop.
end if.
end if.
do if (zmy = 1).
compute mints3 = make(n,(nmeds*mp),0).
do if (zy = 0).
compute datayed = {datayed, z}.
compute datanmy = {datanmy; zname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wzmy = 1 and wzy = 0).
compute datayed = {datayed,w&*z}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.

```

```

compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints3(:,((i*mp)+1))= m(:,(i+1))*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wzmy = 1).
compute mints3(:,((i*mp)+2))=
m(:,(i+1))*w*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', zname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints3}.
end if.
compute decoc = 1.
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
do if (wy = 1 and model > 3).
compute datayed = {datayed, w, x*w}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm2(1,1)=wname.
compute modmatp(1,1) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.

```

```

end if.
do if (zy = 1).
compute datayed = {datayed,z,x&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm2(1,2)=zname.
compute modmatp(1,2) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (wzy = 1).
compute datayed = {datayed,w&*z,x&*w&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vxy = 1).
compute datayed = {datayed, x&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm2(1,3)=vname.
compute modmatp(1,3) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, ' ', ' '}.

```

```

compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (qxy = 1).
compute datayed = {datayed, x&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm2(1,4)=qname.
compute modmatp(1,4) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', qname, ',', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqxy = 1).
compute datayed = {datayed, x&*v&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, 'X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end if.
do if (wvxy = 1).
compute datayed = {datayed, x&*w&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, 'X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute modvalsd = 0.
compute ttt = rsum(modmatp).

```

```

do if (rsum(modmatp) > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm2(1,tmp) = modvnm2(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:ttt).
compute modvnm2=modvnm2(:,1:ttt).
compute modmatv=modmatv(:,1:ttt).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvalsd =
make((modmatv(1,1)*modprod(1,1)), ttt,0).
loop i = 1 to ttt.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvalsd)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvalsd(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
end if.
do if (ttt > 0).
loop i = 1 to ncol(modvalsd).
do if (modvnm2(1,i)=wname).
compute wcol = i.

```



```

end if.
do if (modvnm2(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm2(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm2(1,i)=qname).
compute qcol = i.
end if.
end loop.
compute directv = make(nrow(modvalsd),1,1).
do if (wy = 1).
compute directv ={directv, modvalsd(:,wcol)}.
end if.
do if (zy = 1).
compute directv ={directv, modvalsd(:,zcol)}.
end if.
do if (wzy = 1).
compute directv ={directv,
(modvalsd(:,wcol)&*modvalsd(:,zcol))}.
end if.
do if (vxy = 1).
compute directv ={directv, modvalsd(:,vcol)}.
end if.
do if (qxy = 1).
compute directv ={directv, modvalsd(:,qcol)}.
end if.
do if (vqxy = 1).
compute directv ={directv,
(modvalsd(:,vcol)&*modvalsd(:,qcol))}.
end if.
do if (wvxy = 1).
compute directv ={directv,
(modvalsd(:,vcol)&*modvalsd(:,wcol))}.
end if.
end if.
compute ydatacol=ncol(datayed).
do if (ncovs > 0).
do if (covmy <> 2).

```

```

compute datamed = {datamed,c}.
end if.
do if (covmy <> 1).
compute datayed = {datayed,c}.
end if.
do if (covmy = 1 and model = 4 and mcx > 0).
compute datayed = {datayed,c}.
end if.
compute covmeans = csum(c)/n.
end if.
do if (cluster > 0).
compute datamed = {datamed,cld}.
compute datayed = {datayed, cld}.
compute cldmeans = csum(cld)/n.
end if.
compute mst = 3.
compute mnd = mst+nmeds-1.
compute ydatacol=ncol(datayed).
compute mdatacol=ncol(datamed).
do if (ncovs > 0).
compute datanmy = {datanmy; t(cnames)}.
do if (model > 3).
compute datanmm = {datanmm; t(cnames)}.
end if.
end if.
compute datanmy = {'constant';
datanmy(3:nrow(datanmy),1)}.
do if (model > 3).
compute datanmm = {'constant';
datanmm(3:nrow(datanmm),1)}.
end if.
compute amm = make(2,1,0).
compute abmm = make(2,1,0).
compute mnv = csum(datayed(:,2)/n).
compute mnv = make(n,1,mnv).
compute ssty = csum((datayed(:,2)-mnv)**2).
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
do if (ws=1).

```

```

compute stddevm=diag(sigma).
compute
stddevm=sqrt(stddevm(3:(nmeds+2),1)).
end if.
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute r2xy =
(sigma(2,(3+nmeds))/(stddevy*stddevx))&**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))&**2.
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).
do if (model = 4 and nmeds = 1 and cluster = 0
and ncovs = 0).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.

```

```

compute datatm = datamed.
compute dataty = datayed.
compute mdlInms2 = { '1' ; yname; xname}.
compute mdlInms = {'Model ='; ' Y ='; ' X ='
}.
do if (ws = 1).
compute mdlInms2 = { '1' ; yname}.
compute mdlInms = {'Model ='; ' Y ='}.
end if.
loop i = 1 to ncol(mnames).
compute mdlInms2 = {mdlInms2; mnames(1,i)}.
do if (i = 1 and ncol(mnames) = 1).
compute mdlInms = {mdlInms; ' M ='}.
else.
compute mdlInms = {mdlInms; mlab(i,1)}.
end if.
end loop.
do if (wname <> 'xxx').
compute mdlInms2 = {mdlInms2; wname}.
compute mdlInms = {mdlInms; ' W ='}.
end if.
do if (zname <> 'xxx').
compute mdlInms2 = {mdlInms2; zname}.
compute mdlInms = {mdlInms; ' Z ='}.
end if.
do if (vname <> 'xxx').
compute mdlInms2 = {mdlInms2; vname}.
compute mdlInms = {mdlInms; ' V ='}.
end if.
do if (qname <> 'xxx').
compute mdlInms2 = {mdlInms2; qname}.
compute mdlInms = {mdlInms; ' Q ='}.
end if.
do if (jn = 1 and model = 1 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
do if (jn = 1 and model = 3 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.

```

```

end if.
compute mdpbe=0.
compute medte2=0.
do if ( 1 =4 and mcx > 0).
compute medte2=1.
end if.
do if (((mcx > 0) or (mcm > 0)) AND model =
1).
compute mdpbe=1.
compute dd={datayed(:,4),datayed(:,3)}.
do if (ncovs > 0).
compute dd={datayed(:,6:(5+ncovs)),dd}.
end if.
compute dd={datayed(:,2),dd}.
end if.
compute yes = ((nrow(modvals)=1) and
(contrast = 1) and (model = 3 or model = 2)).
compute yes2 = ((nrow(modvals)=1) and
(contrast = 1) and (model = 2)).
print/title = '***** PROCESS Procedure
for SPSS Release 2.16.3 *****'.
print/title = '      Written by Andrew F. Hayes,
Ph.D.    www.afhayes.com'.
print/title = '  Documentation available in
Hayes (2013).
www.guilford.com/p/hayes3'/space=0.
print mdlnms2/title =
*****
*****'/rnames = mdlnms/format =
a8.
do if (ncovs > 0).
print cnames/title = 'Statistical Controls:'/rlabels
= 'CONTROL='/format a8.
end if.
print n/title = 'Sample size'/format F10.0.
do if (cluster > 0).
print cluster/rnames = cvname/title =
'Clustering Variable and Number of Clusters'.
end if.
do if (model > 3 and ( 'random' <> 'random')).

```

```

compute seedt= 'random'.
print seedt/title='Custom'/format=A12/rlabel =
'Seed:'.
end if.
do if (mdpbe=0 and medte2 = 0).
loop bt = 1 to (boot+1).
do if (bt = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef = 0.
do if (bt > 1).
loop.
compute v=trunc(uniform(n,1)*n)+1.
compute datayed = dataty(v,:).
do if (ws=1).
compute
bmeans=csum(datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed)))/n.
compute cntmat=make(n,nmeds,1).
compute bmeans2=bmeans.
do if (nmeds > 1).
compute bmeans2=mddiag(bmeans).
end if.
compute cntmat=cntmat*bmeans2.
compute datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed))=datayed(:,(ncol(data
yed)-nmeds+1):ncol(datayed))-cntmat.
end if.
compute detcheck=det(t(datayed)*datayed).
compute rk=(detcheck <> 0).
do if (model > 3).
compute datamed = datatm(v,:).
compute detcheck=det(t(datamed)*datamed).
do if (rk=1).
compute rk=(detcheck <> 0).
end if.
end if.
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).

```

```

compute sigma = sigma/(n*(n-1)).
compute temp=diag(sigma).
compute bad = bad+(1-rk).
compute false = 1.
end loop if (rk = 1).
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).
do if (model = 4 and nmeds = 1 and ncovs = 0
and cluster = 0).
compute r2xy =
(sigma(2,4)/(stddevy*stddevx))&**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))&**2.
compute sstot = sigma(2,2)*(n-1).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).

```

```

compute abmm(2,1)=amma*(kappac/kappae).
end if.
end if.
do if (model > 3).
loop im = 1 to nmeds.
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant';
datanmm((2+nmeds):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
do if (model = 6).
do if (im = 1).
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
end if.
do if (im > 1).
compute xm={cons, datamed(:,3:(im+1)),
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant'; datanmm(2:im,1);
datanmm((mnd):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
compute
mmpaths((im+1),(2:im))=t(coeff(2:im,1)).
end if.
end if.
do if (ws = 1).
compute
coeff(1,1)=csum(datamed(:,(2+im)))/n.
end if.
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
do if (bt = 1).

```



```

compute resid=datamed(:,(2+im))-xm*coeff.
compute sse =cssq(resid).
compute mse = sse/(n-ncol(xm)).
compute mnv = csum(data(:,(2+im))/n).
compute mnv = make(n,1,mnv).
compute sstm = csum((data(:,(2+im))-
mnv)**2).
compute k3 = nrow(coeff).
do if (hc3 = 1).
compute h = xm(:,1).
loop i3=1 to n.
compute h(i3,1)= xm(i3,:)*invXtX*t(xm(i3,:)).
end loop.
loop i3=1 to k3.
compute xm(:,i3) = (resid(:,ncol(resid))&/(1-
h))*xm(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xm(:,i3) = sqrt(mse)*xm(:,i3).
end loop.
end if.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute hccov=invXtX*t(xm)*xm*invXtX.
compute mcmats(im,im)=hccov(2,2).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*hccov*lmat)*((t(lmat)
)*coeff))/dfnum.
compute coeff = coeff(1:(nrow(coeff)-
clsdmy),1).
compute mcoeff(im,1)=coeff(2,1).
compute standerr =
sqrt(diag(invXtX*t(xm)*xm*invXtX)).
compute standerr = standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1).

```

```

compute stderr(1,1)=stddevm(im,1)/sqrt(n).
end if.
compute tratio = coeff&/stderr.
compute p = 2*(1-tcdf(abs(tratio), (n-
ncol(xm))))).
compute temp=(n-ncol(xm)).
do if (ws=1).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute temp=(n-1).
end if.
compute xd = abs(xp2).
compute temp = (temp* (exp((temp-
(5/6))*((xd/(temp-(2/3)+(.11/temp))))*(xd/(temp-
(2/3)+(.11/temp)))))-1)).
compute temp1 = coeff-
sqrt(abs(temp))*stderr.
compute temp2 =
coeff+sqrt(abs(temp))*stderr.
compute op = {coeff, stderr, tratio, p, temp1,
temp2}.
do if (ws=1).
compute op=op(1,:).
compute xnm =
apathnam(((nmeds>1)+1+(im-1)),1).
end if.
compute sobel(im,1) = coeff(2,1).
compute sobel(im,2)=stderr(2,1).
compute temp = mnames(1,im).
compute r2full = 1-(sse/sstm).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute summ = {sqrt(r2full), r2full, mse,
Fratio, dfnum, dfden, pfr}.
do if (detail = 1).
print temp/title =
*****
*****'/rlabels = 'Outcome:'/format
= A8.
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels 'R',
'R-sq', 'MSE','F', 'df1', 'df2', 'p'/format = F10.4.

```

```

end if.
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
print op/title = 'Model'/rnames = xnmn/clabels
'coeff', 'se', 't', 'p', 'LLCI', 'ULCI'/format = F10.4.
do if (covcoeff=1 and ws <> 1).
compute
hccovtmp=hccov(1:nrow(op),1:nrow(op)).
compute cnamestp=t(xnmn).
print hccovtmp/title='Covariance matrix of
regression parameter
estimates'/rnames=xnmn/cnames=cnamestp/f
ormat= F10.4.
end if.
do if (nmods > 0 and nrow(mintkey) > 1).
print mintkey/title = 'Product terms key: '/format
= A8.
end if.
do if (ws = 1).
compute wsdf=n-1.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
end if.
end if.
compute ymat(1,im) = coeff(2,1).
do if (ws = 1).
compute ymat(1,im)=coeff(1,1).
end if.
do if (wm = 1).
compute ymat(2,im) = coeff(4,1).
do if (zm = 1).
compute ymat(3,im) = coeff(6,1).
do if (wzm = 1).
compute ymat(4, im) = coeff(8,1).
end if.
end if.
end if.
do if (model = 6).
compute mmpaths((im+1),1)=coeff((im+1),1).

```

```

end if.
end loop.
end if.
loop totlp = 1 to (1+(toteff*(bt = 1))).
do if (toteff = 1 and totlp =2).
compute xy =
{cons,datayed(:,(3+nmeds):ydatacol)}.
else.
compute xy={cons, datayed(:,3:ydatacol)}.
end if.
do if (dichy = 1).
compute pt2 =
make(nrow(datayed(:,2)),1,(csum(datayed(:,2))
/n)).
compute LL3 = datayed(:,2)*ln(pt2)+(1-
datayed(:,2))*ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xy),1,0).
compute LL1 = 0.
compute xy22=xy.
loop jjj = 1 to iterate.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.
compute coeff =
bt1+inv(t(xy22)*xy)*t(xy)*(datayed(:,2)-pt1).
compute pt1 = 1/(1+exp(-(xy*coeff))).
compute itprob = csum((pt1 <
.00000000000001) or (pt1 >
.99999999999999)).
do if (itprob = 0).
compute LL = datayed(:,2)*ln(pt1)+(1-
datayed(:,2))*ln(1-pt1).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
break.
end if.
compute bt1 = coeff.

```

```

compute LL1 = LL2.
end loop.
do if (jjj >= iterate and iterr = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
computer iterr = 1.
end if.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.
compute covmat = inv(t(xy22)*xy).
release xy22.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
end if.
do if (dichy = 0).
compute invXtX = inv(t(xy)*xy).
compute coeff =invXtX*t(xy)*datayed(:,2).
do if (effsize=1 and model > 3 and model < 7
and (ncovs > 0 or clsdmy > 0)).
compute xysdy =
{cons,datayed(:,(4+nmeds):ydatacol)}.
compute coeffsd
=inv(t(xysdy)*xysdy)*t(xysdy)*datayed(:,2).
compute resid=datayed(:,2)-(xysdy*coeffsd).
compute sdycov =sqrt(csum(resid*resid)/(n-
ncol(xysdy))).
compute xvaron=datayed(:,(3+nmeds)).
compute
coeffx2=inv(t(xysdy)*xysdy)*t(xysdy)*xvaron.
compute residx2=xvaron-(xysdy*coeffx2).
compute sdxcov
=sqrt(csum(residx2*residx2)/(n-ncol(xysdy))).
end if.
do if (totlp <> 2).

```

```

compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4 and bt > 1).
compute resid=datayed(:,2)-xy*coeff.
compute sse = cssq(resid).
compute r2full = 1-(sse/sstot).
end if.
do if (bt =1).
compute resid=data(:,2)-xy*coeff.
compute k3 = nrow(coeff).
compute sse = cssq(resid).
compute mse = sse/(n-ncol(xy)).
compute
hcnor=coeff&/(sqrt(diag(mse*invXtX))).
do if (hc3 = 1).
compute h = xy(:,1).
loop i3=1 to n.
compute h(i3,1)= xy(i3,:)*invXtX*t(xy(i3,:)).
end loop.
loop i3=1 to k3.
compute xy(:,i3) = (resid(:,ncol(resid))&/(1-
h))*xy(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xy(:,i3) = sqrt(mse)*xy(:,i3).
end loop.
end if.
compute covmat = (invXtX*t(xy)*xy*invXtX).
end if.
end if.
do if (bt = 1).
do if (model = 2).
compute xy2={cons, datayed(:,3:ydatacol)}.
compute temp = ncol(xy2).
do if (temp > 6).

```

```

compute xy3=xy2(:,7:temp)}.
end if.
compute xy2={xy2(:,1:3), xy2(:,5)}.
do if (temp > 6).
compute xy2={xy2, xy3}.
release xy3.
end if.
compute invXtX = inv(t(xy2)*xy2).
compute coeff2 = invXtX*t(xy2)*datayed(:,2).
compute ssem2=cssq(datayed(:,2)-xy2*coeff2).
release xy2.
end if.
compute standerr = sqrt(diag(covmat)).
do if (totlp =1).
compute
mcmats((nmeds+1):ncol(mcmats),(nmeds+1):n
col(mcmats))=covmat(2:(1+nmeds),2:(1+nmed
s)).
end if.
compute standerr=standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1 and totp=2).
compute standerr(1,1)=stddevy/sqrt(n).
end if.
compute coeffplt = coeff.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*coeff)))/dfnum).
compute coeff=coeff(1:(nrow(coeff)-clsdmy),1).
do if (totlp=1).
compute
mccoeff((nmeds+1):nrow(mccoeff))=coeff(2:(1+
nmeds),1).
end if.
compute bbbb=coeff(2,1).
do if totp = 1.

```

```

compute deco(1,1)=2+nmeds.
compute deco = deco(1:decoc,1).
compute covdirt = make((nrow(covmat)-
clsdmy),(ncol(covmat)-clsdmy),0).
compute covdirt = covmat(deco,:).
compute covdir =
make(nrow(covdirt),ncol(covdirt),0).
compute covdir = covdirt(:,t(deco)).
compute deco=coeff(deco,1).
do if (ttt > 0).
compute sedir =
sqrt(diag(directv*covdir*t(directv))).
compute directv = directv*deco.
end if.
compute sobel(:,3)=coeff(2:(1+nmeds),1).
compute sobel(:,4) = standerr(2:(1+nmeds),1).
compute sobel2 = sobel*sobel.
do if (varorder <> 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)*sobel2(:,4)+sobel2(:,3)*sob
el2(:,2)).
end if.
do if (varorder = 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)*sobel2(:,4)+sobel2(:,3)*sob
el2(:,2)+sobel2(:,2)*sobel2(:,4)).
end if.
compute sobel(:,1) = sobel(:,1)*sobel(:,3).
compute sobel(:,3) = sobel(:,1)/sobel(:,2).
compute sobel(:,4) = 2*(1-
cdfnorm(abs(sobel(:,3)))).
end if.
do if (dichy = 0).
compute tratio = coeff/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
do if (ws= 1 and totlp=2).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute dfden=n-1.
end if.

```



```

compute cnms = {'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute op = {coeff, stderr, tratio, p}.
end if.
do if (dichy = 1).
compute tratio = (coeff&/stderr).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute wald = tratio&*tratio.
compute cnms = {'coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute temp=coeff-abs(xp2)*stderr.
compute op = {coeff, stderr, tratio, p, temp}.
compute temp=coeff+abs(xp2)*stderr.
compute op = {op, temp}.
end if.
do if (detail = 1).
do if (totlp = 2).
print yname/title = '*****'
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:.'/format = A8.
end if.
do if (totlp <> 2).
print yname/title =
'*****'
*****'/rlabels = 'Outcome:.'/format
= A8.
end if.
end if.
do if (dichy = 1 and bt = 1 and totp = 1).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:.'/cnames = nmsd/format = F9.2.
end if.
do if (dichy = 0).
compute r2full = 1-(sse/ssty).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute jndf=dfden.
compute wsdf=dfden.
compute xd = abs(xp2).

```

```

compute jncrit = (dfden* (exp((dfden-
(5/6))*((xd/(dfden-
(2/3)+(.11/dfden)))*(xd/(dfden-
(2/3)+(.11/dfden)))))-1)).
compute summ = {sqrt(r2full), r2full, mse,
fratio, dfnum, dfden, pfr}.
compute temp1=coeff-sqrt(jncrit)*standerr.
compute temp2=coeff+sqrt(jncrit)*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (detail = 1).
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels =
'R', 'R-sq', 'MSE' 'F', 'df1', 'df2', 'p'/format =
F10.4.
end if.
end if.
end if.
do if (dichy = 1).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(coeff)-
1)).
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
do if (detail = 1).
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format =
F10.4.
end if.
end if.
do if (totlp = 2).
compute datanmy={'constant';
datanmy((nmeds+2):nrow(datanmy),1)}.
end if.
do if (detail = 1).
do if (coeffci = 0).

```

```

compute op = op(:,1:(ncol(op)-2)).
end if.
do if (ws=1).
compute op2=op(1:(1+nmeds),:).
compute datanmy2 = {'c" path'; bpathnam(
(1+(nmeds>1)): ((1+(nmeds>1)))+(nmeds-
1)),1)}.
do if (totlp=2).
compute datanmy2={'c path'}.
compute op2=op(1,:).
compute wsdf=n-1.
end if.
print op2/title = 'Model'/rnames =
datanmy2/cnames = cnms/format = F10.4.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
do if (ws <> 1).
print op/title = 'Model'/rnames =
datanmy/cnames = cnms/format = F10.4.
do if (covcoeff=1).
compute
covmattp=covmat(1:nrow(op),1:nrow(op)).
compute cnamestp=t(datanmy).
print covmattp/title='Covariance matrix of
regression parameter
estimates'/rnames=datanmy/cnames=cnamest
p/format= F10.4.
end if.
end if.
end if.
do if (ttt = 0 and totp = 1).
compute deco = op((nmeds+2),:).
do if (ws = 1).
compute deco=op(1,:).
end if.
end if.
do if (ttt = 0 and totp = 2).
compute decotot = op(2,:).
do if (ws = 1).
compute decotot=op(1,:).

```

```

end if.
end if.
do if (nmods > 0 and model > 4 and detail = 1
and nrow(yintkey) > 1)).
print yintkey/title = 'Product terms key: '/format
= A8.
end if.
do if (nmods > 0 and model < 4 and detail = 1).
print yintkey2/title = 'Product terms key: '/format
= A8.
do if ((model = 1 or model = 2) and dichy = 0).
compute temp={((op(4,3)**2)*(1-r2full))/dfden,
op(4,3)**2, 1, dfden, op(4,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(4,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(2,1).
do if (model = 2).
compute temp={temp;((op(6,3)**2)*(1-
r2full))/dfden, op(6,3)**2, 1, dfden, op(6,4)}.
do if (hc3=1).
compute temp(2,1)=((hcnot(6,1)**2)*(1-
r2full))/dfden.
end if.
compute frat2=(dfden*(r2full-(1-
(ssem2/ssty))))/(2*(1-r2full)).
do if (hc3=1).
compute lmat=make(nrow(coeffplt),2,0).
compute lmat(4,1)=1.
compute lmat(6,2)=1.
compute frat2 =
(t(t(lmat)*coeffplt)*inv(t(lmat)*covmat*lmat)*((t(l
mat)*coeffplt))/2).
end if.
compute temp={temp;(r2full-(1-
(ssem2/ssty))),frat2,2,dfden,1-
fcd(frat2,2,dfden)}.
compute rnms={rnms; yintkey2(3,1);'Both'}.
end if.

```

```

print temp/title = 'R-square increase due to
interaction(s):'/rnames=rnms/clabels = 'R2-
chng', 'F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (model = 3 and dichy = 0).
compute temp={((op(8,3)**2)*(1-r2full))/dfden,
op(8,3)**2, dfden, op(8,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(8,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(5,1).
print temp/title = 'R-square increase due to
three-way interaction:.'/rnames=rnms/clabels =
'R2-chng', 'F(1,df2)', 'df2', 'p'/format = F10.4.
end if.
end if.
end if.
do if (model = 6 and totlp = 1).
compute
mmpaths(nrow(mmpaths),1)=coeff(nrow(mmpa
ths),1).
compute
mmpaths(nrow(mmpaths),(2:(nmeds+1)))=t(co
eff(2:(nmeds+1),1)).
end if.
do if (totlp = 1).
loop im = 1 to nmeds.
do if (model < 4).
compute ymat(1,im) = coeff(3,1)*(1-yes).
compute ymat(2,im) = coeff(4,1).
compute cmat(1,im) = covmat(3,3).
compute cmat(2,im) = covmat(4,4).
compute cmat(5,im) = covmat(3,4).
compute jnb1=coeff(3,1).
compute jnb3=coeff(4,1).
compute jnsb1=covmat(3,3).
compute jnsb3=covmat(4,4).
compute jnsb1b3=covmat(3,4).
do if (model = 2 or model = 3).

```

```

compute ymat(3,im) = coeff(6,1).
compute cmat(3,im) = covmat(6,6).
compute cmat(6,im) = covmat(3,6).
compute cmat(8,im) = covmat(4,6).
end if.
do if (model = 3).
compute ymat(4,im) = coeff(8,1).
compute cmat(4,im) = covmat(8,8).
compute cmat(7,im) = covmat(3,8).
compute cmat(9,im) = covmat(4,8).
compute cmat(10,im) = covmat(6,8).
compute jnb1=coeff(4,1).
compute jnb3=coeff(8,1).
compute jnsb1=covmat(4,4).
compute jnsb3=covmat(8,8).
compute jnsb1b3=covmat(4,8).
end if.
end if.
do if (model > 3).
compute ymat(5,im) = coeff((1+im),1).
end if.
do if (xmy = 1).
compute ymat(6,im) = coeff((2+nmeds+im),1).
end if.
do if (vy = 1).
compute ymat(6,im) = coeff((3+nmeds+im),1).
end if.
do if (qy = 1 and vy = 1).
compute ymat(6,im) = coeff((5+nmeds+((im-1)*2)),1).
compute ymat(7,im) = coeff((6+nmeds+((im-1)*2)),1).
end if.
do if (vqy = 1).
compute ymat(6,im) = coeff((6+nmeds+((im-1)*3)),1).
compute ymat(7,im) = coeff((7+nmeds+((im-1)*3)),1).
compute ymat(8,im) = coeff((8+nmeds+((im-1)*3)),1).

```

```

end if.
do if (wmy = 1).
compute ymat(7,im) = coeff((3+nmeds+im-
wy),1).
end if.
do if (wmy = 1 and vy = 1).
compute ymat(7,im) =coeff((4+(nmeds*2)+im-
wy),1).
end if.
do if (wmy = 1 and vy = 1 and wvmy = 1).
compute ymat(7,im) =
coeff((6+(nmeds*2)+((im-1)*2)-wy),1).
compute ymat(8,im) =
coeff((7+(nmeds*2)+((im-1)*2)-wy),1).
end if.
do if (wmy = 1 and zmy = 1).
compute ymat(6,im) = coeff((6-(wzy*3)+(wzm-
1)+(nmeds*2)+((im-1)*2)*wzm)+((im-1)*(1-
wzm))-((zy-wzy)*2),1).
compute ymat(7,im) = coeff((3-
zy+im+nmeds),1).
do if (wzmy = 1).
compute ymat(8,im) = coeff((7-
(wzy*3)+(nmeds*2)+((im-1)*2)),1).
end if.
end if.
do if (nmods > 0 and model <> 5).
loop indlp = 1 to nrow(modvals).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp)).
do if (model > 6).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp))*csum(ym
at(5:8,im)*vmat(5:8,indlp)).
end if.
end loop.
compute indboot((bt+(im-
1)*(boot+1)),:)=t(indeff).
do if (imm =1 or imm = 3).

```

```

compute
indbootp(bt,im)=ymat(2,im)*ymat(5,im).
do if (imm=3).
compute
indbootp(bt,(im+nmeds))=ymat(3,im)*ymat(5,i
m).
do if (zdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxz-cminz).
end if.
end if.
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
end if.
end if.
do if ((imm= 2 or imm = 4) or model = 74).
compute
indbootp(bt,im)=ymat(1,im)*ymat(6,im).
do if (imm=4).
compute
indbootp(bt,(im+nmeds))=ymat(1,im)*ymat(7,i
m).
do if (qdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxq-cminq).
end if.
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
end if.
end if.
do if (imm=7).
compute
indbootp(bt,im)=ymat(2,im)*ymat(6,im).

```



```

compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(2,im)*y
mat(6,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(6,im)*onesmv1)+(ymat(2,im)*y
mat(6,im)*t(immmv1)).
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2))))*(c
maxw-cminw).
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1))))*(c
maxv-cminv).
end if.
end if.
do if (imm=5).
compute
indbootp(bt,im)=ymat(4,im)*ymat(5,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(4,im)*y
mat(5,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(3,im)*ymat(5,im)*onesmv1)+(ymat(4,im)*y
mat(5,im)*t(immmv1)).

```

```

do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).
end if.
do if (zdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxz-
cminz).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxz-cminz).
end if.
end if.
do if (imm=6).
compute
indbootp(bt,im)=ymat(1,im)*ymat(8,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(1,im)*ymat(6,im)*onesmv2)+(ymat(1,im)*y
mat(8,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(7,im)*onesmv1)+(ymat(1,im)*y
mat(8,im)*t(immmv1)).
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-

```

```

1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxv-cminv).
end if.
do if (qdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxq-
cminq).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxq-cminq).
end if.
end if.
do if ((model = 58 or model = 59) and
wvdich=1).
compute
indbootp(bt,im)=(ymat(1,im)*ymat(7,im)*(cmax
w-cminw))+(ymat(2,im)*ymat(5,im)*(cmaxw-
cminw))+(ymat(2,im)*ymat(7,im)*((cmaxw*cma
xw)-(cminw*cminw))).
end if.
end if.
do if (model = 4 or model = 5).
compute
indboot(bt,im)=csum(ymat(1:4,im)&*vmat(1:4,1
))*csum(ymat(5:8,im)&*vmat(5:8,1)).
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.000000000000001.
end if.
compute sumind=rsum(indboot(bt,:)).
do if (im=nmeds).
compute
pmeff(bt,2:(im+1))=indboot(bt,:)/(sumind+coeff(
(2+nmeds),1)).
end if.
compute
rmeff(bt,(im+1))=indboot(bt,im)/coeff((2+nmeds
),1).

```

```

compute
abpseff(bt,(im+1))=indboot(bt,im)/stddevy.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*stddevx.
do if (ncovs > 0 or clsdmy > 0).
compute
abpseff(bt,(im+1))=indboot(bt,im)/sdy cov.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*sdx cov.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,1) = r2my-(r2full-r2xy).
compute abmmr = 1.
do if (indboot(bt,im) < 0).
compute abmmr = cmin(abmm).
end if.
do if (indboot(bt,im) > 0).
compute abmmr = cmax(abmm).
end if.
compute kappa2(bt,1)=indboot(bt,im)/abmmr.
compute tmp = indboot(bt,im)/abmmr.
end if.
end if.
end if.
do if (model = 6).
do if (nmeds = 2).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(4,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mm
paths(4,3).
compute
indboot(bt,3)=mmpaths(3,1)*mmpaths(4,3).
end if.
do if (nmeds = 3).
compute indboot(bt,1) =
mmpaths(2,1)*mmpaths(5,2).
compute indboot(bt,2) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3).

```

```

compute indboot(bt,3) =
mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4).
compute indboot(bt,4) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*m
mpaths(5,4).
compute indboot(bt,5) =
mmpaths(3,1)*mmpaths(5,3).
compute indboot(bt,6) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4).
compute indboot(bt,7) =
mmpaths(4,1)*mmpaths(5,4).
end if.
do if (nmeds = 4).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(6,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mm
paths(6,3).
compute
indboot(bt,3)=mmpaths(2,1)*mmpaths(4,2)*mm
paths(6,4).
compute
indboot(bt,4)=mmpaths(2,1)*mmpaths(5,2)*mm
paths(6,5).
compute
indboot(bt,5)=mmpaths(2,1)*mmpaths(3,2)*mm
paths(4,3)*mmpaths(6,4).
compute
indboot(bt,6)=mmpaths(2,1)*mmpaths(3,2)*mm
paths(5,3)*mmpaths(6,5).
compute
indboot(bt,7)=mmpaths(2,1)*mmpaths(4,2)*mm
paths(5,4)*mmpaths(6,5).
compute
indboot(bt,8)=mmpaths(2,1)*mmpaths(3,2)*mm
paths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,9)=mmpaths(3,1)*mmpaths(6,3).

```

```

compute
indboot(bt,10)=mmpaths(3,1)*mmpaths(4,3)*m
mpaths(6,4).
compute indboot(bt,11)
=mmpaths(3,1)*mmpaths(5,3)*mmpaths(6,5).
compute indboot(bt,12) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4)*m
mpaths(6,5).
compute indboot(bt,13) =
mmpaths(4,1)*mmpaths(6,4).
compute indboot(bt,14) =
mmpaths(4,1)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,15) =
mmpaths(5,1)*mmpaths(6,5).
end if.
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.00000000000001.
end if.
compute
pmeff(bt,:)=indboot(bt,:)/(rsum(indboot(bt,:))+m
mpaths(nrow(mmpaths),1)).
compute rmeff(bt,:) =
indboot(bt,:)/mmpaths(nrow(mmpaths),1).
compute abpseff(bt,:)=indboot(bt,:)/stddevy.
compute
abcseff(bt,:)=(stddevx*indboot(bt,:))/stddevy.
do if (ncovs > 0 or clsdmy > 0).
compute abpseff(bt,:)=indboot(bt,:)/sdy cov.
compute
abcseff(bt,:)=(sdx cov*indboot(bt,:))/sdy cov.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,:) = r2my-(r2full-r2xy).
end if.
end if.
end if.
end loop.
end if.

```

```

end loop.
do if (savboot = 1 and bt > 1).
compute bootstrp((bt-1),:)=bootcoef.
end if.
end loop.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
release datayed, dat, datamed, data, y, x,
datatm, dataty, mints2, cons, mnv, tmp.
do if (mc > 0).
compute x1 = sqrt(-
2*ln(uniform(mc,nrow(mcmats))))&*cos((2*3.14
159265358979)*uniform(mc,nrow(mcmats))).
compute x1=x1*chol(mcmats).
loop i=1 to nrow(x1).
compute x1(i,:)=x1(i,:)+t(mccoeff).
end loop.
loop i = 1 to nmeds.
compute
indboot(2:nrow(indboot),i)=x1(:,i)&*x1(:,(i+nme
ds)).
end loop.
end if.
do if (ttt = 0 and model > 3).
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
end if.
do if (model <> 74).
do if (dichy = 0).
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 't', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.

```

```

print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 't', 'p', 'LLCI', 'ULCI'/format =
F10.4.
else.
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 'Z', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 'Z', 'p', 'LLCI', 'ULCI' /format =
F10.4.
end if.
end if.
end if.
do if (ttt > 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
compute clbs = {modvnm2, 'Effect', 'SE', 't', 'p',
'LLCI', 'ULCI'}.
compute tratio = (directv&/sedir).
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy))))).
compute outp = {modvalsd, directv, sedir,
tratio, p}.
do if (dichy = 0).
compute temp1=directv-sqrt(jncrit)*sedir.
compute temp2=directv+sqrt(jncrit)*sedir.
compute outp = {outp, temp1, temp2}.
end if.
do if (dichy = 1).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute temp = directv-abs(xp2)*sedir.
compute outp = {outp, temp}.
compute temp = directv+abs(xp2)*sedir.
compute outp = {outp, temp}.
compute clbs = {modvnm2, 'Effect', 'SE', 'Z', 'p',
'LLCI', 'ULCI'}.
end if.
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).

```



```

end if.
print outp/title = 'Conditional direct effect(s) of
X on Y at values of the moderator(s):'/format =
F10.4 /cnames = clbs.
end if.
do if (nmods > 0 and model <> 5).
do if (model < 4).
print/title =
*****
*****!
compute zmat(1,1) = 1*(1-yes).
compute cfse = make(nrow(modvals),1,0).
loop #m = 1 to nrow(modvals).
do if (model = 1).
compute zmat(2,1)=modvals(#m,1)**2.
compute zmat(5,1)=2*modvals(#m,1).
end if.
do if (model = 2 or model = 3).
compute zmat(2,1)=modvals(#m,2)**2.
compute zmat(3,1)=modvals(#m,1)**2.
compute
zmat(4,1)=(modvals(#m,1)**2)*(modvals(#m,2)
**2)*(1-yes2).
compute zmat(5,1)=2*modvals(#m,2)*(1-yes).
compute zmat(6,1)=2*modvals(#m,1)*(1-yes).
compute
zmat(7,1)=2*modvals(#m,1)*modvals(#m,2)*(1-
yes).
compute
zmat(8,1)=2*modvals(#m,1)*modvals(#m,2).
compute
zmat(9,1)=2*modvals(#m,1)*(modvals(#m,2)**
2)*(1-yes2).
compute
zmat(10,1)=2*(modvals(#m,1)**2)*modvals(#m
,2)*(1-yes2).
end if.
compute cfse(#m,1)=sqrt(csum(zmat&*cmat)).
end loop.
end if.

```

```

do if (nmods > 0).
compute clbs = {modvnm, 'Effect'}.
loop im = 1 to nmeds.
compute obs = t(indboot(1+(im-1)*(boot+1),:)).
compute outp = {modvals, obs}.
do if (model < 4).
compute tstat = obs&/cfse.
do if (dichy = 0).
compute pval = 2*(1-tcdf(abs(tstat), (n-
ncol(xy))))).
compute temp=obs-sqrt(jncrit)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp=obs+sqrt(jncrit)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 't', 'p', 'LLCI', 'ULCI'}.
compute jnclbs=clbs.
end if.
do if (dichy = 1).
compute pval = 2*(1-cdfnorm(abs(tstat))).
compute temp = obs-abs(xp2)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp = obs+abs(xp2)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute jnclbs=clbs.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indboot((1+(im-
1)*(boot+1),:)).
compute indboot2 = indboot((2+(im-
1)*(boot+1)):(1+(im-1)*(boot+1)+boot),:).
compute mnind = t(csum(indboot2)/boot).
compute stdind=t(sqrt((cssq(indboot2)-
((csum(indboot2)**2)/boot))/(boot-1))).
compute llci=make(1,ncol(indboot2),-999).
compute ulci=make(1,ncol(indboot2),-999).
loop #e = 1 to ncol(indboot2).

```

```

.
compute temp = indboot2(:,#e).
compute temp(GRADE( indboot2(:,#e) )) =
indboot2(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.

```

```

do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {modvals, obs, stdind, t(llci),
t(ulci)}.
compute clbs = {modvnm, 'Effect', 'Boot SE',
'BootLLCI', 'BootULCI'}.
end if.
compute mtemp = mnames(1, im).
compute rlbs =
make(nrow(modvals),1,mnames(1,im)).
do if (model < 4).
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
do if (yes=0).
print outp/title = 'Conditional effect of X on Y at
values of the moderator(s):'/cnames =
clbs/format = F10.4.
else.
compute outp=outp(:,3:ncol(outp)).
compute clbs=clbs(:,3:ncol(clbs)).
compute clbs(1,1)='Contrast'.
print outp/title = 'Contrast of conditional effects
of X on Y'/cnames = clbs/format = F10.4.
end if.

```

```

end if.
do if (model > 5 and mod74dic <> 1).
do if (im = 1).
print/title = 'Conditional indirect effect(s) of X on
Y at values of the moderator(s):'.
end if.
print outp/title = 'Mediator'/rnames =
rlbs/cnames = clbs/format = F10.4.
else if (mod74dic = 1 and model = 74).
do if (im = 1).
print/title = 'Indirect effect(s) of X on Y:'.
compute clbs3=clbs(1,2:ncol(clbs)).
end if.
compute outp3=outp(1,2:ncol(outp)).
print outp3/title = 'Mediator'/rnames =
rlbs/cnames = clbs3/format = F10.4.
end if.
end loop.
loop i = notes to 1 by -1.
do if (note(i,1) = 4 and yes=0).
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 5 and yes=0).
print/title = 'Values for quantitative moderators
are the mean and plus/minus one SD from
mean.'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 14 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    below the mean was replaced
with the minimum because one SD below the
mean'/space=0.

```

```

print/title = '    is outside of the range of the
data.'/space=0.
end if.
do if (note(i,1) = 15 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    above the mean was replaced
with the maximum because one SD above the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.
end loop.
do if (model = 3 and yes=0).
compute jnvals=make(nrow(matw),7,0).
compute jnvals(:,1)=matw.
compute jnvals(:,2)=jnb1+jnb3*jnvals(:,1).
compute
jnvals(:,3)=sqrt(jnsb1+2*jnvals(:,1)*jnsb1b3+(jn
vals(:,1)*jnvals(:,1))*jnsb3).
compute jnvals(:,4)=jnvals(:,2)/jnvals(:,3).
do if (dichy = 0).
compute jnvals(:,5)=2*(1-tcdf(abs(jnvals(:,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(:,5)=2*(1-
cdfnorm(abs(jnvals(:,4)))).
end if.
compute jnvals(:,6)=jnvals(:,2)-
sqrt(jncrit)*jnvals(:,3).
compute
jnvals(:,7)=jnvals(:,2)+sqrt(jncrit)*jnvals(:,3).
compute clbs={clbs(:,1),clbs(:,3:ncol(clbs))}.
do if (coeffci = 0).
compute jnvals =jnvals(:,1:(ncol(jnvals)-2)).
end if.
print jnvals/title = 'Conditional effect of X*M
interaction at values of W:'/cnames
=clbs/format = F10.4.

```

```

end if.
do if (jn = 1 and (model = 1 or model = 3) and
jndich = 0 and yes=0).
print/title = '***** JOHNSON-
NEYMAN TECHNIQUE
*****'.
compute ajn =(jncrit*jnsb3)-(jnb3*jnb3).
compute bjn = 2*((jncrit*jnsb1b3)-(jnb1*jnb3)).
compute cjn = (jncrit*jnsb1)-(jnb1*jnb1).
compute radarg = (bjn*bjn)-(4*ajn*cjn).
compute den = 2*ajn.
compute nrts = 0.
do if (radarg >= 0 and den <> 0).
compute x21 = (-bjn+sqrt(radarg))/den.
compute x22 = (-bjn-sqrt(radarg))/den.
compute roots = 0.
do if (x21 >= jnmin and x21 <= jnmax).
compute nrts = 1.
compute roots = {roots; x21}.
end if.
do if (x22 >= jnmin and x22 <= jnmax).
compute nrts = nrts + 1.
compute roots = {roots; x22}.
end if.
compute roots={roots,make(nrow(roots),2,0)}.
compute modtemp=m.
do if (model=3).
compute modtemp=w.
end if.
do if (nrts > 0).
compute roots = roots(2:nrow(roots),1:3).
compute roots(1,2)=(csum(modtemp <
roots(1,1))/n)*100.
compute roots(1,3)=(csum(modtemp >
roots(1,1))/n)*100.
do if (nrow(roots)=2).
compute roots(2,2)=(csum(modtemp <
roots(2,1))/n)*100.
compute roots(2,3)=(csum(modtemp >
roots(2,1))/n)*100.

```

```

end if.
print roots/title = 'Moderator value(s) defining
Johnson-Neyman significance
region(s)'/clabels = 'Value', '% below', '%
above'/format = F10.4.
compute jnvals=make((21+nrts),7,0).
loop i= 0 to 20.
compute jnvals((i+1),1)=jnmin+(i*((jnmax-
jnmin)/20)).
end loop.
loop i = 1 to nrts.
loop j = 2 to nrow(jnvals).
do if ((roots(i,1) > jnvals((j-1),1)) and (roots(i,1)
< jnvals(j,1))).
compute
jnvals((j+1):(21+i),1)=jnvals(j:(20+i),1).
compute jnvals(j,1)=roots(i,1).
end if.
end loop.
end loop.
loop i = 1 to nrow(jnvals).
compute jnvals(i,2)=jnb1+jnb3*jnvals(i,1).
compute
jnvals(i,3)=sqrt(jnsb1+2*jnvals(i,1)*jnsb1b3+(jn
vals(i,1)*jnvals(i,1))*jnsb3).
compute jnvals(i,4)=jnvals(i,2)/jnvals(i,3).
do if (dichy = 0).
compute jnvals(i,5)=2*(1-tcdf(abs(jnvals(i,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(i,5)=2*(1-
cdfnorm(abs(jnvals(i,4)))).
end if.
compute jnvals(i,6)=jnvals(i,2)-
sqrt(jncrit)*jnvals(i,3).
compute
jnvals(i,7)=jnvals(i,2)+sqrt(jncrit)*jnvals(i,3).
end loop.
do if (model = 1).

```



```

print jnvals/title = 'Conditional effect of X on Y
at values of the moderator (M)'/cnames
=jnclbs/format = F10.4.
end if.
do if (model = 3).
compute
jnclbs={jnclbs(:,1),jnclbs(:,3:ncol(jnclbs))}.
print jnvals/title = 'Conditional effect of X*M on
Y at values of the moderator (W)'/cnames
=jnclbs/format = F10.4.
end if.
end if.
do if (nrts = 0).
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
else.
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
end if.
do if (model < 4 and (plot = 1)).
compute dataplot =
make((nrow(modvals)*nrow(matx)),(ncol(modv
als)+1),0).
compute tmp = 1.
loop i = 1 to nrow(modvals).
loop j = 1 to nrow(matx).
compute dataplot(tmp,:)= {matx(j,1),
modvals(i,:)}.
compute tmp=tmp+1.
end loop.
end loop.
compute dataplot = {dataplot,
make(nrow(dataplot),(1+dichy),0)}.
compute dataplo2 = make(nrow(dataplot),1,1).
do if (model = 1).

```

```

compute dataplo2 = {dataplo2, dataplot(:,2),
dataplot(:,1), (dataplot(:,1)*dataplot(:,2))}.
end if.
do if (model = 2 or model = 3).
compute dataplo2 = {dataplo2, dataplot(:,3),
dataplot(:,1), (dataplot(:,1)*dataplot(:,3)),
dataplot(:,2), (dataplot(:,1)*dataplot(:,2))}.
do if (model = 3).
compute dataplo2 = {dataplo2,
(dataplot(:,2)*dataplot(:,3)),
(dataplot(:,1)*dataplot(:,2)*dataplot(:,3))}.
end if.
end if.
loop i = 1 to nrow(dataplot).
compute tmp=dataplo2(i,:).
do if (ncovs > 0).
compute tmp = {tmp, covmeans}.
end if.
do if (cluster > 0).
compute tmp = {tmp, cldmeans}.
end if.
compute dataplot(i,(ncol(dataplot)-
(dichy)))=tmp*coeffplt.
do if (dichy = 1).
compute
dataplot(i,(ncol(dataplot)))=exp(tmp*coeffplt)/(1
+exp(tmp*coeffplt)).
end if.
end loop.
compute clbs = {xname, modvnm, 'yhat'}.
do if (dichy = 1).
compute clbs = {xname, modvnm, 'ln(odds)',
'prob'}.
end if.
print/title =
*****
*****!

```

```

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (model = 1).
do if (dichy = 0).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Technological_Advan
Operational_Performance.' /space=1.
else if (dichy = 1).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Technological_Advan
logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH Operational_Performance BY
Technological_Advan.' /space=0.
end if.

```

```

do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH Operational_Performance BY
Plan_Orga_Pract.' /space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH logodds BY Technological_Advan.'
/space=0.
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH prob BY Technological_Advan.'
/space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Plan_Orga_Pract.'
/space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Plan_Orga_Pract.' /space=0.
end if.
end if.
end if.
do if (model = 2 or model = 3).
do if (dichy = 0).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Technological_Advan
Operational_Performance.' /space=1.
end if.
do if (dichy = 1).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Technological_Advan
logodds prob.' /space=1.
end if.

```

```

print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH Operational_Performance BY
Technological_Advan/PANEL ROWVAR=.'
/space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH Operational_Performance BY
Plan_Orga_Pract/PANEL ROWVAR=.'
/space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH logodds BY
Technological_Advan/PANEL ROWVAR=.'
/space=0.
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH prob BY Technological_Advan/PANEL
ROWVAR=.' /space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Plan_Orga_Pract/PANEL
ROWVAR=.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Plan_Orga_Pract/PANEL
ROWVAR=.' /space=0.

```

```

end if.
end if.
end if.
do if (ncovs > 0).
print/title = '* Estimates are based on setting
covariates to their sample means.'.
end if.
end if.
end if.
do if ((imm > 0) or (model = 74 and
mod74dic=0) or ((model = 58 or model = 59)
and wvdich=1))).
do if (imm > 4 and imm < 8).
compute
indbootp={indbootp(:,1:nmeds),indboop1,indbo
op2}.
end if.
compute obsprod = t(indbootp(1,:)).
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indbootp(1,:).
compute indbootp = indbootp(2:(boot+1),:).
compute mnindp = t(csum(indbootp)/boot).
compute stdindp=t(sqrt((cssq(indbootp)-
((csum(indbootp)**2)/boot))/(boot-1))).
compute llcip=make(1,ncol(indbootp),-999).
compute ulcip=make(1,ncol(indbootp),-999).
loop #e = 1 to ncol(indbootp).
.
compute temp = indbootp(:,#e).
compute temp(GRADE( indbootp(:,#e) )) =
indbootp(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.

```

```

compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcip(1,#e)=llcit.
compute ulcip(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).

```

```

compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {obsprod, stdindp, t(llcip),
t(ulcip)}.
compute clbs = {'Effect', 'SE(Boot)', 'BootLLCI',
'BootULCI'}.
do if (model = 8 or model = 12).
print/title='-----'.
compute outpimm=outp(1:nmeds,:).
print/title = 'Indirect effect of highest order
product:.'/space=0.
print outpimm/title = 'Mediator'/cnames =
clbs/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
compute clbs3 = {'Index', 'SE(Boot)',
'BootLLCI', 'BootULCI'}.
do if (imm < 5).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=outp((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.

```



```

end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'.
compute outpimm=outp(1:nmeds,:).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
compute
outpimm=outp((nmeds+1):nrow(outp),:).
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),clbs3}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3(1,1)=modvnm(1,1).
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.

```

```

print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
end if.
do if (boot = 0).
do if (model = 8 or model = 12).
print/title='-----'.
compute obsimm=obsprod(1:nmeds,:).
print/title = 'Indirect effect of highest order
product.'/space=0.
print obsimm/title = 'Mediator'/clabels =
'Effect'/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
do if ((imm > 0 and imm < 5) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
loop k4= 1 to nmods.
do if (nmods > 1).

```

```

print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=obsprod((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
end loop.
end if.
do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'.
compute outpimm=obsprod(1:nmeds,1).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
compute
outpimm=obsprod((nmeds+1):nrow(obsprod),:)
.
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),'Index'}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm((((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3={modvnm(1,1),'Index'}.
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).

```

```

end if.
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
end if.
end if.
end if.
compute conmake=0.
compute concols = 0.
do if ((model > 3 and model < 7) and (contrast
> 0) and nmods = 0 and nmeds > 1 ).
compute concols =
((ncol(indboot)*(ncol(indboot)-1))/2).
compute indcon=make(nrow(indboot),concols,-
999).
compute conkey = { ' ' ' ' ' ' }.
compute temp=1.
compute conmake=1.
loop i = 1 to (ncol(indboot)-1).
loop j = (i+1) to (ncol(indboot)).
compute indcon(:,temp)=indboot(:,i)-
indboot(:,j).
do if (contrast = 2).
compute indcon(:,temp)=abs(indboot(:,i))-
abs(indboot(:,j)).
end if.
do if (model <= 6).
compute conkey={conkey; mnames(1,i),
'minus', mnames(1,j)}.
end if.
do if (model = 6).
compute conkey={conkey; indlbl2(i,1), 'minus',
indlbl2(j,1)}.
end if.
compute temp=temp+1.
end loop.
end loop.
end loop.

```

```

end if.
do if (model = 4 or model = 5).
compute clbs = {'Effect'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obs = {csum(obs); obs}.
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute rlbs = {rlbs;
cntname(1:ncol(indcon),1)}.
end if.
compute outp = obs.
compute outp2=outp.
do if (effsize = 1 and dichy = 0).
compute
pmeff(:,1)=rsum(pmeff(:,2:ncol(pmeff))).
compute
rmeff(:,1)=rsum(rmeff(:,2:ncol(rmeff))).
compute
abpseff(:,1)=rsum(abpseff(:,2:ncol(abpseff))).
compute
abcseff(:,1)=rsum(abcseff(:,2:ncol(abcseff))).
compute eff = {pmeff, rmeff, abpseff, abcseff}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute eff = {eff, r245}.
compute r245obs = {r245(1,1);r245(1,1)}.
compute kappa2ob = kappa2(1,1).
end if.
compute pmobs = t(pmeff(1,1:(nmeds+1))).
compute rmobs = t(rmeff(1,1:(nmeds+1))).
compute psobs = t(abpseff(1,1:(nmeds+1))).
compute csobs = t(abcseff(1,1:(nmeds+1))).
do if (contrast = 0).
compute outp2 = {obs, psobs, csobs, pmobs,
rmobs}.
end if.
do if (contrast > 0).
compute obs2=obs(1:nrow(psobs),:).

```

```

compute outp2 = {obs2, psobs, csobs, pmobs,
rmobs}.
end if.
compute clbs = {'ab', 'ab_ps', 'ab_cs', 'ab/c',
'ab/c'}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute outp2 = {outp2, r245obs}.
compute clbs = {clbs, 'R-sq_med'}.
do if (boot = 0 and mc = 0).
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
do if (boot = 0 and mc = 0).
do if (nmeds = 1).
compute outp2 = outp2(2,:).
compute rlbs = rlbs(2,1).
end if.
do if (model=5).
compute outp2=outp2(:,1:3).
end if.
print outp2/title = 'Indirect effect(s) of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1 and nmeds >
1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0 or mc > 0).
compute indboot = {rsum(indboot), indboot}.
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.

```

```

end if.
compute ones = make(bootsz,1,1).
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(bootsz+1),:).
compute mnind = t(csum(indboot)/bootsz).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/bootsz))/(bootsz-1))).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).

.
compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).

```

```

compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
do if (effsize = 1 and dichy = 0 and boot > 0).
compute estmte=eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).
.
compute temp = eff(:,#e).

```



```

compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if

```

```

.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
end if.
end if.
do if (boot > 0 or mc > 0).
compute outp = {obs, stdind, t(llci), t(ulci)}.
do if (nmeds = 1).
compute outp = outp(2,.).
compute rlbs = rlbs(2,1).
end if.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
do if (mc > 0).
compute clbs = {'Effect', 'MC SE', 'MC LLCI',
'MC ULCI'}.
end if.
print outp/title = 'Indirect effect of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (dichy = 0 and effsize = 1 and boot > 0).
compute outp = {psobs,
stdindf((2*(nmeds+1)+1):(3*(nmeds+1)),1),
t(llcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))),
t(ulcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,.).
end if.
print outp/title = 'Partially standardized indirect
effect of X on Y'/rnames = rlbs/cnames =
clbs/format = F10.4.

```

```

compute outp = {csobs,
stdindf((3*(nmeds+1)+1):(4*(nmeds+1)),1),
t(llcif(1,(3*(nmeds+1)+1):(4*(nmeds+1)))),
t(ulcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))}.
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Completely standardized
indirect effect of X on Y'/rnames = rlbs/cnames
= clbs/format = F10.4.
compute outp = {pmobs,
stdindf(1:(nmeds+1),1),t(llcif(1,1:(nmeds+1))),
t(ulcif(1,1:(nmeds+1))) }.
do if (model = 4).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
compute outp = {rmobs, stdindf(((nmeds+1)+1)
:(2*(nmeds+1)) ,1),
t(llcif(1,((nmeds+1)+1):(2*(nmeds+1)))),
t(ulcif(1,((nmeds+1)+1):(2*(nmeds+1))))}.
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
do if (nmeds = 1 and cluster = 0 and ncovs = 0
and model = 4).
compute r245obs = r245obs(1,1).
compute outp = {r245obs,
stdindf((4*(nmeds+1)+1):(4*(nmeds+1)+1),1),
t(llcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1))),
t(ulcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1))))}.
print outp/title = 'R-squared mediation effect
size (R-sq_med)'/rnames = rlbs/cnames =
clbs/format = F10.4.

```

```

compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
end if.
do if (normal = 1).
do if (nmeds = 1).
print sobel/title = 'Normal theory tests for
indirect effect'/clabels = 'Effect', 'se', 'Z',
'p'/format = F10.4.
end if.
do if (nmeds > 1).
compute rlbs2 = rlbs(2:nrow(rlbs),1).
print sobel/title = 'Normal theory tests for
specific indirect effects'/rnames = rlbs2/clabels
= 'Effect', 'se', 'Z', 'p'/format = F10.4.
end if.
end if.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
do if (model = 6).
compute clbs = {'eff'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obsfsum=csum(obs).
compute obs = {obsfsum; obs}.
compute indlbl = indlbl(1:nrow(obs),1).
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute indlbl = {indlbl;
cntname(1:ncol(indcon),1)}.
end if.
compute obs2=obs.

```

```

do if (boot = 0).
do if (dichy = 0 and effsize = 1).
compute clbs = {'eff', 'eff_ps', 'eff_cs', 'eff/c',
'eff/c'}.
do if (ncovs = 0).
compute obs = {obs, obs/stddevy,
(obs*stddevx/stddevy), obs/ctot,
obs/mmpaths(nrow(mmpaths),1)}.
end if.
do if (ncovs > 0 or clsdmy > 0).
compute obs = {obs, obs/sdyconv,
(obs*sdxconv/sdyconv),
obs/(obsfsum+mmpaths(nrow(mmpaths),1)),
obs/mmpaths(nrow(mmpaths),1)}.
end if.
compute obs2=obs.
do if (contrast > 0).
compute obs2 = obs(1:(nrow(obs)-concols),:).
end if.
end if.
print obs2/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute indboot = {rsum(indboot), indboot}.
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(boot+1),:).
compute mnind = t(csum(indboot)/boot).

```

```

compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/boot))/(boot-1))).
compute temp = nrow(indboot).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).

.

compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.

```

```

compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute obs = {obs, stdind, t(llci), t(ulci)}.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
print obs/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (effsize = 1 and dichy = 0).
compute indboot=indboot(:,1:(ncol(indboot)-
concols)).
compute eff = {rsum(abpseff), abpseff,
rsum(abcseff), abcseff, rsum(pmeff), pmeff,
rsum(rmeff), rmeff}.
compute effobs = eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).

```

```

loop #e = 1 to ncol(eff).

.

compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(effobs(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.

```



```

do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute temp2 = stdindf(1:ncol(indboot),1).
compute temp3 = effobs(:,1:ncol(indboot)).
compute templow = llcif(1,1:ncol(indboot)).
compute temphi = ulcif(1,1:ncol(indboot)).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Partially standardized indirect
effect of X on Y'/cnames = clbs/rnames =
indlbl/format = F10.4.
compute temp2 =
stdindf((ncol(indboot)+1):(2*ncol(indboot)),1).
compute temp3 =
effobs(:,(ncol(indboot)+1):(2*ncol(indboot))).
compute templow =
llcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute temphi =
ulcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Completely standardized
indirect effect of X on Y'/cnames = clbs/rnames
= indlbl/format = F10.4.

```

```

compute temp2 =
stdindf((2*(ncol(indboot))+1):(3*ncol(indboot)),1
).
compute temp3 =
effobs(:,(2*(ncol(indboot))+1):(3*ncol(indboot)))
.
compute templow =
llcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute temphi =
ulcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
compute temp =
eff(:,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temp2 =
stdindf((3*(ncol(indboot))+1):(4*ncol(indboot)),1
).
compute temp3 =
effobs(:,(3*(ncol(indboot))+1):(4*ncol(indboot)))
.
compute templow =
llcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temphi =
ulcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
end if.
end if.
do if (nmeds = 2).
compute effkey = {xname, '->', mnames(1,1), '-
->', yname, ',', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname}.

```

```

compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, '', ''}.
compute effkey = {indl(2:4,1), effkey}.
end if.
do if (nmeds = 3).
compute effkey = {xname, '->', mnames(1,1), '->', yname, '', '', ''}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname, '
', ''}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,3), '->', yname, '
', ''}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, '', '', ''}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', mnames(1,3), '->', yname, '
', ''}.
compute effkey = {effkey; xname, '->',
mnames(1,3), '->', yname, '', '', ''}.
compute effkey = {indl(2:8,1), effkey}.
end if.
do if (nmeds = 4).
compute effkey = {xname, '->', mnames(1,1), '->', yname, '', '', '', ''}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname, '
', '', ''}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,3), '->', yname, '
', '', ''}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,4), '->', yname, '
', '', ''}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname, '', ''}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,4), '->', yname, ' ', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, ' ', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', mnames(1,4), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, ' ', ' ', ' ', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,4), '->', yname, '
', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, ' ', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, ' ', ' ', ' ', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', mnames(1,4), '->', yname, '
', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,4), '->', yname, ' ', ' ', ' ', ' ', ' ', ' '}.
compute effkey = {indlbl(2:16,1), effkey}.
end if.
print effkey/title = 'Indirect effect key'/format =
A8.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
end if.

```

```

DO IF (medte2=1).
compute
name=t(mdlms2(2:nrow(mdlms2),1)).
compute tmpdat=datayed(:,2:ncol(datayed)).
compute dd=tmpdat(:,1).
compute
dd={dd,tmpdat(:,(2+nmeds)),tmpdat(:,2:(1+nmeds))}.
compute stratify=0.
do if (ncovs > 0).
compute cname = cnames.
compute
dd={dd,tmpdat(:,(3+nmeds):(2+nmeds+ncovs))}.
end if.
compute method = 1.
compute samples = boot.
do if (mc > 0).
compute samples = mc.
compute method=2.
end if.
compute omnibus = 1.
compute booterr = 0.
do if (mcx < 0 or mcx > 4).
compute mcx = 1.
end if.
compute ovals = ncol(design(dd(:,1))).
compute nte = notes.
compute criterr = 0.
compute xskip = 0.
compute dumok = 0.
compute badend = 0.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute priorlo = -9999999.
compute priorhi = 9999999.

```

```

compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute reps = samples.
do if (contrast = 1).
compute note(nte,1) = 26.
compute nte=nte+1.
end if.
do if (effsize = 1).
compute note(nte,1) = 27.
compute nte=nte+1.
end if.
do if (normal = 1).
compute note(nte,1) = 28.
compute nte=nte+1.
end if.
do if (method = 2).
do if (omnibus = 1).
compute note(nte,1) = 25.
compute nte=nte+1.
end if.
end if.
compute samples = 0.
do if (reps < 1000).
compute reps = 1000.
end if.
end if.
do if (method = 1).
compute samples = reps.
end if.
compute temp = ncol(dd).
compute n = nrow(dd).
compute ones = make(n,1,1).
compute temp=dd(:,2).
compute nxd = ncol(temp).
compute nx = nxd.
do if (nx = 1 and mcx > 0).
compute temp = dd.
compute temp(GRADE(dd(:,2)),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,2)).
compute nvls = ncol(dummy).

```

```

compute nnvls = csum(dummy).
compute mnvls = cmin(t(nnvls)).
do if (mnvls < 2).
compute errs = errs + 1.
compute runerrs(errs,1) = 37.
compute criterr = 1.
end if.
do if (nvls > 9).
compute errs = errs+1.
compute runerrs(errs,1) = 36.
compute criterr = 1.
end if.
do if (criterr = 0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,2).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,2) <> nnvls((temp-1),1)).
compute nnvls(temp,1)=dd(i,2).
compute temp = temp+1.
end if.
end loop.
do if (mcx > 0).
compute x = dummy(:,2:ncol(dummy)).
compute nx = ncol(x).
compute minus1 = make(1,ncol(x),-1).
compute xdes=make((nx+1),3,0).
compute xdes(1,1)=dd(1,2).
compute xdes(1,2)=1.
compute temp = 2.
loop k = 2 to n.
do if (dd(k,2) <> dd((k-1),2)).
compute xdes(temp,2) = k.
compute xdes(temp,1) = dd(k,2).
compute xdes((temp-1),3) = k-1.
compute temp=temp+1.
end if.
end loop.
compute xdes((temp-1),3)=n.

```

```

compute xdes = {xdes, (xdes(:,3)-xdes(:,2)+1)};
do if (stratify <> 0).
compute note(nte,1) = 7.
compute nte=nte+1.
end if.
do if (mcx = 4).
loop k = 1 to n.
do if (rsum(x(k,:)) = 0).
compute x(k,:) = minus1.
end if.
end loop.
end if.
do if (mcx = 2 or mcx = 3).
compute note(nte,1)=4.
compute nte = nte+1.
loop k = 1 to n.
do if (rsum(x(k,:)) > 0).
loop i = 1 to ncol(x).
do if (x(k,i) = 0).
compute x(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcx = 3).
compute conmat1={-8,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1; 0,0,0,0,0,-
3,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.
loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute x(k,:)=conmat1((rsum(x(k,:))+1),:).
end loop.
end if.

```



```

end if.
compute xdata = x.
compute xskip = 1.
compute xname = ddd(1,1:nx).
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcx = 4).
compute dummat(1,:) = minus1.
end if.
do if (mcx = 2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcx = 3).
compute dummat=conmat1.
end if.
compute dummat={nnvls, dummat}.
compute xname2={name(1,2), xname}.
end if.
end if.
end if.
compute nm=nmeds.
compute r2b = make(nm,1,-999).
compute indeff = make((nx*nm),1,-999).
compute cov =
make((nx*nm+nm),(nx*nm+nm),0).
compute mns=make(nrow(cov),1,0).
compute ny = ncol(dd)-nm-nxd-ncovs.
compute y = dd(:,1).
compute ydata = y.
compute yname = name(1,1).
do if (xskip = 0).
compute x = dd(:,2:(1+nxd)).
compute xdata = x.
compute xname = name(1,2:(1+nxd)).

```

```

compute indlbs = t(xname).
end if.
compute m = dd(:,(2+nx):(1+nx+nm)).
compute mdata = m.
compute mlab = { ' M1 = ';' ' M2 = ';' ' M3 = ';' '
M4 = ';' ' M5 = ';' ' M6 = ';' ' M7 = ';' ' M8 = ';' '
M9 = ';' ' M10 = '}.
compute mlab = {mlab; ' M11 = ';' ' M12 = ';' '
M13 = ';' ' M14 = ';' ' M15 = '}.
compute mname =
name(1,(2+nx):(1+nx+nm)).
do if (ncovs > 0).
compute c =
dd(:,(2+nx+nm):(1+nx+nm+ncovs)).
compute x = {x, c}.
compute xdata = x.
compute cdata = c.
compute xname = {xname, cname}.
end if.
compute nms={yname; t(mname);
t(xname(1,1:(ncol(xname)-ncovs)))}.
do if (dumok = 1).
compute nms={yname; t(mname); name(1,2)}.
end if.
do if (nm < 16).
do if (dumok = 1).
print dummat/title = 'Coding of categorical X
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
compute
boots=make((samples+1),((nx+1)*nm),-999).
do if (errs = 0 and criterr = 0).
loop k = 1 to (samples+1).
do if (k = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef=0.

```

```

do if (k > 1).
compute u=trunc(uniform(n,1)*n)+1.
compute xdata = x(u,:).
compute mdata = m(u,:).
compute ydata = y(u,:).
do if (ncovs > 0).
compute cdata = c(u,:).
end if.
end if.
loop i = 1 to nm.
compute start = 1+(i-1)*nx.
compute xmat = {ones, xdata}.
do if (covmy=2).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = mdata(:,i).
compute amat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*amat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute r2cha = r2.
compute adjr2cha = adjr2.
do if (ncovs > 0 and covmy <> 2).
compute xmat = {ones, cdata}.
compute atmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*atmat)**2).
compute r2step1 = (sstotal-ssresid2)/sstotal.
compute r2cha = r2-r2step1.
compute adjr2cha = adjr2- (1-((1-r2step1)*(n-
1)/(n-ncol(xmat)))).
compute xmat = {ones, xdata}.
end if.
compute bootcoef={bootcoef,t(amat)}.

```

```

compute r2b(i,1)=adjr2cha.
do if (k = 1).

.
compute hcinvtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvtX*t( xmat )* xmat
*hcinvtX
.
compute tratio = amat&/sqrt(diag(covmat)).
compute rnms = mname(1,i).
print rnms/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.
compute lmat = ident(nrow(amat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*amat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*amat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).

```

```

compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/format = F10.4
/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'.
compute op = {amat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp))))*(xp2/(temp-
(2/3)+(.11/temp)))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute cov(start:(start+(nx-
1)),start:(start+(nx-
1)))=covmat(2:(ncol(covmat)-(ncovs*(covmy <>
2))),2:(ncol(covmat)-(ncovs*(covmy <> 2)))).
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns(start:(start+(nx-
1)),1)=amat(2:(1+nx)).
end loop.
compute xmat = {ones,mdata,xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (k = 1).

```

```

print yname/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.
do if (ovals = 2).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:'/'cnames = nmsd/'format = F9.2.
end if.
end if.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(yamat)/n)).
compute LL3 = yamat*&ln(pt2)+(1-yamat*)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= ymat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.

```

```

compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute bootcoef={bootcoef,t(bmat)}.
compute LL1 = LL2.
compute covmat=varb.
end if.
do if (ovals <> 2).
compute bmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*bmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute bootcoef={bootcoef,t(bmat)}.
end if.
do if (k = 1).
do if (ovals = 2).

```

```

compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).
compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms2/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute
opdirsum=outp((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
end if.
do if (ovals <> 2).
.
compute hcinvXtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).

```



```

loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvtX*t( xmat )* xmat
*hcinvtX
.
compute lmat = ident(nrow(bmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/(ncol(xmat)-1).
compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat((2+nmeds):(2+nmeds+(nx-1)),:)
= lmat2.
compute fratiodr =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/nx.
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute df2=(n-ncol(xmat)).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), df2, pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = bmat&/sqrt(diag(covmat)).

```

```

compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute op = {bmat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat)))) }.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*(xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms2/format =
F10.4.
compute
opdirsum=op((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
compute xmat = {ones,mdata}.
do if (ncovs > 0).
compute xmat = {xmat, cdata}.
end if.
compute btmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*btmat)**2).
compute r2direct = r2-((sstotal-
ssresid2)/sstotal).
compute pfr = 1-fcdf(fratiodr,(nx),(n-1-nx-nm-
ncovs)).
compute r2direct = {r2direct, fratiodr, nx, (n-1-
nx-nm-ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter

```

```

estimates'/rnames=rnms2/cnames=rnms2/form
at= F10.4.
end if.
end if.
do if (k = 1 and toteff = 1).
print yname/title = '*****'
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:'/format = A8.
compute xmat = {ones, xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(ymat)/n)).
compute LL3 = ymat*&ln(pt2)+(1-ymat)*&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= ymat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.

```

```

do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&*ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*ulp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute LL1 = LL2.
compute covmat=varb.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(bmat)-
1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'

```

```

'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms = {'Constant'; t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).
compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute optotsum=outp(2:(1+nx),:).
end if.
do if (ovals <> 2).
compute cmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*cmat.
compute ssresid = csum((resid)&**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).

.

compute hcinvtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.

```

```

compute xmat(:,i3) = ( resid(:,ncol( resid
))&/(1-h))&* xmat(:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat(:,i3) = sqrt( mse )&* xmat(:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(cmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = cmata/sqrt(diag(covmat)).
compute op = {cmat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp))))*(xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.

```

```

print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute optotsum=op(2:(1+nx),:).
compute ssresid2 = ssresid.
compute r2total = r2.
compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat(2:(1+nx),:) = lmat2.
compute fratio3 =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/nx.
do if (ncovs > 0).
compute xmat = {ones, cdata}.
compute ctmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*ctmat)**2).
compute r2total = r2-((sstotal-ssresid2)/sstotal).
end if.
compute pfr = 1-fcdf(fratio3,(nx),(n-1-nx-
ncovs)).
compute r2tot2 = {r2total, fratio3, nx, (n-1-nx-
ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns((nrow(mns)-
nm+1):nrow(mns),1)=bmat(2:(1+nm),1).
compute tmp = 1.
loop i = 1 to (nm*nx).
compute
boots(k,i)=mns(i,1)*mns(((nm*nx)+tmp),1).
do if ((i/nx)=trunc(i/nx)).

```

```

compute
boots(k,((nm*nx)+tmp))=r2b(tmp,1)*mns(((nm*
nx)+tmp),1).
compute tmp = tmp+1.
end if.
end loop.
do if (savboot = 1 and k > 1).
compute bootstrp((k-1),:)=bootcoef.
end if.
end loop.
do if (method = 1).
compute llci=make(1,ncol(boots),-999).
compute ulci=make(1,ncol(boots),-999).
compute indeff = t(boots(1,1:(nm*nx))).
compute indeff22=t(boots(1,:)).
compute omni = t(boots(1,((nm*nx)+1) :
ncol(boots))).
do if (nrow(boots) > 1).
compute boots = boots(2:nrow(boots),:).
compute bootse = t(sqrt((((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
loop i = 1 to ncol(boots).
.
compute temp = boots(:,i).
compute temp(GRADE( boots(:,i) )) = boots(:,i).
compute badlo = 0.
compute badhi = 0.
do if ( (indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(indeff22(i,1)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).

```



```

compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,i)=llcit.
compute ulci(1,i)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute llci = t(llci).

```

```

compute ulci= t(ulci).
compute llci={llci,ulci}.
end if.
end if.
do if (method = 2).
compute tmp=1.
loop i = 1 to (nm*nx).
compute
indeff(i,1)=mns(i,1)*mns(((nm*nx)+tmp),1).
do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = sqrt(-
2*ln(uniform(reps,nrow(cov))))&*cos((2*3.1415
9265358979)*uniform(reps,nrow(cov))).
compute boots=boots*chol(cov).
loop i = 1 to ncol(boots).
compute boots(:,i)=boots(:,i)+mns(i,1).
end loop.
compute tmp=1.
loop i = 1 to (nm*nx).
compute
boots(:,i)=boots(:,i)*boots(:,((nm*nx)+tmp)).
compute temp = boots(:,i).
compute temp(GRADE(boots(:,i))) = boots(:,i).
compute boots(:,i) = temp.
do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = boots(:,1:(nm*nx)).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
compute llci =
t({(boots(cilow,:));(boots(cihigh,:))}).
end if.
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.

```

```

else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
compute rnmssum=rnms(2:(1+nx),1).
do if (ovals <> 2).
print optotsum/title = 'Relative total effects of X
of Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.
print r2tot2/title = 'Omnibus test of total effect of
X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print optotsum/title = 'Relative total effects of X
on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
end if.
compute
rnmssum=rnms2((2+nmeds):(1+nmeds+nx),1).
do if (ovals <> 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.
print r2direct/title = 'Omnibus test of direct
effect of X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
do if (omnibus = 1 and (method = 1 or method
= 3)).
compute indlbs = {indlbs; 'Omnibus'}.
end if.
loop i = 1 to nm.

```

```

compute op = mname(1,i).
print op/title = 'Relative indirect effect(s) of X on
Y through:.'/format = A8/space=0.
do if (method = 2).
compute clbs = {'Effect', 'SE MC', 'MC LLCI',
'MC ULCI'}.
compute op = {indeff, bootse(1:(nm*nx)), llci}.
end if.
do if (method = 1).
compute clbs = {'Effect', 'SE(boot)', 'LLCI',
'ULCI'}.
compute op = indeff.
do if (samples > 0).
compute op = {indeff, bootse(1:(nm*nx),:),
llci(1:(nm*nx),:)}).
end if.
end if.
compute temp = op((((i-1)*nx)+1):(nx*i),:).
do if ((method = 1) and omnibus = 1).
do if (samples = 0).
compute temp={temp;omni(i,1)}.
end if.
do if (samples > 0).
compute temp = {temp; omni(i,1),
bootse(((nm*nx)+i),:), llci(((nm*nx)+i),:)}).
end if.
end if.
print temp/title = '.'/rnames = indlbs/cnames =
clbs/format = F10.4 /space=0.
print/title = '-----'.
end loop.
end if.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.

```

```

do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
END IF.
DO IF (mdpbe=1).
compute nm={yname,xname,mnames}.
do if (ncovs > 0).
compute
nm={yname,cnames,xname,mnames}.
end if.
compute tpx=xname.
compute x = dd(:,2:ncol(dd)).
compute tpy = yname.
compute tempy= dd(:,1).
compute n = nrow(dd).
compute nx=1.
compute criterr=0.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute ddd1 = {'int_1', 'int_2', 'int_3', 'int_4',
'int_5', 'int_6', 'int_7', 'int_8', 'int_9'}.
compute dumok = 0.
compute mcfoc=mcx.
compute mcmmod=mcm.
compute mcloc=(mcfoc>0).
compute con = make(n,1,1).
compute ncovs=ncol(x)-2.
do if (mcfoc > 0 or mcmmod > 0).
compute temp = dd.
compute temp(GRADE(dd(:,(ncol(dd)-
mcloc))),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,ncol(dd)-
mcloc)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute toosmall=rsum(nnvls < 2).
compute mnvls = cmin(t(nnvls)).
do if ((rsum(nnvls < 2)) > 0).

```

```

compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=37.
end if.
do if (nvls > 10).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=36.
end if.
do if (criterr=0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,ncol(dd)-mcloc).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,ncol(dd)-mcloc) <> nnvls((temp-
1),1)).
compute nnvls(temp,1)=dd(i,ncol(dd)-mcloc).
compute temp = temp+1.
end if.
end loop.
compute dummy = dummy(:,2:ncol(dummy)).
do if (mcfoc=4 or mcmmod=4).
compute minus1=make(1,ncol(dummy),-1).
loop k = 1 to n.
do if (rsum(dummy(k,:)) = 0).
compute dummy(k,:)=minus1.
end if.
end loop.
end if.
do if (mcfoc = 2 or mcfoc = 3) or (mcmmod = 2 or
mcmmod = 3)).
loop k = 1 to n.
do if (rsum(dummy(k,:)) > 0).
loop i = 1 to ncol(dummy).
do if (dummy(k,i) = 0).
compute dummy(k,i) = 1.
else.
break.
end if.

```

```

end loop.
end if.
end loop.
do if (mcfoc = 3 or mcmmod=3).
compute conmat1={-8,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1; 0,0,0,0,0,-
3,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.
loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute
dummy(k,:)=conmat1((rsum(dummy(k,:))+1),:).
end loop.
end if.
end if.
compute nx = ncol(dummy).
compute xname = ddd(1,1:nx).
compute xdata = dummy.
compute xname = ddd(1,1:nx).
compute xname2={nm(1,(ncol(nm)-mclloc)),
xname}.
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcfoc = 2 or mcmmod=2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcfoc = 3 or mcmmod=3).
compute dummat=conmat1.
end if.
do if (mcfoc = 4 or mcmmod = 4).

```

```

compute dummat(1,:)=minus1.
end if.
compute dummat={nnvls, dummat}.
end if.
end if.
do if (criterr=0).
compute y = dd(:,1).
compute ovals = ncol(design(dd(:,1))).
compute itprob = 0.
do if (ovals = 2).
compute omx = cmax(y(:,1)).
compute omn = cmin(y(:,1)).
compute y(:,1) = (y(:,1) = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute sstotal = csum((y-(csum(y)/n))&**2).
compute outv = t(nm(1,1)).
compute mdtr = nm(1,ncol(dd)).
compute fciv = nm(1,(ncol(dd)-1)).
do if (mcfoc <> 0 or mcmod <> 0).
compute inter=make(n,nx,0).
loop i = 1 to nx.
compute inter(:,i)=dummy(:,i)&*dd(:,ncol(dd)-
(1-mcloc)).
end loop.
do if (ncovs = 0).
compute x={con,dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y)))).
end if.
do if (ncovs > 0).
compute
x={con,dd(:,2:(1+ncovs)),dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={t(nm(1,2:(1+ncovs)));
nm(1,ncol(dd)-(1-mcloc));

```



```

t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
end if.
compute xmns = csum(x)&/n.
compute focvals = {1,0,0}.
compute highwarn = 0.
compute lowwarn = 0.
do if (ncol(x) > (2*nx+2)).
compute covmns = {1, xmns(1,2:(ncol(x)-1-
(2*nx)))}.
compute focvals = {covmns,0,0}.
end if.
compute dfres = n-ncol(x).
do if (ovals = 2).
compute pt2 = make(n,1,(csum(y)/n)).
compute LL3 = y*&ln(pt2)+(1-y)&ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(x),1,0).
compute LL1 = 0.

.

compute pt1lp= pt1.
compute xlp= x.
compute ylp= y.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*ulp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(ulp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).

```

```

compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&*ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*ulp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
else if (ovals <> 2).
compute invXtX=inv(t(x)*x).
compute b = invXtX*t(x)*y.
compute resid = y-(x*b).
compute ssresid=csum(resid*&resid).
compute msresid=ssresid/dfres.
compute varb = msresid*invXtX.
compute
seint=sqrt(varb(nrow(varb),nrow(varb))).
compute k3 = nrow(b).
do if (hc3 = 1).
compute xhc=x.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)= xhc(i3,:)*invXtX*t(xhc(i3,:)).
end loop.

```

```

loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).
end loop.
compute varb=(invXtX*t(xhc)*xhc*invXtX).
end if.
compute seb = sqrt(diag(varb)).
compute r2 = 1-(ssresid/sstotal).
do if (mcfoc <> 0 or mcmmod <> 0).
compute ytp=y.
compute xtp=x(:,1:(ncol(x)-nx)).
compute btp = inv(t(xtp)*xtp)*t(xtp)*ytp.
compute residt = ytp-(xtp*btp).
compute ssresidt=csum(residt&*residt).
compute r2noint = 1-(ssresidt/sstotal).
end if.
compute pr = ncol(x)-1.
compute lmat = ident(nrow(b)).
compute lmat = lmat(:,2:ncol(lmat)).
compute f =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
pr).
compute pf = 1-fcdf(f,pr,dfres).
compute pf = {sqrt(r2),r2,msresid,f,pr,dfres,pf}.
compute xd=abs(xp2).
end if.
do if (itprob = 0).
compute tstat = b&/seb.
do if (ovals <> 2).
compute p = 2*(1-tcdf(abs(tstat), dfres)).
else if (ovals = 2).
compute p = 2*(1-cdfnorm(abs(tstat))).
end if.
compute outp = {b,seb,tstat,p}.
compute nms = {'constant'; nms; 'interact'}.
do if (ovals <> 2).
compute tval = (dfres* (exp((dfres-
(5/6))*((xd/(dfres-(2/3)+(.11/dfres)))*(xd/(dfres-
(2/3)+(.11/dfres)))))-1)).

```

```

compute outp={outp,(b-
sqrt(tval)&*seb),(b+sqrt(tval)&*seb)}.
end if.
compute bb = tval.
do if (dumok = 1).
do if (mcfoc > 0).
print dummat/title = 'Coding of categorical X
variable for analysis: '/cnames = xname2/format
= F5.2.
else if (mcmod > 0).
print dummat/title = 'Coding of categorical M
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
print yname/title =
*****
*****'/rlabels = 'Outcome: '/format
= A8.
do if (ovals = 2).
compute nmsd = {outv, 'Analysis'}.
print rcd/title = 'Coding of binary Y for
analysis: '/cnames = nmsd/format = F10.4.
compute outp = {outp,(outp(:,1)-
sqrt(tval)&*outp(:,2)),(outp(:,1)+sqrt(tval)&*outp
(:,2))}.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.

```

```

print outp/title = 'Logistic Regression
Model'/rnames = nms/clabels 'Coeff' 'se' 'Z' 'p'
'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
compute varbtmp=varb.
compute btmp=b.
compute LL2f=LL2.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=x(:,1:(ncol(x)-nx)).
compute bt1 = make(ncol(xlp),1,0).

.

compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(yp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.

```

```

do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&*ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL2f.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(LLdiff,nx).
compute rcha={LLdiff,nx,pchi}.
end if.
do if (ovals <> 2).
print pf/title = 'Model Summary'/clabels = 'R' 'R-
sq' 'MSE' 'F' 'df1' 'df2' 'p'/format F10.4.
compute lmat=make(nrow(b),1,0).
compute lmat(nrow(lmat),1)=1.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
1).
compute
rcha=((b(nrow(b),1)/seint)*(b(nrow(b),1)/seint))*
(1-r2)/dfres.
compute rcha = {rcha, fcha, 1, dfres,
outp((pr+1),4)}.
print outp/title = 'Model'/rnames = nms/clabels
'coeff' 'se' 't' 'p' 'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter

```

```

estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute rcha=r2-r2noint.
compute lmat=make((nrow(b)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
nx).
compute pvalr2c=1-fcdf(fcha,nx,dfres).
compute rcha = {rcha, fcha, nx, dfres,pvalr2c}.
end if.
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute intkey = {'a', 'b', 'c', 'd', 'e'}.
loop i = 1 to nx.
compute intkey={intkey; ddd1(1,i), ' : ', ddd(1,i),
' X ', nm(1,ncol(dd)-(1-mcloc))}.
end loop.
compute intkey=intkey(2:nrow(intkey),:).
print intkey/title='Product terms key: '/format =
a8.
end if.
do if (ovals <> 2).
print rcha/title = 'R-square increase due to
interaction: '/clabels 'R2-chng' 'F' 'df1' 'df2'
'p'/format F10.4.
else if (ovals = 2).
print rcha/title = 'Likelihood ratio test for
interaction: '/clabels 'Chi-sq' 'df' 'p'/format F10.4.
end if.
print/title =
*****
*****'.
compute mdvar = x(:,(ncol(x)-1)).
do if (mcfoc = 0 and mcmmod = 0).
compute g1 = b((ncol(x)-2),1).
compute g3 = b(ncol(x),1).

```

```

compute vg1 = varb((ncol(x)-2),(ncol(x)-2)).
compute vg3 = varb(ncol(x),ncol(x)).
compute covg1g3 = varb((ncol(x)-2), ncol(x)).
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute mdvar = x(:,(ncol(x)-(2*n))).
end if.
compute mdmin = cmin(mdvar).
compute mdmax = cmax(mdvar).
compute fvar = x(:,(ncol(x)-2)).
compute nval = ncol(design(mdvar)).
compute fvmin = cmin(fvar).
compute fvmax = cmax(fvar).
do if (mmodval = 999).
compute mnmd = csum(mdvar)/n.
compute tmp = make(n,1,mnmd).
compute sdmd = sqrt(csum(((mdvar-
tmp)**2))/(n-1)).
compute probeval = {mnmd-sdmd; mnmd;
mnmd+sdmd}.
do if (probeval(1,1) < mdmin).
compute lowwarn = 1.
compute probeval(1,1)=mdmin.
end if.
do if (probeval(nrow(probeval),1) > mdmax).
compute highwarn = 1.
compute probeval(nrow(probeval),1)=mdmax.
end if.
do if (quantile = 1).
compute tmp = mdvar.
compute tmp(GRADE(mdvar(:,1)),:) = mdvar.
compute mdvar = tmp.
compute
probeval={mdvar(trunc(0.1*n),1);mdvar(trunc(0
.25*n),1);mdvar(trunc(0.5*n),1);mdvar(trunc(0.7
5*n),1);mdvar(trunc(0.9*n),1)}.
end if.
end if.
do if (nval = 2).
compute probeval = make(2,1,0).

```



```

compute probeval(1,1)=cmin(mdvar).
loop i = 1 to n.
do if (mdvar(i,1) <> probeval(1,1)).
compute probeval(2,1) = mdvar(i,1).
BREAK.
end if.
end loop.
end if.
do if (mmodval <> 999).
compute probeval = mmodval.
end if.
compute outp = make(nrow(probeval),7,0).
do if (mcfoc > 0 or mcmmod > 0).
compute focvals=make(1,ncol(x)+1,1).
do if (mcfoc > 0).
print/title = 'Conditional effect of X on Y at
values of the moderator:'.
print/title = ' '/space=0.
compute rnn2=mdtr.
compute matt=make(nx,6,0).
end if.
loop jj=1 to nrow(probeval).
do if (mcfoc > 0).
loop ii=1 to nx.
compute g1=b((ncol(x)-(2*nx)+ii),1).
compute g3=b((ncol(x)-nx+ii),1).
compute vg1=varb((ncol(x)-(2*nx)+ii),(ncol(x)-
(2*nx)+ii)).
compute vg3=varb((ncol(x)-nx+ii),(ncol(x)-
nx+ii)).
compute covg1g3=varb((ncol(x)-
(2*nx)+ii),(ncol(x)-nx+ii)).
compute x2 = probeval(jj,1).
compute w1 = g1+g3*x2.
compute varw1 =
vg1+(2*x2*covg1g3)+((x2*x2)*vg3).
compute sew1 = sqrt(varw1).
compute t1 = w1/sew1.
do if (ovals <> 2).
compute LLCI = (w1-sqrt(tval))*sew1).

```

```

compute ULCl = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-tcdf(abs(t1), dfres)).
compute cnms = {'Coeff', 'se', 't', 'p', 'LLCl',
'ULCl'}.
compute matt(ii,:)={w1,sew1,t1,p,llci,ulci}.
else if (ovals = 2).
compute LLCl = (w1-sqrt(tval)&*sew1).
compute ULCl = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-cdfnorm(abs(t1))).
compute cnms = {'Coeff', 'se', 'Z', 'p', 'LLCl',
'ULCl'}.
compute matt(ii,:)={w1,sew1,t1,p,llci,ulci}.
end if.
end loop.
compute rnms=t(xname).
compute mdvalpr=probeval(jj,1).
print mdvalpr/title = 'Moderator
value:.'/rnames=rnn2/format= F10.4 /space=0.
print matt/title='
'/cnames=cnms/rnames=rnms/format= F10.4
/space=0.
compute xprob=x(:,(ncol(x)-(2*nx)))-mdvalpr.
loop kk = 1 to nx.
compute xprob={xprob,
(xprob(:,1)&*x(:,(ncol(x)-(2*nx)+kk)))}.
end loop.
compute xprob={x(:,1:((ncol(x)-(2*nx))-
1)),xprob}.
do if (ovals <> 2).
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute residc = y-(xprob*bmultc).
compute ssresidc=csum(residc&*residc).
compute r2c = r2-(1-(ssresidc/sstotal)).
compute fcha2=(dfres*r2c)/(nx*(1-r2)).
do if (hc3 = 1).
loop kk = 1 to nx.
compute xprob={xprob, x(:,(ncol(x)-
(2*nx)+kk))}.
end loop.

```

```

compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute k3 = nrow(bmultc).
compute xhc=xprob.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)=
xhc(i3,:)*inv(t(xprob)*xprob)*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).
end loop.
compute
varbc=(inv(t(xprob)*xprob)*t(xhc)*xhc*inv(t(xpr
ob)*xprob)).
compute lmat=make((nrow(bmultc)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha2 =
(t(t(lmat)*bmultc)*inv(t(lmat)*varbc*lmat)*((t(lm
at)*bmultc)))/nx).
end if.
compute pvalr2cc=1-fcdf(fcha2,nx,dfres).
compute rcha2 = {r2c, fcha2, nx,
dfres,pvalr2cc}.
print rcha2/title = 'Test of equality of conditional
means at this value of the moderator'/clabels
'R2-chng' 'F' 'df1' 'df2' 'p'/format F10.4.
else if (ovals = 2).
compute btmp=b.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=xprob.
compute bt1 = make(ncol(xlp),1,0).

.

compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.

```

```

loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp&*ln(pt1lp)+(1-ylp)&*ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL4.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(LLdiff,nx).
compute rcha2 = {LLdiff, nx,pchi}.

```

```

print rcha2/title = 'Test of equality of log odds
conditioned on this moderator value'/clabels
'Chi-sq' 'df' 'p'/format F10.4.
end if.
end if.
compute
tttt=make(nrow(dummat),1,probeval(jj,1)).
compute tttt={tttt,dummat(:,2:ncol(dummat))}.
loop kkk=1 to nx.
compute tttt={tttt,tttt(:,1)*tttt(:,1+kkk)}.
end loop.
compute ones=make(nrow(dummat),1,1).
do if (ncol(x) > (2*nx+2)).
compute
covmnmat=make(nrow(tttt),ncol(covmns),0).
loop kkk=1 to nrow(tttt).
compute covmnmat(kkk,:)=covmns.
end loop.
compute tttt={covmnmat,tttt}.
else.
compute tttt={ones,tttt}.
end if.
compute focvals={focvals;dummat(:,1),tttt}.
do if (mcfoc > 0).
compute yhat={dummat(:,1),(tttt*b)}.
compute cmnms={fciv, 'yhat'}.
do if (ovals <> 2).
print yhat/title = 'Estimated conditional means
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (ovals = 2).
print yhat/title = 'Estimated conditional log odds
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (jj <> nrow(probeval)).
print/title='-----'/space=0.
end if.
end if.

```

```

end loop.
compute focvals=focvals(2:nrow(focvals),:).
compute yhat=focvals(:,2:ncol(focvals))*b.
compute
focvals={focvals(:,1),focvals(:,(ncol(focvals)-
(2*nx))),yhat}.
compute cnms={fciv,mdtr,'yhat'}.
do if (mcmod > 0).
compute cnms={mdtr,fciv,'yhat'}.
end if.
end if.
do if (mcmod > 0).
compute outp=make((nx+1),7,0).
compute bcatm={b((ncol(x)-
(2*nx)),1);b((ncol(x)-nx+1):ncol(x),1)}.
compute outp(:,1)=dummat(:,1).
compute bcatcov=varb((ncol(x)-
nx):ncol(x),(ncol(x)-nx):ncol(x)).
compute bcatcov(1,1)=varb((ncol(x)-
(2*nx)),(ncol(x)-(2*nx))).
compute
bcatcov(2:nrow(bcatcov),1)=varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx)).
compute
bcatcov(1,2:nrow(bcatcov))=t(varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx))).
loop i = 1 to nrow(dummat).
compute
catmval={1,dummat(i,2:ncol(dummat))}.
compute condeff=catmval*bcatm.
compute
condse=sqrt(catmval*bcatcov*t(catmval)).
compute outp(i,2:3)={condeff,condse}.
end loop.
compute outp(:,4)=outp(:,2)&/outp(:,3).
compute outp(:,5) = 2*(1-tcdf(abs(outp(:,4)),
dfres)).
compute outp(:,6) = (outp(:,2)-
sqrt(tval)&*outp(:,3)).

```

```

compute outp(:,7) =
(outp(:,2)+sqrt(tval)&*outp(:,3)).
do if (ovals <> 2).
compute cnmms = {xname2(1,1), 'coeff', 'se',
't', 'p', 'LLCI', 'ULCI'}.
end if.
do if (ovals = 2).
compute outp(:,5) = 2*(1-
cdfnorm(abs(outp(:,4)))).
compute cnmms = {xname2(1,1), 'coeff', 'se',
'Z', 'p', 'LLCI', 'ULCI'}.
end if.
print outp/title = 'Conditional Effect of Focal
Predictor in Groups Defined by the Moderator
Variable:.'/cnames=cnmms/format = F10.4.
end if.
do if (nval > 2 and (mmodval = 999) and
mcmmod = 0).
do if (quantile <> 1).
print/title = 'Moderator values are the sample
mean and plus/minus one SD from mean'.
else.
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
end if.
do if (highwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD above the mean was'.
print/title = 'replaced with the maximum
because one SD above the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
do if (lowwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD below the mean was'.
print/title = 'replaced with the minimum
because one SD below the mean is
outside'/space=0.

```

```

print/title = 'of the range of the data.'/space=0.
end if.
end if.
do if (nval = 2 and (mmodval = 999)).
print/title = 'The moderator variable is
dichotomous'.
end if.
do if (mmodval <> 999 and mcmmod = 0 and
((mmodval < mdmin) or (mmodval > mdmax))).
print/title = 'Warning: Moderator value specified
is outside of the range of the data'.
end if.
do if (plot = 1).
do if (mcfoc = 0 and mcmmod=0).
compute cnms = {t(nms((ncol(x)-2):(ncol(x)-
1),1)), 'yhat'}.
end if.
do if (ovals = 2).
compute prob =
exp(focvals(:,3))/(1+exp(focvals(:,3))).
compute focvals = {focvals, prob}.
compute cnms = {cnms, 'prob'}.
end if.
print/title =
*****
*****!

print/title = 'Data for visualizing conditional
effect of X on Y'.

```



```

print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.' /space=0.
do if (ovals = 2).
do if (mcfoc=0).
print/title = 'DATA LIST
FREE/Technological_Advan Plan_Orga_Pract
logodds prob.' /space=1.
end if.
do if (mcmod=0).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Technological_Advan
logodds prob.' /space=1.
end if.
else if (ovals <> 2).
do if (mcfoc=0).
print/title = 'DATA LIST
FREE/Technological_Advan Plan_Orga_Pract
Operational_Performance.' /space=1.
end if.
do if (mcmod=0).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Technological_Advan
Operational_Performance.' /space=1.
end if.
end if.
print/title = 'BEGIN DATA.' /space=0.
print focvals/title = ' /format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (ovals <> 2 and mcmod=0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH Operational_Performance BY
Plan_Orga_Pract.' /space=0.
end if.
do if (ovals <> 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH Operational_Performance BY
Technological_Advan.' /space=0.

```

```

end if.
do if (ovals = 2 and mcmmod=0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Plan_Orga_Pract.'
/space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Plan_Orga_Pract.' /space=0.
end if.
do if (ovals = 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH logodds BY Technological_Advan.'
/space=0.
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH prob BY Technological_Advan.'
/space=0.
end if.
do if (ncovs > 0).
print/title = 'NOTE: For data above, covariates
are set to their sample means.'.
end if.
end if.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
end if.
END IF.
end if.
do if (bad > 0).
compute note(notes,1) = 9.
compute notes = notes + 1.

```

```

end if.
print/title = '***** ANALYSIS
NOTES AND WARNINGS
*****'.

loop i = 1 to errs.
do if (runerrs(i,1) = 1).
print/title = 'ERROR: One of your declared
mediators is dichotomous. This procedure
can"t be used.'.
end if.
do if (runerrs(i,1) = 2).
print/title = 'ERROR: For model 6, this
procedure limits the number of mediators to
four.'.
end if.
do if (runerrs(i,1) = 3).
print/title = 'ERROR: For models 1, 2, and 3,
only a single variable can be listed in the M
list.'.
end if.
do if (runerrs(i,1) = 4).
print/title = 'ERROR: You requested a model
involving W but did not provide a valid W
variable name.'.
end if.
do if (runerrs(i,1) = 5).
print/title = 'ERROR: You requested a model
involving Z but did not provide a valid Z
variable name.'.
end if.
do if (runerrs(i,1) = 6).
print/title = 'ERROR: You requested a model
involving Q but did not provide a valid Q
variable name.'.
end if.
do if (runerrs(i,1) = 7).
print/title = 'ERROR: You requested a model
involving V but did not provide a valid V
variable name.'.
end if.

```

```
do if (runerrs(i,1) = 8).
print/title = 'ERROR: You specified a W
variable for a model that does not need it'.
end if.
do if (runerrs(i,1) = 9).
print/title = 'ERROR: You specified a Z variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 10).
print/title = 'ERROR: You specified a Q variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 11).
print/title = 'ERROR: You specified a V variable
for a model that does not need it.'.
end if.
do if (runerrs(i,1) = 12).
print/title = 'ERROR: The variable specified for
W has already been assigned.'.
end if.
do if (runerrs(i,1) = 13).
print/title = 'ERROR: The variable specified for
Z has already been assigned.'.
end if.
do if (runerrs(i,1) = 14).
print/title = 'ERROR: The variable specified for
Q has already been assigned.'.
end if.
do if (runerrs(i,1) = 15).
print/title = 'ERROR: The variable specified for
V has already been assigned.'.
end if.
do if (runerrs(i,1) = 16).
print/title = 'ERROR: You did not provide a
valid Y variable name.'.
end if.
do if (runerrs(i,1) = 17).
print/title = 'ERROR: The variable specified for
Y has already been assigned.'.
end if.
```

```
do if (runerrs(i,1) = 18).
print/title = 'ERROR: Model 6 requires more
than one mediator.'.
end if.
do if (runerrs(i,1) = 19).
print/title = 'ERROR: You have not specified a
valid model number.'.
end if.
do if (runerrs(i,1) = 20).
print/title = 'ERROR: At least one and only one
variable must be listed for X.'.
end if.
do if (runerrs(i,1) = 21).
print/title = 'ERROR: At least one and only one
variable must be listed for Y.'.
end if.
do if (runerrs(i,1) = 22).
print/title = 'ERROR: Iteration didn't converge
to a solution. Interpret results with caution.'.
end if.
do if (runerrs(i,1) = 23).
print/title = 'ERROR: You specified a clustering
variable that does not exist in your variable
list.'.
end if.
do if (runerrs(i,1) = 24).
print/title = 'ERROR: You specified a clustering
variable that has already been assigned.'.
end if.
do if (runerrs(i,1) = 25).
print/title = 'ERROR: One or more of your M
variables is not listed in the variables list.'.
end if.
do if (runerrs(i,1) = 26).
print/title = 'ERROR: A maximum of 20 cluster
units is allowed. Use multilevel modeling
instead.'.
end if.
do if (runerrs(i,1) = 27).
```

```

print/title = 'ERROR: One of the variables in
your model is a constant.'.
end if.
do if (runerrs(i,1) = 28).
print/title = 'ERROR: Dichotomous Y is not
permitted with WS option.'.
end if.
do if (runerrs(i,1) = 29).
print/title = 'ERROR: Insufficient number of
variables in vars= list when using WS option.'.
end if.
do if (runerrs(i,1) = 30).
print/title = 'ERROR: Too many variables in
vars= list when using WS option. Covariates
not allowed.'.
end if.
do if (runerrs(i,1) = 31).
print/title = 'ERROR: mmodval and wmodval
can"t both be set to zero with contrast option.'.
end if.
do if (runerrs(i,1) = 32).
print/title = 'ERROR: You did not provide a
valid X variable name.'.
end if.
do if (runerrs(i,1) = 33).
print/title = 'ERROR: PROCESS requires all
variable names to be eight characters or
fewer.'.
print/title = '      Please shorten variable names
and reexecute.'/space=0.
end if.
do if (runerrs(i,1)=34).
print/title = 'ERROR: X and M cannot both be
specified as multicategorical.'.
end if.
do if (runerrs(i,1)=35).
print/title = 'ERROR: Multicategorical variable
specification available only for models 1 and
4.'.
end if.

```

```

do if (runerrs(i,1)=36).
print/title = 'ERROR: Categorical variables
cannot have more than 10 categories.'.
end if.
do if (runerrs(i,1)=37).
print/title = 'ERROR: Each group must have at
least two cases.'.
end if.
do if (runerrs(i,1)=38).
print/title = 'ERROR: Cluster option not
available with multicategorical variables.'.
end if.
do if (runerrs(i,1)=39).
print/title = 'ERROR: A maximum of 10
variables is allowed in the M= list.'.
end if.
do if (runerrs(i,1) = 41).
print/title = 'ERROR: MCM option not permitted
in model 4.'.
end if.
end loop.
do if (errs = 0).
do if (boot > 1 or mc > 0).
do if (bconoff = 1 and boot > 1).
print boot/title = 'Number of bootstrap samples
for bias corrected bootstrap confidence
intervals:' /format = F8.0.
end if.
do if (bconoff = 0 and boot > 1).
print boot/title = 'Number of bootstrap samples
for percentile bootstrap confidence
intervals:' /format = F8.0.
end if.
do if (mc > 1).
print mc/title = 'Number of samples for Monte
Carlo confidence intervals:' /format = F8.0.
end if.
do if (booterr = 1).
compute badend = badend(1,2:ncol(badend)).

```

```

print badend/title = 'WARNING: Bootstrap CI
endpoints below not trustworthy. Decrease
confidence or increase bootstraps'/format =
F10.4.
end if.
end if.
print conf/title = 'Level of confidence for all
confidence intervals in output:.'/format = F8.2.
do if ((center = 1 or ws=1) and (ncol(centvar) >
1)).
compute centvar = centvar(1,2:ncol(centvar)).
print centvar/title = 'NOTE: The following
variables were mean centered prior to
analysis:.'/format = a8.
end if.
loop i = 1 to notes.
do if (note(i,1) = 1).
print/title = 'NOTE: Confidence level restricted
to between 50 and 99.9999%. 95% confidence
is provided in output'.
end if.
do if (note(i,1) = 2).
print/title = 'NOTE: Effect size measures for
indirect effects are not available for models
with dichotomous outcomes'.
end if.
do if (note(i,1) = 3).
print/title = 'NOTE: All standard errors for
continuous outcome models are based on the
HC3 estimator'.
end if.
do if (note(i,1) = 6).
print/title = 'NOTE: The number of bootstrap
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 7).
print/title = 'NOTE: The Johnson-Neyman
method is available only for Models 1 and 3'.
end if.

```



```

do if (note(i,1) = 8).
print/title = 'NOTE: The Johnson-Neyman
method cannot be used with a dichotomous
moderator'.
end if.
do if (note(i,1)=9).
print bad/title = 'NOTE: Some bootstrap
samples had to be replaced. The number of
such replacements was:'.
end if.
do if (note(i,1) = 11).
print nmiss/title = 'NOTE: Some cases were
deleted due to missing data. The number of
such cases was:'.
end if.
do if (note(i,1) = 12).
print /title = 'NOTE: Monte Carlo method
available only for models 4 and 5.
Bootstrapping was used instead.'.
end if.
do if (note(i,1) = 13).
print/title = 'NOTE: The number of Monte Carlo
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 19).
print/title = 'NOTE: Effect sizes not available for
within-subject analyses.'.
end if.
do if (note(i,1) = 16).
print/title = 'NOTE: Normal theory tests not
available for within-subject analyses.'.
end if.
do if (note(i,1) = 17).
print/title = 'NOTE: Monte Carlo confidence
intervals not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 18 and warnrep=0).

```

```
print/title = 'WARNING: You have requested
OLS estimation with a dichotomous criterion.'.
print/title = 'Interpret model coefficients and
inferential statistics with caution.'/space=0.
compute warnrep=1.
end if.
do if (note(i,1) = 20).
print/title = 'NOTE: Saving of bootstrap
estimates not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 22).
print/title = 'NOTE: Effect size option with
covariates requires covariates in models of M
and Y.'.
end if.
do if (note(i,1) = 23).
print/title = 'NOTE: Johnson-Neyman method
not available with multicategorical X'.
end if.
do if (note(i,1) = 24).
print/title = 'NOTE: MMODVAL option not
available with a multicategorical moderator'.
end if.
do if (note(i,1) = 25).
print/title = 'NOTE: Omnibus test for relative
indirect effects is not available with the MC
option.'.
end if.
do if (note(i,1) = 26).
print/title = 'NOTE: CONTRAST option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 27).
print/title = 'NOTE: EFFSIZE option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 28).
print/title = 'NOTE: NORMAL option not
available with multicategorical X.'.
```

| | | |
|-----------|----------------|---|
| | | end if. do if (note(i,1) = 31). print/title = 'NOTE: The TOTAL option is not available when Y is dichotomous.' end if. do if (note(i,1) = 32). print/title = 'NOTE: Kappa-squared is disabled from output as of version 2.16.' end if. do if (note(i,1) = 33). print/title = 'NOTE: COVMY option ignored in models 1, 2, and 3.' end if. end loop. end if. end matrix. |
| Resources | Processor Time | 00:00:01.05 |
| | Elapsed Time | 00:00:01.01 |

Run MATRIX procedure:

***** PROCESS Procedure for SPSS Release 2.16.3 *****

Written by Andrew F. Hayes, Ph.D. www.afhayes.com
 Documentation available in Hayes (2013). www.guilford.com/p/hayes3

Model = 1
 Y = Operatio
 X = Plan_Org
 M = Technolo

Sample size
 856

Outcome: Operatio

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|----------|--------|----------|-------|
| .8529 | .7274 | .3274 | 757.8709 | 3.0000 | 852.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|---------|-------|---------|-------|---------|--------|
| constant | -1.0652 | .1595 | -6.6802 | .0000 | -1.3782 | -.7523 |
| Technolo | .9710 | .0621 | 15.6282 | .0000 | .8491 | 1.0930 |
| Plan_Org | .8717 | .0600 | 14.5314 | .0000 | .7540 | .9895 |

| | | | | | | |
|-------|--------|-------|---------|-------|--------|--------|
| int_1 | -.1433 | .0148 | -9.6675 | .0000 | -.1724 | -.1142 |
|-------|--------|-------|---------|-------|--------|--------|

Product terms key:

| | | | |
|-------|----------|---|----------|
| int_1 | Plan_Org | X | Technolo |
|-------|----------|---|----------|

R-square increase due to interaction(s):

| | | | | | |
|-------|---------|---------|--------|----------|-------|
| | R2-chng | F | df1 | df2 | p |
| int_1 | .0299 | 93.4610 | 1.0000 | 852.0000 | .0000 |

Conditional effect of X on Y at values of the moderator(s):

| Technolo | Effect | se | t | p | LLCI | ULCI |
|----------|--------|-------|---------|-------|-------|-------|
| 2.7372 | .4794 | .0387 | 12.3768 | .0000 | .4034 | .5554 |
| 3.8107 | .3255 | .0397 | 8.1983 | .0000 | .2476 | .4034 |
| 4.8843 | .1716 | .0465 | 3.6936 | .0002 | .0804 | .2628 |

Values for quantitative moderators are the mean and plus/minus one SD from mean.

Values for dichotomous moderators are the two values of the moderator.

***** JOHNSON-NEYMAN TECHNIQUE *****

There are no statistical significance transition points within the observed range of the moderator.

Data for visualizing conditional effect of X on Y

Paste text below into a SPSS syntax window and execute to produce plot.

DATA LIST FREE/Plan_Orga_Pract Technological_Advan Operational_Performance.
BEGIN DATA.

| | | |
|--------|--------|--------|
| 2.8484 | 2.7372 | 2.9582 |
| 3.8575 | 2.7372 | 3.4419 |
| 4.8666 | 2.7372 | 3.9257 |
| 2.8484 | 3.8107 | 3.5623 |
| 3.8575 | 3.8107 | 3.8908 |
| 4.8666 | 3.8107 | 4.2193 |
| 2.8484 | 4.8843 | 4.1664 |
| 3.8575 | 4.8843 | 4.3396 |
| 4.8666 | 4.8843 | 4.5128 |

END DATA.

GRAPH/SCATTERPLOT=Plan_Orga_Pract WITH Operational_Performance BY
Technological_Advan.

***** ANALYSIS NOTES AND WARNINGS *****

Level of confidence for all confidence intervals in output:
95.00

----- END MATRIX -----

restore.

Matrix

| Notes | | |
|----------------|--------------------------------|----------------------|
| Output Created | | 20-OCT-2020 16:04:02 |
| Comments | | |
| Input | Active Dataset | DataSet1 |
| | Filter | <none> |
| | Weight | <none> |
| | Split File | <none> |
| | N of Rows in Working Data File | 856 |

Syntax

```
matrix.  
get dat/file = */variables =  
Operational_Performance Opra_Pract  
Technological_Advan /names =  
vnames/missing = 9999.  
compute ninit = nrow(dat).  
get dat/file = */variables =  
Operational_Performance Opra_Pract  
Technological_Advan /names =  
vnames/missing = omit.  
get tmp/file = */variables =  
Operational_Performance /names =  
yname/missing = omit.  
get tmp2/file = */variables = Opra_Pract  
/names = xname/missing = omit.  
get tmp/file = */variables =  
Technological_Advan /names =  
mnames/missing = omit.  
get tmp/file = */variables = w999999t z999999t  
v999999t q999999t.  
compute wname=tmp(1,1).  
do if (wname = ' ').  
compute wname = 'xxx'.  
end if.  
compute zname=tmp(1,2).  
do if (zname = ' ').  
compute zname = 'xxx'.  
end if.  
compute vname=tmp(1,3).  
do if (vname = ' ').  
compute vname = 'xxx'.  
end if.  
compute qname=tmp(1,4).  
do if (qname = ' ').  
compute qname = 'xxx'.  
end if.  
compute n = nrow(dat).  
compute p0=-.322232431088.  
compute p1 = -1.  
compute p2 = -.342242088547.
```

```

compute p3 = -.0204231210245.
compute p4 = -.0000453642210148.
compute q0 = .0993484626060.
compute q1 = .588581570495.
compute q2 = .531103462366.
compute q3 = .103537752850.
compute q4 = .0038560700634.
compute badend = 0.
compute priorlo = -9999999.
compute priorhi = 9999999.
compute criterr = 0.
compute cluster = 0.
compute clsdmy = 0.
compute jndich = 0.
compute wvdich=0.
compute zqdich=0.
compute mod74dic=0.
compute warnrep=0.
compute contrast = trunc( 0 ).
do if (contrast > 2 or contrast < 0).
compute contrast = 0.
end if.
compute booterr = 0.
compute effsize = ( 0 = 1).
compute note = make(10,1,0).
compute notes = 1.
compute iterr = 0.
compute clsmth = 0.
compute quantile = ( 0 = 1).
compute mmodval= 999.
compute jn = ( 1 = 1).
compute covcoeff=( 0 =1).
compute center = ( 0 = 1).
compute detail = ( 1 = 1).
compute coeffci = ( 1 = 1).
compute conf = 95.
compute longname = ( 1 =1).
compute bconoff=( 0 <> 1).
compute covmy = trunc( 0 ).
do if (covmy < 0 or covmy > 2).

```

```

compute covmy = 0.
end if.
do if (trunc(conf) >= 100 or (trunc(conf) <=
50)).
compute conf = 95.
compute note(notes,1) = 1.
compute notes = notes + 1.
end if.
do if (n < ninit).
compute nmiss = ninit-n.
compute note(notes,1) = 11.
compute notes = notes + 1.
end if.
compute errs = 0.
compute quantd = {0,0,0,0,0,0}.
compute quantc = {0,0,0,0,0,0}.
compute mcheck = 0.
compute ttt = 0.
compute plot = trunc( 1 ).
compute runerrs = make(50,1,0).
compute model = trunc( 1 ).
compute normal = 0.
compute ws= ( 0 =1).
compute olsdichm= ( 0 =1).
compute olsdichy= ( 0 =1).
compute mcx=trunc( 0 ).
compute mcm=trunc( 0 ).
do if (ws = 1).
do if (effsize = 1).
compute note(notes,1) = 19.
compute notes = notes + 1.
compute effsize=0.
end if.
do if (normal=1).
compute note(notes,1) = 16.
compute notes = notes + 1.
compute normal=0.
end if.
end if.
do if (model <> 4).

```



```

compute ws = 0.
end if.
do if (model > 0 and model < 4 and covmy <>
0).
compute covmy=0.
compute note(notes,1) = 33.
compute notes = notes + 1.
end if.
print/space=0.
do if (jn = 1 and model <> 1 and model <> 3).
compute note(notes,1) = 7.
compute notes = notes + 1.
end if.
do if ((model > 76) or (model < 1)).
compute model = 77.
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 19.
end if.
compute itprob=0.
do if (mcx < 0).
compute mcx=0.
end if.
do if (mcx > 4).
compute mcx=0.
end if.
do if (mcm < 0).
compute mcm=0.
end if.
do if (mcm > 4).
compute mcm=0.
end if.
do if (mcx > 0 and mcm > 0).
compute errs=errs+1.
compute runerrs(errs,1)=34.
compute criterr=1.
end if.
do if ((model <> 1 and model <> 4) and (mcx >
0 or mcm > 0)).
compute errs=errs+1.

```

```

compute runerrs(errs,1)=35.
compute criterr=1.
end if.
do if ((model = 4) and (mcm > 0)).
compute errs=errs+1.
compute runerrs(errs,1)=41.
compute criterr=1.
end if.
do if ((mcx > 0) and jn <> 0).
compute jn=0.
compute note(notes,1) = 23.
compute notes = notes + 1.
end if.
do if (mmodval <> 999 and mcm > 0).
compute note(notes,1) = 24.
compute notes = notes + 1.
end if.
compute toteff = 0.
compute toteff = (( 0 = 1)*( 1 = 4 or 1 = 6)).
compute varorder = 2.
compute hc3 = ( 0 <> 0).
compute cname = 'xxx'.
compute centvar = {'xxx'}.
compute nmeds = ncol(mnames).
do if (longname=0).

```

```
compute criterr = 1.  
compute errs = errs+1.  
compute runerrs(errs,1) = 33.
```

1

2

3

end if.

compute modelm =

```
{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,2;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,3;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,5;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,6;  
1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,7;  
1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1,8;  
1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,9;  
1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,3,10;  
1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,5,11;  
1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,5,12;  
1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,5,13;  
0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,2,14;  
0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,2,15;  
0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,4,16;
```

0,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,4,17;
0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,6,18;
0,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,6,19;
0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,6,20;
1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,7,21;
1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,7,22;
1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,23;
1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,0,24; 1,1,1,1

,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,25;
1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,0,0,26;
1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,0,0,27;
1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,7,28;
1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,0,7,29;
1,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,30;
1,1,0,1,0,0,1,1,0,1,0,0,0,0,0,0,0,31;
1,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,0,32;
1,1,1,1,0,0,1,1,1,1,0,0,0,0,0,0,0,33;
1,1,1,1,0,0,1,0,0,1,0,0,0,0,0,0,0,34;
1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,35;
1,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,36;
1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,37;
1,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,38;
1,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,39;
1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,0,40;
1,0,0,1,1,0,1,0,0,1,1,0,0,0,0,0,0,41;
1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,42;
1,0,0,1,1,1,1,0,0,1,1,1,0,0,0,0,0,43;
1,0,0,1,1,1,1,0,0,1,0,0,0,0,0,0,0,44;
1,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,45;
1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,46;
1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,47;
1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,48;
1,1,0,1,1,0,1,1,0,0,0,

0,0,0,0,0,0,0,0,49;
1,1,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,50;
1,1,1,1,1,0,1,1,1,0,0,0,0,0,0,0,0,51;
1,1,1,1,1,0,0,0,0,1,1,0,0,0,0,0,0,52;
1,1,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,53;

```

1,1,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,54;
1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,55;
1,1,1,1,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,56;
1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,57;
1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,58;
1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,59;
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,60;
1,1,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,61;
1,1,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,62;
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,0,0,0,0,63;
1,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,64;
1,0,0,1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,65;
1,0,0,1,0,0,0,0,0,0,1,0,0,1,0,0,0,0,0,66;
1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,67;
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,68;
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,69;
1,0,0,1,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,70;
1,0,0,1,0,0,1,0,0,1,0,0,1,1,1,0,0,0,0,71;
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,1,1,0,0,72;
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,1,1,0,0
,73; 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,74;
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,75;
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,1,0,0,0,76;
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,77}.
compute wm = modelm(model, 1).
compute zm = modelm(model, 2).
compute wzm = modelm(model, 3).
compute vy = modelm(model, 4).
compute qy = modelm(model, 5).
compute vqy = modelm(model, 6).
compute wy = modelm(model, 7).
compute zy = modelm(model, 8).
compute wzy = modelm(model, 9).
compute vxy = modelm(model, 10).
compute qxy = modelm(model, 11).
compute vqxy = modelm(model, 12).
compute wmy = modelm(model, 13).
compute wvmy = modelm(model, 14).
compute wvxy = modelm(model, 15).

```

```

compute zmy = modelm(model,16).
compute wzmy = modelm(model,17).
compute xmy = modelm(model,18).
compute imm=modelm(model,19).
do if (model = 74).
compute vdich=0.
end if.
do if (ncol(xname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 20.
compute criterr = 1.
end if.
do if (ncol(mnames) > 10).
compute errs = errs+1.
compute runerrs(errs,1) = 39.
compute criterr = 1.
end if.
do if (ncol(yname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 21.
compute criterr = 1.
end if.
do if (( 999 =0) and ( 999 =0) and (model=3 or
model=2) and (contrast=1)).
compute errs = errs+1.
compute runerrs(errs,1) = 31.
compute criterr = 1.
end if.
compute xlist = (wm or zm or wzm or wy or zy
or wzy or vxy or qxy or vqxy or wvxy or xmy).
compute mlist = (vy or qy or vqy or zmy or wmy
or wzmy or xmy or (model < 4)).
compute bad=0.
do if (criterr = 0).
compute werr = 0.
compute verr = 0.
compute qerr = 0.
compute zerr = 0.
compute yerr = 1.
compute xerr = 1.

```



```

compute wlist = (wm or wzm or wy or wzy or
wm or wvmy or wvmy or wvxy or wzmy).
do if (wlist = 1 and wname = 'xxx').
compute werr = 1.
compute wlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (wlist = 1 and ((wname = qname) or
(wname = vname) or (wname = zname) or
(wname = xname) or (wname = yname))).
compute werr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 12).
end if.
compute zlist = (zm or wzm or zy or wzy or
zmy or wzmy).
do if (zlist = 1 and zname = 'xxx').
compute zerr = 1.
compute zlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (zlist = 1 and ((zname = qname) or
(zname = vname) or (zname = wname) or
(zname = xname) or (zname = yname))).
compute zerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 13.
end if.
compute qlist = (qy or vqy or qxy or vqxy).
do if (qlist = 1 and qname = 'xxx').
compute qerr = 1.
compute qlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (qlist = 1 and ((qname = zname) or
(qname = vname) or (qname = wname) or
(qname = xname) or (qname = yname))).

```

```

compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 14.
end if.
compute vlist = (vy or vqy or vxy or vqxy or
wvmy or wvxy).
do if (vlist = 1 and vname = 'xxx').
compute verr = 1.
compute vlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (vlist = 1 and ((vname = zname) or
(vname = qname) or (vname = wname) or
(vname = xname) or (vname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 15.
end if.
do if (wlist = 0 and wname <> 'xxx').
compute werr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 8.
end if.
do if (zlist = 0 and zname <> 'xxx').
compute zerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 9.
end if.
do if (qlist = 0 and qname <> 'xxx').
compute qerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 10.
end if.
do if (vlist = 0 and vname <> 'xxx').
compute verr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 11.
end if.
do if (hc3 = 1).

```

```

compute note(notes,1) = 3.
compute notes = notes+1.
end if.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute cons = make(n,1,1).
compute temp = (n*sscp(dat))-
(t(csum(dat))*(csum(dat))).
compute temp = temp/(n*(n-1)).
compute temp = csum(((diag(temp))=0)).
compute temp2=1.
do if (ws = 1).
compute temp2 = (n*sscp(tmp2))-
(t(csum(tmp2))*(csum(tmp2))).
compute temp2 = temp2/(n*(n-1)).
end if.
do if ((temp > 0 and ws = 0) or (temp > 0 and
ws = 1 and temp2 > 0)).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 27.
end if.
compute nmeds = ncol(mnames).
compute mcmats=ident(nmeds*2).
compute mccoef=make(nmeds*2,1,0).
compute sobel = make(nmeds,4,-999).
do if (model = 6 and nmeds > 4).
compute errs = errs+1.
compute runerrs(errs,1)=2.
end if.
do if (model < 4 and nmeds > 1).
compute errs = errs+1.
compute runerrs(errs,1) = 3.
end if.
compute nmods = (model = 74).
compute bad = 0.
compute intcnt = 1.

```

```

compute modvals = 0.
compute modvalsd = 0.
compute yintemp = {'int_1', 'int_2', 'int_3',
'int_4', 'int_5', 'int_6', 'int_7', 'int_8', 'int_9',
'int_10', 'int_11', 'int_12', 'int_13', 'int_14',
'int_15'}.
compute yintemp = {yintemp, 'int_16', 'int_17',
'int_18', 'int_19', 'int_20', 'int_21', 'int_22',
'int_23', 'int_24', 'int_25', 'int_26', 'int_27',
'int_28'}.
compute yintemp = {yintemp, 'int_29', 'int_30',
'int_31', 'int_32', 'int_33', 'int_34', 'int_35',
'int_36', 'int_37', 'int_38', 'int_39', 'int_40'}.
compute yintemp = {yintemp, 'int_41', 'int_42',
'int_43', 'int_44', 'int_45', 'int_46', 'int_47',
'int_48', 'int_49', 'int_50', 'int_51', 'int_52'}.
compute yintemp = {yintemp, 'int_53', 'int_54',
'int_55', 'int_56', 'int_57', 'int_58', 'int_59',
'int_60', 'int_61', 'int_62', 'int_63', 'int_64'}.
compute
cntname={'(C1)';'(C2)';'(C3)';'(C4)';'(C5)';'(C6)';(
C7)';'(C8)';'(C9)';'(C10)';'(C11)';'(C12)';'(C13)';(
C14)';'(C15)';'(C16)';'(C17)'}.
compute
cntname={cntname;'(C18)';'(C19)';'(C20)';'(C21
)';'(C22)';'(C23)';'(C24)';'(C25)';'(C26)';'(C27)';(
C28)';'(C29)';'(C30)';'(C31)'}.
compute
cntname={cntname;'(C32)';'(C33)';'(C34)';'(C35
)';'(C36)';'(C37)';'(C38)';'(C39)';'(C40)';'(C41)';(
C42)';'(C43)';'(C44)';'(C45)'}.
compute
cntname={cntname;'(C46)';'(C47)';'(C48)';'(C49
)';'(C50)';'(C51)';'(C52)';'(C53)';'(C54)';'(C55)';(
C56)';'(C57)';'(C58)';'(C59)'}.
compute
cntname={cntname;'(C60)';'(C61)';'(C62)';'(C63
)';'(C64)';'(C65)';'(C66)';'(C67)';'(C68)';'(C69)';(
C70)';'(C71)';'(C72)';'(C73)'}.

```

```

compute
cntname={cntname;'(C74)';'(C75)';'(C76)';'(C77
)';'(C78)';'(C79)';'(C80)';'(C81)';'(C82)';'(C83)';'(
C84)';'(C85)';'(C86)';'(C87)}.
compute
cntname={cntname;'(C88)';'(C89)';'(C90)';'(C91
)';'(C92)';'(C93)';'(C94)';'(C95)';'(C96)';'(C97)';'(
C98)';'(C99)';'(C100)';'(C101)}.
compute
cntname={cntname;'(C102)';'(C103)';'(C104)';'(
C105)}.
compute apathnam={'a path'; 'a1 path'; 'a2
path'; 'a3 path'; 'a4 path'; 'a5 path'; 'a6 path';
'a7 path'; 'a8 path'; 'a9 path'; 'a10 path'}.
compute bpathnam={'b path'; 'b1 path'; 'b2
path'; 'b3 path'; 'b4path'; 'b5path'; 'b6 path'; 'b7
path'; 'b8 path'; 'b9 path'; 'b10 path'}.
compute modvnm = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute modvnm2 = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute mlab = { ' M1 ='; ' M2 ='; ' M3 ='; '
M4 ='; ' M5 ='; ' M6 ='; ' M7 ='; ' M8 ='; '
M9 ='; ' M10 ='.
compute m = make(n,nmeds,1).
compute mmat = make(16,nmeds,0).
compute ymat = make(8,nmeds,0).
compute deco = make(10,1,0).
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
compute iterate = abs(trunc( 10000 )).
compute converge = abs( 0.00000001 ).
compute boot = abs(trunc( 5000 )).
do if (ws=1 and 0 > 0).
compute note(notes,1)=17.
compute notes=notes+1.
end if.
compute mc=(abs(trunc( 0 ))*(1-ws)).

```

```

do if (mc > 0 and model > 5).
do if (boot = 0).
compute boot=mc.
end if.
compute mc=0.
compute note(notes,1)=12.
compute notes = notes+1.
end if.
compute adjust = 0.
do if (mc > 0 and (model > 3 and model < 6)).
compute boot=0.
compute bconoff=0.
end if.
do if (boot > 0 and mc > 0).
compute mc = 0.
end if.
compute savboot = 0.
compute saveboot = ( 0 = 1).
do if (ws=1 and saveboot=1).
compute saveboot=0.
compute note(notes,1) = 20.
compute notes = notes + 1.
compute normal=0.
end if.
do if (saveboot =1 and boot > 0 and model >
3).
compute savboot = 1.
end if.
do if (boot <> 0 or mc <> 0).
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
loop.
compute cilow = rnd(bootsz*(1-(conf/100))/2).
compute cihigh =
trunc((bootsz*(conf/100)+(bootsz*(1-
(conf/100))/2)))+1.
do if (cilow < 1 or cihigh > bootsz).

```

```

compute
bootz=trunc((bootz+1000)/1000)*1000.
compute adjust = 1.
end if.
end loop if (cilow gt 0 and cihigh le bootz).
compute boot=bootz.
do if (mc > 0).
compute mc = bootz.
end if.
do if (boot > 0 and mc > 0 and (model > 3 and
model < 6)).
compute boot=0.
end if.
do if (adjust = 1 and boot > 0).
compute note(notes,1)=6.
compute notes = notes+1.
end if.
do if (adjust = 1 and mc > 0).
compute note(notes,1)=13.
compute notes = notes+1.
end if.
end if.
do if (model = 6 and nmeds > 1).
compute mmpaths =
make((nmeds+2),(nmeds+2),0).
do if (nmeds = 2).
compute indboot = make(boot+1, 3, 999).
else if (nmeds = 3).
compute indboot = make(boot+1, 7, 999).
else if (nmeds = 4).
compute indboot = make(boot+1, 15, 999).
end if.
compute indlbl = {'Total:','Ind1 :','Ind2 :','Ind3
:','Ind4 :','Ind5 :','Ind6 :','Ind7 :','Ind8 :','Ind9
:','Ind10:','Ind11:','Ind12:','Ind13:','Ind14:','
Ind15:'}.
compute indlbl2 = {'Ind1','Ind2','Ind3','Ind4';
'Ind5','Ind6','Ind7','Ind8','Ind9','Ind10';
'Ind11','Ind12','Ind13','Ind14','Ind15'}.
compute indces = make(boot+1, 4, 999).

```

```

end if.
do if (model < 4).
compute boot = 0.
compute cmat = make(10,1,0).
compute zmat = make(10,1,0).
end if.
compute nvarch = make(1,ncol(dat),0).
compute wmatch = 0.
compute zmatch = 0.
compute vmatch = 0.
compute qmatch = 0.
compute mmatch=0.
compute minprobe=0.
compute maxprobe=0.
loop i = 1 to ncol(vnames).
do if (vnames(:,i)=yname).
compute y = dat(:,i).
compute nvarch(1,i)=1.
compute yerr = 0.
do if ((yname = xname) or (yname = wname) or
(yname = zname) or (yname = vname) or
(yname = qname)).
compute errs = errs+1.
compute runerrs(errs,1)=17.
end if.
end if.
do if (vnames(:,i)=xname).
compute x = dat(:,i).
compute nvarch(1,i)=1.
compute xdich = 1.
compute xerr=0.
loop jj = 1 to n.
do if ((x(jj,1) <> cmin(x)) and (x(jj,1) <>
cmax(x))).
compute xdich = 0.
break.
end if.
end loop.
compute xmean = csum(x)/n.

```



```

do if (center = 1 and (model < 4 or xlist > 0)
and mcx = 0).
compute meanvec = make(n,1, xmean).
compute x = x-meanvec.
compute centvar = {centvar, xname}.
end if.
compute xmean = csum(x)/n.
compute tmp = x-(cons*xmean).
compute xsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
do if (xdich = 0).
compute quantc(1,6) = 1.
compute matx = {xmean-xsd; xmean;
xmean+xsd}.
do if ( 999 = 999 and quantile = 0).
do if ((matx(1,1) < cmin(x)) and model = 74).
compute matx(1,1)=cmin(x).
compute minprobe=1.
end if.
do if ((matx(3,1) > cmax(x)) and model = 74).
compute matx(3,1)=cmax(x).
compute maxprobe=1.
end if.
end if.
do if (quantile = 1).
compute quantd(1,6) = 1.
compute quantc(1,6) = 0.
compute tmp = x.
compute tmp(GRADE(x)) = x.
compute matx =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (xdich = 1).
compute matx = {cmin(x); cmax(x)}.
do if (model=74).
compute matx=cmin(x).
compute mod74dic=1.
end if.

```

```

end if.
do if ( 999 <> 999).
compute matx = 999.
compute quantd(1,6) = 0.
compute quantc(1,6) = 0.
end if.
end if.
do if (werr = 0 and wlist = 1).
do if (vnames(:,i)=wname).
compute werr = 0.
compute wmatch = 1.
compute w = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(w)/n).
compute w = w-meanvec.
compute centvar = {centvar, wname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute wmean = csum(w)/n.
compute tmp = w-(cons*wmean).
compute wsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute wdich = 1.
loop jj = 1 to n.
do if ((w(jj,1) <> cmin(w)) and (w(jj,1) <>
cmax(w))).
compute wdich = 0.
break.
end if.
end loop.
do if (model = 3).
compute jndich=wdich.
compute jnmin=cmin(w).
compute jnmax=cmax(w).
end if.
do if (wdich = 0).
compute matw = {wmean-wsd; wmean;
wmean+wsd}.
do if ( 999 = 999 and quantile = 0).
do if (matw(1,1) < cmin(w)).

```

```

compute matw(1,1)=cmin(w).
compute minprobe=1.
end if.
do if (matw(3,1) > cmax(w)).
compute matw(3,1)=cmax(w).
compute maxprobe=1.
end if.
end if.
compute quantc(1,1) = 1.
do if (quantile = 1).
compute quantd(1,1) = 1.
compute quantc(1,1) = 0.
compute tmp = w.
compute tmp(GRADE(w)) = w.
compute matw =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (wdich = 1).
compute matw = {cmin(w); cmax(w)}.
compute wvdich=1.
compute cmaxw=cmax(w).
compute cminw=cmin(w).
end if.
do if ( 999 <> 999).
compute matw = 999.
compute quantd(1,1) = 0.
compute quantc(1,1) = 0.
end if.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm(1,1)=wname.
compute modmatp(1,1) = 1.
end if.
end if.
do if (zerr = 0 and zlist = 1).
do if (vnames(:,i)=zname).
compute zerr = 0.

```

```

compute zmatch = 1.
compute z = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(z)/n).
compute z = z-meanvec.
compute centvar = {centvar, zname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute zmean = csum(z)/n.
compute tmp = z-(cons*zmean).
compute zsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute zdich = 1.
loop jj = 1 to n.
do if ((z(jj,1) <> cmin(z)) and (z(jj,1) <>
cmax(z))).
compute zdich = 0.
break.
end if.
end loop.
do if (zdich = 0).
compute matz = {zmean-zsd; zmean;
zmean+zsd}.
do if ( 999 = 999 and quantile = 0).
do if (matz(1,1) < cmin(z)).
compute matz(1,1)=cmin(z).
compute minprobe=1.
end if.
do if (matz(3,1) > cmax(z)).
compute matz(3,1)=cmax(z).
compute maxprobe=1.
end if.
end if.
compute quantc(1,2) = 1.
do if (quantile = 1).
compute quantd(1,2) = 1.
compute quantc(1,2) = 0.
compute tmp = z.
compute tmp(GRADE(z)) = z.

```

```

compute matz = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (zdich = 1).
compute matz = {cmin(z); cmax(z)}.
compute cmaxz=cmax(z).
compute cminz=cmin(z).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matz = 999.
compute quantd(1,2) = 0.
compute quantc(1,2) = 0.
end if.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm(1,2)=zname.
compute modmatp(1,2) = 1.
end if.
end if.
do if (verr = 0 and vlist = 1).
do if (vnames(:,i)=vname).
compute verr = 0.
compute vmatch = 1.
compute v = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(v)/n).
compute v = v-meanvec.
compute centvar = {centvar, vname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute vmean = csum(v)/n.
compute tmp = v-(cons*vmean).
compute vsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute vdich = 1.
loop jj = 1 to n.

```

```

do if ((v(jj,1) <> cmin(v)) and (v(jj,1) <>
cmax(v))).
compute vdich = 0.
break.
end if.
end loop.
do if (vdich = 0).
compute matv = {vmean-vsd; vmean;
vmean+vsd}.
do if ( 999 = 999 and quantile = 0).
do if (matv(1,1) < cmin(v)).
compute matv(1,1)=cmin(v).
compute minprobe=1.
end if.
do if (matv(3,1) > cmax(v)).
compute matv(3,1)=cmax(v).
compute maxprobe=1.
end if.
end if.
compute quantc(1,3) = 1.
do if (quantile = 1).
compute quantd(1,3) = 1.
compute quantc(1,3) = 0.
compute tmp = v.
compute tmp(GRADE(v)) = v.
compute matv = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (vdich = 1).
compute matv = {cmin(v); cmax(v)}.
compute wvdich=1.
compute cmaxv=cmax(v).
compute cminv=cmin(v).
end if.
do if ( 999 <> 999).
compute matv = 999.
compute quantd(1,3) = 0.
compute quantc(1,3) = 0.

```

```

end if.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm(1,3)=vname.
compute modmatp(1,3) = 1.
end if.
end if.
do if (qerr = 0 and qlist = 1).
do if (vnames(:,i)=qname).
compute qerr = 0.
compute qmatch = 1.
compute q = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(q)/n).
compute q = q-meanvec.
compute centvar = {centvar, qname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute qmean = csum(q)/n.
compute tmp = q-(cons*qmean).
compute qsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute qdich = 1.
loop jj = 1 to n.
do if ((q(jj,1) <> cmin(q)) and (q(jj,1) <>
cmax(q))).
compute qdich = 0.
break.
end if.
end loop.
do if (qdich = 0).
compute matq = {qmean-qsd; qmean;
qmean+qsd}.
do if ( 999 = 999 and quantile = 0).
do if (matq(1,1) < cmin(q)).
compute matq(1,1)=cmin(q).
compute minprobe=1.
end if.
do if (matq(3,1) > cmax(q)).
compute matq(3,1)=cmax(q).

```

```

compute maxprobe=1.
end if.
end if.
compute quantc(1,4) = 1.
do if (quantile = 1).
compute quantd(1,4) = 1.
compute quantc(1,4) = 0.
compute tmp = q.
compute tmp(GRADE(q)) = q.
compute matq = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (qdich = 1).
compute matq = {cmin(q); cmax(q)}.
compute cmaxq=cmax(q).
compute cminq=cmin(q).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matq = 999.
compute quantd(1,4) = 0.
compute quantc(1,4) = 0.
end if.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm(1,4)=qname.
compute modmatp(1,4) = 1.
end if.
end if.
do if (vnames(:,i)=cname).
compute cld = dat(:,i).
compute cvname = vnames(:,i).
compute nvarch(1,i)=1.
compute clsmtch = 1.
end if.
loop j = 1 to ncol(mnames).
do if (vnames(:,i)=mnames(1,j)).
compute mmatch = mmatch + 1.

```



```

compute m(:,j)=dat(:,i).
do if (center = 1 and nvarch(1,i) = 0 and mlist >
0 and mcm = 0).
compute meanvec = make(n,1,csum(m(:,j))/n).
compute m(:,j) = m(:,j)-meanvec.
compute centvar = {centvar, mnames(1,j)}.
end if.
compute nvarch(1,i)=1.
compute dichm = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute dichm = 0.
break.
end if.
end loop.
do if ((dichm = 1) and (olsdichm = 1)).
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichm = 1 and model > 3 and mcheck = 0
and oldsdichm = 0).
compute errs= errs+1.
compute runerrs(errs,1) = 1.
compute mcheck = 1.
end if.
do if ((model <= 3) and (ncol(mnames)=1)).
compute nmods = nmods + 1.
compute mmean = csum(m(:,j))/n.
compute tmp = m(:,j)-(cons*mmean).
compute msd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute mdich = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute mdich = 0.
break.
end if.
end loop.
do if (model = 1).

```

```

compute jndich=mdich.
compute jnmin=cmin(m(:,j)).
compute jnmax=cmax(m(:,j)).
end if.
do if (mdich = 0).
compute matm = {mmean-msd; mmean;
mmean+msd}.
do if ( 999 = 999 and quantile = 0).
do if (matm(1,1) < cmin(m(:,j))).
compute matm(1,1)=cmin(m(:,j)).
compute minprobe=1.
end if.
do if (matm(3,1) > cmax(m(:,j))).
compute matm(3,1)=cmax(m(:,j)).
compute maxprobe=1.
end if.
end if.
compute quantc(1,5) = 1.
do if (quantile = 1).
compute quantd(1,5) = 1.
compute quantc(1,5) = 0.
compute tmp = m(:,j).
compute tmp(GRADE(m(:,j))) = m(:,j).
compute matm = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (mdich = 1).
compute matm = {cmin(m); cmax(m)}.
end if.
do if ( 999 <> 999).
compute matm = 999.
compute quantd(1,5) = 0.
compute quantc(1,5) = 0.
end if.
compute modmatv(1,5)=nrow(matm).
compute modmat((1:nrow(matm)), 5) = matm.
compute modvnm(1,5)=mnames(1,j).
compute modmatp(1,5) = 1.

```

```

end if.
end if.
end loop.
end loop.
do if (minprobe=1).
compute note(notes,1) = 14.
compute notes = notes + 1.
end if.
do if (maxprobe=1).
compute note(notes,1) = 15.
compute notes = notes + 1.
end if.
do if (cname <> 'xxx' and clsmatch = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 23.
end if.
do if (cname <> 'xxx').
do if ((cname = zname) or (cname = vname)
or (cname = wname) or (cname = xname) or
(cname = yname) or (cname = qname)).
compute errs = errs+1.
compute runerrs(errs,1) = 24.
end if.
end if.
do if (wlist = 1 and werr = 0 and wmatch = 0).
compute werr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (zlist = 1 and zerr = 0 and zmatch = 0).
compute zerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (qlist = 1 and qerr = 0 and qmatch = 0).
compute qerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (vlist = 1 and verr = 0 and vmatch = 0).

```

```

compute verr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (yerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 16.
end if.
do if (xerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 32.
end if.
do if (model = 6 and nmeds < 2).
compute errs = errs+1.
compute runerrs(errs,1) = 18.
end if.
do if (mmatch < ncol(mnames)).
compute errs=errs+1.
compute runerrs(errs,1) = 25.
end if.
do if (cname <> 'xxx').
compute cld = design(cld).
compute cluster = ncol(cld).
compute cld = cld(:,2:ncol(cld)).
compute clsdmy = ncol(cld).
do if (clsdmy > 19).
compute errs = errs+1.
compute runerrs(errs,1) = 26.
end if.
do if (mcx > 0 or mcm > 0).
compute errs = errs+1.
compute runerrs(errs,1) = 38.
end if.
end if.
compute dichy = 1.
loop jj = 1 to n.
do if ((y(jj,1) <> cmin(y)) and (y(jj,1) <>
cmax(y))).
compute dichy = 0.
break.

```

```

end if.
end loop.
do if ((dichy = 1) and (olsdichy = 1)).
compute dichy=0.
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichy = 1).
do if (toteff=1).
compute toteff=0.
compute note(notes,1) = 31.
compute notes = notes + 1.
end if.
compute jncrit=xp2*xp2.
do if (ws = 1).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 28.
end if.
end if.
compute ncovs = ncol(dat)-rsum(nvarch).
do if (effsize = 1 and covmy <> 0 and ncovs >
0 and model > 3 and model < 7).
compute note(notes,1) = 22.
compute notes = notes + 1.
compute effsize = 0.
end if.
do if (ws = 1 and ncovs < nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 29.
end if.
do if (ws = 1 and ncovs > nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 30.
end if.
end if.
do if (errs = 0).
do if (rsum(quantd) > 0).

```

```

compute note(notes,1) = 4.
compute notes = notes+1.
end if.
do if (rsum(quantc) > 0).
compute note(notes,1) = 5.
compute notes = notes+1.
end if.
do if (ncovs > 0).
compute c = make(n,ncovs,0).
compute cnames = {'x'}.
compute j = 1.
loop i = 1 to ncol(vnames).
do if (nvarch(1,i)) = 0.
compute c(:,j) = dat(:,i).
compute nvarch(1,i)=1.
compute j=j+1.
compute cnames = {cnames, vnames(:,i)}.
end if.
end loop.
compute cnames = cnames(1,2:ncol(cnames)).
do if (ws = 1).
compute covmean=csum(c)/n.
loop i = 1 to ncovs.
compute meanvec = make(n,1, covmean(1,i)).
compute c(:,i) = c(:,i)-meanvec.
end loop.
compute centvar = {centvar, cnames}.
compute x=c(:,1:ncovs).
do if (ncovs=nmeds).
compute ncovs=0.
end if.
end if.
end if.
compute names = {yname, xname, mnames,
wname, zname, vname, qname}.
do if (ncovs > 0).
compute names = {names, cnames}.
end if.
do if (dichy = 1 and effsize = 1).
compute note(notes,1) = 2.

```

```

compute notes = notes+1.
end if.
do if (model > 3 and model < 6).
compute indeff=make(nmeds,1,0).
compute indboot=make(boot+1,nmeds,999).
do if (mc > 0).
compute indboot=make(mc+1,nmeds,999).
end if.
do if (effsize = 1 and dichy = 0).
compute rmeff=make(boot+1,nmeds+1,999).
compute abpseff=make(boot+1,nmeds+1,999).
compute abcseff=make(boot+1,nmeds+1,999).
compute pmeff=make(boot+1,nmeds+1,999).
compute r245 = make(boot+1,1,999).
compute kappa2 = make(boot+1,1,999).
end if.
end if.
do if (model = 6 and effsize = 1 and dichy = 0).
compute
rmeff=make(boot+1,ncol(indboot),999).
compute
abpseff=make(boot+1,ncol(indboot),999).
compute
abcseff=make(boot+1,ncol(indboot),999).
compute
pmeff=make(boot+1,ncol(indboot),999).
end if.
do if (nmods > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm(1,tmp) = modvnm(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:nmods).
compute modvnm=modvnm(:,1:nmods).
compute modmatv=modmatv(:,1:nmods).

```

```

loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvals =
make((modmatv(1,1)*modprod(1,1)),
nmods,0).
loop i = 1 to nmods.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvals)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvals(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
do if (model = 74).
compute modvals=matx.
compute modvnm=xname.
end if.
compute vmat = make(8,nrow(modvals),0).
compute vmat(1,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute vmat(5,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute indeff=make(nrow(modvals),1,0).
do if (model <> 5).
compute indboot = make(((boot+1)*nmeds),
nrow(modvals), -99999999).
compute indbootp =
make(boot+1,(nmeds*nmods), -99999999).
do if (imm > 4 and imm < 8).
do if (imm = 7).

```



```

compute immmv1=matw.
compute immmv2=matv.
end if.
do if (imm = 5).
compute immmv1=matw.
compute immmv2=matz.
end if.
do if (imm = 6).
compute immmv1=matv.
compute immmv2=matq.
end if.
compute
indboop1=make((boot+1),(nmeds*(nrow(immmv2))),-9999999).
compute
indboop2=make((boot+1),(nmeds*(nrow(immmv1))),-9999999).
compute onesmv1=make(1,nrow(immmv1),1).
compute onesmv2=make(1,nrow(immmv2),1).
compute
nimvs=nrow(immmv1)+nrow(immmv2).
compute cmmval=0.
compute cmmbs1={'xx'}.
compute cmmbs2={'xx'}.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv2}.
compute
cmmbs1=make(nrow(immmv2),1,mnames(1,i)).
compute cmmbs1={cmmbs1;cmmbs1}.
end loop.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv1}.
compute
cmmbs2=make(nrow(immmv1),1,mnames(1,i)).
compute cmmbs2={cmmbs2;cmmbs2}.
end loop.
compute cmmval=cmmval(2:nrow(cmmval),:).
end if.
end if.
end if.

```

```

do if (nmods> 0).
loop i = 1 to ncol(modvals).
do if (modvnm(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm(1,i)=qname).
compute qcol = i.
end if.
end loop.
end if.
do if (dichy = 1).
compute omx = cmax(y).
compute omn = cmin(y).
compute y = (y = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute data = {cons,y,m,x}.
compute datamed = data.
compute datayed = data.
compute datanm = {'constant'; yname;
t(mnames); xname}.
compute datanmm = {'constant'; yname;
t(mnames); xname}.
compute datanmy = {'constant'; yname;
t(mnames); xname}.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
do if (model < 4 and errs = 0).
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' ', ' '}.
compute datayed = {datayed, x&*m}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).

```

```

compute vmat(1,i) = 1.
compute vmat(2,i) = modvals(i,1).
end loop.
compute mmat = make(16,nmeds,1).
end if.
do if (model = 2 or model = 3).
compute int1 = x&*w.
compute datayed = {datayed, w, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,2).
compute vmat(3,i) = modvals(i,1).
compute vmat(4,i) = modvals(i,1)*modvals(i,2).
end loop.
end if.
do if (model = 3).
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*m.
compute int2 = x&*w&*m.
compute datayed = {datayed, int1, int2}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' X', wname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (model = 4 or model = 5).
compute vmat = make(8,1,1).
end if.
compute yintkey2 = yintkey.
do if (wm = 1).
compute int1 = x&*w.

```

```

compute datamed = {datamed, w,int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmm = {datanmm; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,wcol).
end loop.
do if (zm = 1).
compute int1 = x&*z.
compute datamed = {datamed, z, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(3,i) = modvals(i,zcol).
end loop.
end if.
do if (wzm = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*z.
compute int2 = x&*w&*z.
compute datamed = {datamed, int1, int2}.
loop i = 1 to nrow(modvals).
compute vmat(4,i) =
modvals(i,wcol)*modvals(i,zcol).
end loop.
end if.

```

```

end if.
compute mdatacol=ncol(datamed).
compute mintkey = yintkey.
compute yintkey = {' ',' ',' ',' ',' ',' '}.
compute medints = intcnt-1.
do if (vy = 1 or xmy = 1).
compute mp = 1.
do if (xmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,1).
end loop.
end if.
do if (vy = 1).
compute datayed = {datayed, v}.
compute datanmy = {datanmy; vname}.
compute mmods = 1.
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,vcol).
end loop.
do if (qy = 1).
compute mp = 2.
compute datayed = {datayed,q}.
compute datanmy = {datanmy; qname}.
compute mmods = 2.
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,qcol).
end loop.
end if.
do if (vqy = 1).
compute mp = 3.
compute datayed = {datayed, v&*q}.
compute mmods = 3.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,vcol)*modvals(i,qcol).
end loop.
end if.
end if.
compute mints = make(n,(nmeds*mp),0).
loop i = 0 to (nmeds-1).

```

```

do if (i = 0 and vqy = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
vname, ' X', qname, '', ''}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, '', ''}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (xmy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))&*x.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', xname, '', ''}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (qy = 1).
compute mints(:,((i*mp)+2))=m(:,(i+1))&*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', qname, '', ''}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqy = 1).
compute mints(:,((i*mp)+3))=m(:,(i+1))&*v&*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end loop.

```

```

compute datayed = {datayed, mints}.
end if.
compute mp = 1.
do if (wvmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,wcol)*modvals(i,vcol).
end loop.
end if.
compute mints2 = make(n,(nmeds*mp),0).
do if (wmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,wcol).
end loop.
do if (wy = 0 and model > 3).
compute datayed = {datayed, w}.
compute datanmy = {datanmy; wname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wvmy = 1).
compute datayed = {datayed,w&*v}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints2(:,((i*mp)+1))= m(:,(i+1))&*w.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wvmy = 1).
compute mints2(:,((i*mp)+2))=
m(:,(i+1))&*w&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', vname}.

```

```

compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints2}.
end if.
compute mp = 1.
do if (zmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,zcol).
end loop.
do if (wzmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,zcol)&*modvals(i,wcol).
end loop.
end if.
end if.
do if (zmy = 1).
compute mints3 = make(n,(nmeds*mp),0).
do if (zy = 0).
compute datayed = {datayed, z}.
compute datanmy = {datanmy; zname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wzmy = 1 and wzy = 0).
compute datayed = {datayed,w&*z}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints3(:,((i*mp)+1))= m(:,(i+1))&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.

```



```

compute intcnt = intcnt+1.
do if (wzmy = 1).
compute mints3(:,(i*mp)+2))=
m(:,(i+1))*w*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', zname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints3}.
end if.
compute decoc = 1.
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
do if (wy = 1 and model > 3).
compute datayed = {datayed, w, x*w}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm2(1,1)=wname.
compute modmatp(1,1) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (zy = 1).
compute datayed = {datayed,z,x*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm2(1,2)=zname.
compute modmatp(1,2) = 1.

```

```

compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ',', ' '}.
compute datanmy = {datanmy; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (wzy = 1).
compute datayed = {datayed,w&*z,x&*w&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ',', ' '}.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vxy = 1).
compute datayed = {datayed, x&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm2(1,3)=vname.
compute modmatp(1,3) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, ',', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (qxy = 1).
compute datayed = {datayed, x&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.

```

```

compute modvnm2(1,4)=qname.
compute modmatp(1,4) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqxy = 1).
compute datayed = {datayed, x&*v&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, 'X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end if.
do if (wvxy = 1).
compute datayed = {datayed, x&*w&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, 'X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute modvalsd = 0.
compute ttt = rsum(modmatp).
do if (rsum(modmatp) > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm2(1,tmp) = modvnm2(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.

```

```

end loop.
compute modmat=modmat(:,1:ttt).
compute modvnm2=modvnm2(:,1:ttt).
compute modmatv=modmatv(:,1:ttt).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvalsd =
make((modmatv(1,1)*modprod(1,1)), ttt,0).
loop i = 1 to ttt.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvalsd)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvalsd(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
end if.
do if (ttt > 0).
loop i = 1 to ncol(modvalsd).
do if (modvnm2(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm2(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm2(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm2(1,i)=qname).
compute qcol = i.

```

```

end if.
end loop.
compute directv = make(nrow(modvalsd),1,1).
do if (wy = 1).
compute directv ={directv, modvalsd(:,wcol)}.
end if.
do if (zy = 1).
compute directv ={directv, modvalsd(:,zcol)}.
end if.
do if (wzy = 1).
compute directv ={directv,
(modvalsd(:,wcol)&*modvalsd(:,zcol))}.
end if.
do if (vxy = 1).
compute directv ={directv, modvalsd(:,vcol)}.
end if.
do if (qxy = 1).
compute directv ={directv, modvalsd(:,qcol)}.
end if.
do if (vqxy = 1).
compute directv ={directv,
(modvalsd(:,vcol)&*modvalsd(:,qcol))}.
end if.
do if (wvxy = 1).
compute directv ={directv,
(modvalsd(:,vcol)&*modvalsd(:,wcol))}.
end if.
end if.
compute ydatacol=ncol(datayed).
do if (ncovs > 0).
do if (covmy <> 2).
compute datamed = {datamed,c}.
end if.
do if (covmy <> 1).
compute datayed = {datayed,c}.
end if.
do if (covmy = 1 and model = 4 and mcx > 0).
compute datayed = {datayed,c}.
end if.
compute covmeans = csum(c)/n.

```

```

end if.
do if (cluster > 0).
compute datamed = {datamed,cld}.
compute datayed = {datayed, cld}.
compute cldmeans = csum(cld)/n.
end if.
compute mst = 3.
compute mnd = mst+nmeds-1.
compute ydatacol=ncol(datayed).
compute mdatacol=ncol(datamed).
do if (ncovs > 0).
compute datanmy = {datanmy; t(cnames)}.
do if (model > 3).
compute datanmm = {datanmm; t(cnames)}.
end if.
end if.
compute datanmy = {'constant';
datanmy(3:nrow(datanmy),1)}.
do if (model > 3).
compute datanmm = {'constant';
datanmm(3:nrow(datanmm),1)}.
end if.
compute amm = make(2,1,0).
compute abmm = make(2,1,0).
compute mnv = csum(datayed(:,2)/n).
compute mnv = make(n,1,mnv).
compute ssty = csum((datayed(:,2)-mnv)**2).
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
do if (ws=1).
compute stddevm=diag(sigma).
compute
stddevm=sqrt(stddevm(3:(nmeds+2),1)).
end if.
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute r2xy =
(sigma(2,(3+nmeds)))/(stddevy*stddevx)**2.

```

```

compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))**2.
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).
do if (model = 4 and nmeds = 1 and cluster = 0
and ncovs = 0).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
compute datatm = datamed.
compute dataty = datayed.
compute mdlrms2 = { '1' ; yname; xname}.
compute mdlrms = {'Model ='; ' Y ='; ' X ='
}.
do if (ws = 1).
compute mdlrms2 = { '1' ; yname}.
compute mdlrms = {'Model ='; ' Y ='}.
end if.

```

```

loop i = 1 to ncol(mnames).
compute mdlInms2 = {mdlInms2; mnames(1,i)}.
do if (i = 1 and ncol(mnames) = 1).
compute mdlInms = {mdlInms; '  M = '}.
else.
compute mdlInms = {mdlInms; mlab(i,1)}.
end if.
end loop.
do if (wname <> 'xxx').
compute mdlInms2 = {mdlInms2; wname}.
compute mdlInms = {mdlInms; '  W = '}.
end if.
do if (zname <> 'xxx').
compute mdlInms2 = {mdlInms2; zname}.
compute mdlInms = {mdlInms; '  Z = '}.
end if.
do if (vname <> 'xxx').
compute mdlInms2 = {mdlInms2; vname}.
compute mdlInms = {mdlInms; '  V = '}.
end if.
do if (qname <> 'xxx').
compute mdlInms2 = {mdlInms2; qname}.
compute mdlInms = {mdlInms; '  Q = '}.
end if.
do if (jn = 1 and model = 1 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
do if (jn = 1 and model = 3 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
compute mdpbe=0.
compute medte2=0.
do if ( 1=4 and mcx > 0).
compute medte2=1.
end if.
do if (((mcx > 0) or (mcm > 0)) AND model =
1).
compute mdpbe=1.

```



```

compute dd={datayed(:,4),datayed(:,3)}.
do if (ncovs > 0).
compute dd={datayed(:,6:(5+ncovs)),dd}.
end if.
compute dd={datayed(:,2),dd}.
end if.
compute yes = ((nrow(modvals)=1) and
(contrast = 1) and (model = 3 or model = 2)).
compute yes2 = ((nrow(modvals)=1) and
(contrast = 1) and (model = 2)).
print/title = '***** PROCESS Procedure
for SPSS Release 2.16.3 *****'.
print/title = '      Written by Andrew F. Hayes,
Ph.D.    www.afhayes.com'.
print/title = '  Documentation available in
Hayes (2013).
www.guilford.com/p/hayes3/space=0.
print mdlnms2/title =
*****
*****'/rnames = mdlnms/format =
a8.
do if (ncovs > 0).
print cnames/title = 'Statistical Controls:'/rlabels
= 'CONTROL='/format a8.
end if.
print n/title = 'Sample size'/format F10.0.
do if (cluster > 0).
print cluster/rnames = cvname/title =
'Clustering Variable and Number of Clusters'.
end if.
do if (model > 3 and ( 'random' <> 'random')).
compute seedt= 'random'.
print seedt/title='Custom'/format=A12/rlabel =
'Seed:'.
end if.
do if (mdpbe=0 and medte2 = 0).
loop bt = 1 to (boot+1).
do if (bt = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)),-999).

```

```

end if.
compute bootcoef = 0.
do if (bt > 1).
loop.
compute v=trunc(uniform(n,1)*n)+1.
compute datayed = dataty(v,:).
do if (ws=1).
compute
bmeans=csum(datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed)))/n.
compute cntmat=make(n,nmeds,1).
compute bmeans2=bmeans.
do if (nmeds > 1).
compute bmeans2=mdia(bmeans).
end if.
compute cntmat=cntmat*bmeans2.
compute datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed))=datayed(:,(ncol(data
yed)-nmeds+1):ncol(datayed))-cntmat.
end if.
compute detcheck=det(t(datayed)*datayed).
compute rk=(detcheck <> 0).
do if (model > 3).
compute datamed = datatm(v,:).
compute detcheck=det(t(datamed)*datamed).
do if (rk=1).
compute rk=(detcheck <> 0).
end if.
end if.
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
compute temp=diag(sigma).
compute bad = bad+(1-rk).
compute false = 1.
end loop if (rk = 1).
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).

```

```

compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).
do if (model = 4 and nmeds = 1 and ncovs = 0
and cluster = 0).
compute r2xy =
(sigma(2,4)/(stddevy*stddevx))&**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))&**2.
compute sstot = sigma(2,2)*(n-1).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
end if.
do if (model > 3).
loop im = 1 to nmeds.
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.

```

```

compute xnm = {'constant';
datanmm((2+nmeds):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
do if (model = 6).
do if (im = 1).
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
end if.
do if (im > 1).
compute xm={cons, datamed(:,3:(im+1)),
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant'; datanmm(2:im,1);
datanmm((mnd):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
compute
mmpaths((im+1),(2:im))=t(coeff(2:im,1)).
end if.
end if.
do if (ws = 1).
compute
coeff(1,1)=csum(datamed(:,(2+im)))/n.
end if.
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
do if (bt = 1).
compute resid=datamed(:,(2+im))-xm*coeff.
compute sse =cssq(resid).
compute mse = sse/(n-ncol(xm)).
compute mnv = csum(data(:,(2+im)))/n.
compute mnv = make(n,1,mnv).
compute sstm = csum((data(:,(2+im))-
mnv)**2).

```

```

compute k3 = nrow(coeff).
do if (hc3 = 1).
compute h = xm(:,1).
loop i3=1 to n.
compute h(i3,1)= xm(i3,:)*invXtX*t(xm(i3,:)).
end loop.
loop i3=1 to k3.
compute xm(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xm(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xm(:,i3) = sqrt(mse)&*xm(:,i3).
end loop.
end if.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute hccov=invXtX*t(xm)*xm*invXtX.
compute mcmats(im,im)=hccov(2,2).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*hccov*lmat)*((t(lmat)
)*coeff))/dfnum.
compute coeff = coeff(1:(nrow(coeff)-
clsdmy),1).
compute mccoef(im,1)=coeff(2,1).
compute standerr =
sqrt(diag(invXtX*t(xm)*xm*invXtX)).
compute standerr = standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1).
compute standerr(1,1)=stddevm(im,1)/sqrt(n).
end if.
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-
ncol(xm)))).
compute temp=(n-ncol(xm)).
do if (ws=1).

```

```

compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute temp=(n-1).
end if.
compute xd = abs(xp2).
compute temp = (temp* (exp((temp-
(5/6))*((xd/(temp-(2/3)+(.11/temp)))*(xd/(temp-
(2/3)+(.11/temp)))))-1)).
compute temp1 = coeff-
sqrt(abs(temp))*standerr.
compute temp2 =
coeff+sqrt(abs(temp))*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (ws=1).
compute op=op(1,:).
compute xnmn =
apathnam(((nmeds>1)+1+(im-1)),1).
end if.
compute sobel(im,1) = coeff(2,1).
compute sobel(im,2)=standerr(2,1).
compute temp = mnames(1,im).
compute r2full = 1-(sse/sstm).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute summ = {sqrt(r2full), r2full, mse,
Fratio, dfnum, dfden, pfr}.
do if (detail = 1).
print temp/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.
do if (ws <> 1).
print summ/title = 'Model Summary'/'clabels 'R',
'R-sq', 'MSE','F', 'df1', 'df2', 'p'/'format = F10.4.
end if.
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
print op/title = 'Model'/'rnames = xnmn/'clabels
'coeff', 'se', 't', 'p', 'LLCI', 'ULCI'/'format = F10.4.
do if (covcoeff=1 and ws <> 1).

```

```

compute
hccovtmp=hccov(1:nrow(op),1:nrow(op)).
compute cnamestp=t(xmnm).
print hccovtmp/title='Covariance matrix of
regression parameter
estimates'/rnames=xmnm/cnames=cnamestp/f
ormat= F10.4.
end if.
do if (nmods > 0 and nrow(mintkey) > 1).
print mintkey/title = 'Product terms key:.'/format
= A8.
end if.
do if (ws = 1).
compute wsdf=n-1.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
end if.
end if.
compute ymat(1,im) = coeff(2,1).
do if (ws = 1).
compute ymat(1,im)=coeff(1,1).
end if.
do if (wm = 1).
compute ymat(2,im) = coeff(4,1).
do if (zm = 1).
compute ymat(3,im) = coeff(6,1).
do if (wzm = 1).
compute ymat(4, im) = coeff(8,1).
end if.
end if.
end if.
do if (model = 6).
compute mmpaths((im+1),1)=coeff((im+1),1).
end if.
end loop.
end if.
loop totlp = 1 to (1+(toteff*(bt = 1))).
do if (toteff = 1 and totlp =2).
compute xy =
{cons,datayed(:,(3+nmeds):ydatacol)}.

```

```

else.
compute xy={cons, datayed(:,3:ydatacol)}.
end if.
do if (dichy = 1).
compute pt2 =
make(nrow(datayed(:,2)),1,(csum(datayed(:,2))
/n)).
compute LL3 = datayed(:,2)*ln(pt2)+(1-
datayed(:,2))*ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xy),1,0).
compute LL1 = 0.
compute xy22=xy.
loop jjj = 1 to iterate.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.
compute coeff =
bt1+inv(t(xy22)*xy)*t(xy)*(datayed(:,2)-pt1).
compute pt1 = 1/(1+exp(-(xy*coeff))).
compute itprob = csum((pt1 <
.00000000000001) or (pt1 >
.99999999999999)).
do if (itprob = 0).
compute LL = datayed(:,2)*ln(pt1)+(1-
datayed(:,2))*ln(1-pt1).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
break.
end if.
compute bt1 = coeff.
compute LL1 = LL2.
end loop.
do if (jjj >= iterate and iterr = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
computer iterr = 1.
end if.

```



```

loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)&*pt1&*(1-pt1).
end loop.
compute covmat = inv(t(xy22)*xy).
release xy22.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
end if.
do if (dichy = 0).
compute invXtX = inv(t(xy)*xy).
compute coeff =invXtX*t(xy)*datayed(:,2).
do if (effsize=1 and model > 3 and model < 7
and (ncovs > 0 or clsdmy > 0)).
compute xysdy =
{cons,datayed(:,(4+nmeds):ydatacol)}.
compute coeffsd
=inv(t(xysdy)*xysdy)*t(xysdy)*datayed(:,2).
compute resid=datayed(:,2)-(xysdy*coeffsd).
compute sdy cov =sqrt(csum(resid&*resid)/(n-
ncol(xysdy))).
compute xvaron=datayed(:,(3+nmeds)).
compute
coeffx2=inv(t(xysdy)*xysdy)*t(xysdy)*xvaron.
compute residx2=xvaron-(xysdy*coeffx2).
compute sdx cov
=sqrt(csum(residx2&*residx2)/(n-ncol(xysdy))).
end if.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4 and bt > 1).
compute resid=datayed(:,2)-xy*coeff.
compute sse = cssq(resid).
compute r2full = 1-(sse/sstot).

```

```

end if.
do if (bt = 1).
compute resid=data(:,2)-xy*coeff.
compute k3 = nrow(coeff).
compute sse = cssq(resid).
compute mse = sse/(n-ncol(xy)).
compute
hcnott=coeff&/(sqrt(diag(mse*invXtX))).
do if (hc3 = 1).
compute h = xy(:,1).
loop i3=1 to n.
compute h(i3,1)= xy(i3,:)*invXtX*t(xy(i3,:)).
end loop.
loop i3=1 to k3.
compute xy(:,i3) = (resid(:,ncol(resid))&/(1-
h))*xy(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xy(:,i3) = sqrt(mse)*xy(:,i3).
end loop.
end if.
compute covmat = (invXtX*t(xy)*xy*invXtX).
end if.
end if.
do if (bt = 1).
do if (model = 2).
compute xy2={cons, datayed(:,3:ydatacol)}.
compute temp = ncol(xy2).
do if (temp > 6).
compute xy3=xy2(:,7:temp)}.
end if.
compute xy2={xy2(:,1:3), xy2(:,5)}.
do if (temp > 6).
compute xy2={xy2, xy3}.
release xy3.
end if.
compute invXtX = inv(t(xy2)*xy2).
compute coeff2 =invXtX*t(xy2)*datayed(:,2).

```

```

compute ssem2=cssq(datayed(:,2)-xy2*coeff2).
release xy2.
end if.
compute standerr = sqrt(diag(covmat)).
do if (totlp =1).
compute
mcmats((nmeds+1):ncol(mcmats),(nmeds+1):n
col(mcmats))=covmat(2:(1+nmeds),2:(1+nmed
s)).
end if.
compute standerr=standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1 and totp=2).
compute standerr(1,1)=stddevy/sqrt(n).
end if.
compute coeffplt = coeff.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*coeff)))/dfnum).
compute coeff=coeff(1:(nrow(coeff)-clsdmy),1).
do if (totlp=1).
compute
mccoeff((nmeds+1):nrow(mccoeff))=coeff(2:(1+
nmeds),1).
end if.
compute bbbb=coeff(2,1).
do if totp = 1.
compute deco(1,1)=2+nmeds.
compute deco = deco(1:decoc,1).
compute covdirt = make((nrow(covmat)-
clsdmy),(ncol(covmat)-clsdmy),0).
compute covdirt = covmat(deco,:).
compute covdir =
make(nrow(covdirt),ncol(covdirt),0).
compute covdir = covdirt(:,t(deco)).
compute deco=coeff(deco,1).

```

```

do if (ttt > 0).
compute sedir =
sqrt(diag(directv*covdir*t(directv))).
compute directv = directv*deco.
end if.
compute sobel(:,3)=coeff(2:(1+nmeds),1).
compute sobel(:,4) = standerr(2:(1+nmeds),1).
compute sobel2 = sobel&*sobel.
do if (varorder <> 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)&*sobel2(:,4)+sobel2(:,3)&*sob
el2(:,2)).
end if.
do if (varorder = 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)&*sobel2(:,4)+sobel2(:,3)&*sob
el2(:,2)+sobel2(:,2)&*sobel2(:,4)).
end if.
compute sobel(:,1) = sobel(:,1)&*sobel(:,3).
compute sobel(:,3) = sobel(:,1)&/sobel(:,2).
compute sobel(:,4) = 2*(1-
cdfnorm(abs(sobel(:,3)))).
end if.
do if (dichy = 0).
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
do if (ws= 1 and totlp=2).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute dfden=n-1.
end if.
compute cnms = {'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute op = {coeff, standerr, tratio, p}.
end if.
do if (dichy = 1).
compute tratio = (coeff&/standerr).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute wald = tratio&*tratio.
compute cnms = {'coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.

```

```

compute temp=coeff-abs(xp2)*standerr.
compute op = {coeff, standerr, tratio, p, temp}.
compute temp=coeff+abs(xp2)*standerr.
compute op = {op, temp}.
end if.
do if (detail = 1).
do if (totlp = 2).
print yname/title = '*****'
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:.'/format = A8.
end if.
do if (totlp <> 2).
print yname/title =
*****
*****'/rlabels = 'Outcome:.'/format
= A8.
end if.
end if.
do if (dichy = 1 and bt = 1 and totp = 1).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:.'/cnames = nmsd/format = F9.2.
end if.
do if (dichy = 0).
compute r2full = 1-(sse/ssty).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute jndf=dfden.
compute wsdf=dfden.
compute xd = abs(xp2).
compute jncrit = (dfden* (exp((dfden-
(5/6))*((xd/(dfden-
(2/3)+(.11/dfden)))*(xd/(dfden-
(2/3)+(.11/dfden)))))-1)).
compute summ = {sqrt(r2full), r2full, mse,
fratio, dfnum, dfden, pfr}.
compute temp1=coeff-sqrt(jncrit)*standerr.
compute temp2=coeff+sqrt(jncrit)*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.

```

```

do if (detail = 1).
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels =
'R', 'R-sq', 'MSE' 'F', 'df1', 'df2', 'p'/format =
F10.4.
end if.
end if.
end if.
do if (dichy = 1).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(coeff)-
1)).
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
do if (detail = 1).
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format =
F10.4.
end if.
end if.
do if (totlp = 2).
compute datanmy={'constant';
datanmy((nmeds+2):nrow(datanmy),1)}.
end if.
do if (detail = 1).
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
do if (ws=1).
compute op2=op(1:(1+nmeds),:).
compute datanmy2 = {'c" path'; bpathnam(
(1+(nmeds>1)): ((1+(nmeds>1))+(nmeds-
1)),1)}.
do if (totlp=2).
compute datanmy2={'c path'}.
compute op2=op(1,:).

```

```

compute wsdf=n-1.
end if.
print op2/title = 'Model'/rnames =
datanmy2/cnames = cnms/format = F10.4.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
do if (ws <> 1).
print op/title = 'Model'/rnames =
datanmy/cnames = cnms/format = F10.4.
do if (covcoeff=1).
compute
covmattp=covmat(1:nrow(op),1:nrow(op)).
compute cnamestp=t(datanmy).
print covmattp/title='Covariance matrix of
regression parameter
estimates'/rnames=datanmy/cnames=cnamest
p/format= F10.4.
end if.
end if.
end if.
do if (ttt = 0 and totlp = 1).
compute deco = op((nmeds+2),:).
do if (ws = 1).
compute deco=op(1,:).
end if.
end if.
do if (ttt = 0 and totlp = 2).
compute decotot = op(2,:).
do if (ws = 1).
compute decotot=op(1,:).
end if.
end if.
do if (nmods > 0 and model > 4 and detail = 1
and nrow(yintkey) > 1)).
print yintkey/title = 'Product terms key: '/format
= A8.
end if.
do if (nmods > 0 and model < 4 and detail = 1).
print yintkey2/title = 'Product terms key: '/format
= A8.

```

```

do if ((model = 1 or model = 2) and dichy = 0).
compute temp={((op(4,3)**2)*(1-r2full))/dfden,
op(4,3)**2, 1, dfden, op(4,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(4,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(2,1).
do if (model = 2).
compute temp={temp;((op(6,3)**2)*(1-
r2full))/dfden, op(6,3)**2, 1, dfden, op(6,4)}.
do if (hc3=1).
compute temp(2,1)=((hcnot(6,1)**2)*(1-
r2full))/dfden.
end if.
compute frat2=(dfden*(r2full-(1-
(ssem2/ssty))))/(2*(1-r2full)).
do if (hc3=1).
compute lmat=make(nrow(coeffplt),2,0).
compute lmat(4,1)=1.
compute lmat(6,2)=1.
compute frat2 =
(t(t(lmat)*coeffplt)*inv(t(lmat)*covmat*lmat)*((t(l
mat)*coeffplt))/2).
end if.
compute temp={temp;(r2full-(1-
(ssem2/ssty))),frat2,2,dfden,1-
fcdf(frat2,2,dfden)}.
compute rnms={rnms; yintkey2(3,1),'Both'}.
end if.
print temp/title = 'R-square increase due to
interaction(s):'/rnames=rnms/clabels = 'R2-
chng', 'F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (model = 3 and dichy = 0).
compute temp={((op(8,3)**2)*(1-r2full))/dfden,
op(8,3)**2, dfden, op(8,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(8,1)**2)*(1-
r2full))/dfden.

```



```

end if.
compute rnms=yintkey2(5,1).
print temp/title = 'R-square increase due to
three-way interaction:.'/rnames=rnms/clabels =
'R2-chng', 'F(1,df2)', 'df2', 'p'/format = F10.4.
end if.
end if.
end if.
do if (model = 6 and totlp = 1).
compute
mmpaths(nrow(mmpaths),1)=coeff(nrow(mmpa
ths),1).
compute
mmpaths(nrow(mmpaths),(2:(nmeds+1)))=t(co
eff(2:(nmeds+1),1)).
end if.
do if (totlp = 1).
loop im = 1 to nmeds.
do if (model < 4).
compute ymat(1,im) = coeff(3,1)*(1-yes).
compute ymat(2,im) = coeff(4,1).
compute cmat(1,im) = covmat(3,3).
compute cmat(2,im) = covmat(4,4).
compute cmat(5,im) = covmat(3,4).
compute jnb1=coeff(3,1).
compute jnb3=coeff(4,1).
compute jnsb1=covmat(3,3).
compute jnsb3=covmat(4,4).
compute jnsb1b3=covmat(3,4).
do if (model = 2 or model = 3).
compute ymat(3,im) = coeff(6,1).
compute cmat(3,im) = covmat(6,6).
compute cmat(6,im) = covmat(3,6).
compute cmat(8,im) = covmat(4,6).
end if.
do if (model = 3).
compute ymat(4,im) = coeff(8,1).
compute cmat(4,im) = covmat(8,8).
compute cmat(7,im) = covmat(3,8).
compute cmat(9,im) = covmat(4,8).

```

```

compute cmat(10,im) = covmat(6,8).
compute jnb1=coeff(4,1).
compute jnb3=coeff(8,1).
compute jnsb1=covmat(4,4).
compute jnsb3=covmat(8,8).
compute jnsb1b3=covmat(4,8).
end if.
end if.
do if (model > 3).
compute ymat(5,im) = coeff((1+im),1).
end if.
do if (xmy = 1).
compute ymat(6,im) = coeff((2+nmeds+im),1).
end if.
do if (vy = 1).
compute ymat(6,im) = coeff((3+nmeds+im),1).
end if.
do if (qy = 1 and vy = 1).
compute ymat(6,im) = coeff((5+nmeds+((im-1)*2)),1).
compute ymat(7,im) = coeff((6+nmeds+((im-1)*2)),1).
end if.
do if (vqy = 1).
compute ymat(6,im) = coeff((6+nmeds+((im-1)*3)),1).
compute ymat(7,im) = coeff((7+nmeds+((im-1)*3)),1).
compute ymat(8,im) = coeff((8+nmeds+((im-1)*3)),1).
end if.
do if (wmy = 1).
compute ymat(7,im) = coeff((3+nmeds+im-wy),1).
end if.
do if (wmy = 1 and vy = 1).
compute ymat(7,im) =coeff((4+(nmeds*2)+im-wy),1).
end if.
do if (wmy = 1 and vy = 1 and wvmy = 1).

```

```

compute ymat(7,im) =
coeff((6+(nmeds*2)+((im-1)*2)-wy),1).
compute ymat(8,im) =
coeff((7+(nmeds*2)+((im-1)*2)-wy),1).
end if.
do if (wmy = 1 and zmy = 1).
compute ymat(6,im) = coeff((6-(wzy*3)+(wzm-
1)+(nmeds*2)+((im-1)*2)*wzm)+((im-1)*(1-
wzm))-((zy-wzy)*2),1).
compute ymat(7,im) = coeff((3-
zy+im+nmeds),1).
do if (wzmy = 1).
compute ymat(8,im) = coeff((7-
(wzy*3)+(nmeds*2)+((im-1)*2)),1).
end if.
end if.
do if (nmods > 0 and model <> 5).
loop indlp = 1 to nrow(modvals).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp)).
do if (model > 6).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp))*csum(ym
at(5:8,im)*vmat(5:8,indlp)).
end if.
end loop.
compute indboot((bt+(im-
1)*(boot+1)),:)=t(indeff).
do if (imm = 1 or imm = 3).
compute
indbootp(bt,im)=ymat(2,im)*ymat(5,im).
do if (imm=3).
compute
indbootp(bt,(im+nmeds))=ymat(3,im)*ymat(5,i
m).
do if (zdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxz-cminz).
end if.

```

```

end if.
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
end if.
end if.
do if ((imm= 2 or imm = 4) or model = 74).
compute
indbootp(bt,im)=ymat(1,im)*ymat(6,im).
do if (imm=4).
compute
indbootp(bt,(im+nmeds))=ymat(1,im)*ymat(7,i
m).
do if (qdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxq-cminq).
end if.
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
end if.
end if.
do if (imm=7).
compute
indbootp(bt,im)=ymat(2,im)*ymat(6,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(2,im)*y
mat(6,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(6,im)*onesmv1)+(ymat(2,im)*y
mat(6,im)*t(immmv1)).
do if (wdich=1).

```

```

compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxv-cminv).
end if.
end if.
do if (imm=5).
compute
indbootp(bt,im)=ymat(4,im)*ymat(5,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(4,im)*y
mat(5,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(3,im)*ymat(5,im)*onesmv1)+(ymat(4,im)*y
mat(5,im)*t(immmv1)).
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).

```

```

end if.
do if (zdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxz-
cminz).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxz-cminz).
end if.
end if.
do if (imm=6).
compute
indbootp(bt,im)=ymat(1,im)*ymat(8,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(1,im)*ymat(6,im)*onesmv2)+(ymat(1,im)*y
mat(8,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(7,im)*onesmv1)+(ymat(1,im)*y
mat(8,im)*t(immmv1)).
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxv-cminv).
end if.
do if (qdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxq-
cminq).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-

```

```

1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxq-cminq).
end if.
end if.
do if ((model = 58 or model = 59) and
wvdich=1).
compute
indbootp(bt,im)=(ymat(1,im)*ymat(7,im)*(cmax
w-cminw))+(ymat(2,im)*ymat(5,im)*(cmaxw-
cminw))+(ymat(2,im)*ymat(7,im)*((cmaxw*cma
xw)-(cminw*cminw))).
end if.
end if.
do if (model = 4 or model = 5).
compute
indboot(bt,im)=csum(ymat(1:4,im)*vmat(1:4,1
))*csum(ymat(5:8,im)*vmat(5:8,1)).
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.000000000000001.
end if.
compute sumind=rsum(indboot(bt,:)).
do if (im=nmeds).
compute
pmeff(bt,2:(im+1))=indboot(bt,:)/(sumind+coeff(
(2+nmeds),1)).
end if.
compute
rmeff(bt,(im+1))=indboot(bt,im)/coeff((2+nmeds
),1).
compute
abpseff(bt,(im+1))=indboot(bt,im)/stddevy.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*stddevx.
do if (ncovs > 0 or clsdmy > 0).
compute
abpseff(bt,(im+1))=indboot(bt,im)/sdy cov.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*sdx cov.
end if.

```

```

do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,1) = r2my-(r2full-r2xy).
compute abmmr = 1.
do if (indboot(bt,im) < 0).
compute abmmr = cmin(abmm).
end if.
do if (indboot(bt,im) > 0).
compute abmmr = cmax(abmm).
end if.
compute kappa2(bt,1)=indboot(bt,im)/abmmr.
compute tmp = indboot(bt,im)/abmmr.
end if.
end if.
end if.
do if (model = 6).
do if (nmeds = 2).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(4,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3).
compute
indboot(bt,3)=mmpaths(3,1)*mmpaths(4,3).
end if.
do if (nmeds = 3).
compute indboot(bt,1) =
mmpaths(2,1)*mmpaths(5,2).
compute indboot(bt,2) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3).
compute indboot(bt,3) =
mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4).
compute indboot(bt,4) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4).
compute indboot(bt,5) =
mmpaths(3,1)*mmpaths(5,3).
compute indboot(bt,6) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4).

```



```

compute indboot(bt,7) =
mmpaths(4,1)*mmpaths(5,4).
end if.
do if (nmeds = 4).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(6,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(6,3).
compute
indboot(bt,3)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(6,4).
compute
indboot(bt,4)=mmpaths(2,1)*mmpaths(5,2)*mmpaths(6,5).
compute
indboot(bt,5)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(6,4).
compute
indboot(bt,6)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3)*mmpaths(6,5).
compute
indboot(bt,7)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,8)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,9)=mmpaths(3,1)*mmpaths(6,3).
compute
indboot(bt,10)=mmpaths(3,1)*mmpaths(4,3)*mmpaths(6,4).
compute indboot(bt,11)
=mmpaths(3,1)*mmpaths(5,3)*mmpaths(6,5).
compute indboot(bt,12) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,13) =
mmpaths(4,1)*mmpaths(6,4).

```

```

compute indboot(bt,14) =
mmpaths(4,1)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,15) =
mmpaths(5,1)*mmpaths(6,5).
end if.
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.00000000000001.
end if.
compute
pmeff(bt,:)=indboot(bt,:)/(rsum(indboot(bt,:))+m
mpaths(nrow(mmpaths),1)).
compute rmeff(bt,:) =
indboot(bt,:)/mmpaths(nrow(mmpaths),1).
compute abpseff(bt,:)=indboot(bt,:)/stddevy.
compute
abcseff(bt,:)=(stddevx*indboot(bt,:))/stddevy.
do if (ncovs > 0 or clsdmy > 0).
compute abpseff(bt,:)=indboot(bt,:)/sdyconv.
compute
abcseff(bt,:)=(sdxcov*indboot(bt,:))/sdyconv.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,:) = r2my-(r2full-r2xy).
end if.
end if.
end if.
end loop.
end if.
end loop.
do if (savboot = 1 and bt > 1).
compute bootstrp((bt-1),:)=bootcoef.
end if.
end loop.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.

```

```

release datayed, dat, datamed, data, y, x,
datatm, dataty, mints2, cons, mnv, tmp.
do if (mc > 0).
compute x1 = sqrt(-
2*ln(uniform(mc,nrow(mcmats))))&*cos((2*3.14
159265358979)*uniform(mc,nrow(mcmats))).
compute x1=x1*chol(mcmats).
loop i=1 to nrow(x1).
compute x1(i,:)=x1(i,:)+t(mccoeff).
end loop.
loop i = 1 to nmeds.
compute
indboot(2:nrow(indboot),i)=x1(:,i)&*x1(:,(i+nme
ds)).
end loop.
end if.
do if (ttt = 0 and model > 3).
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
end if.
do if (model <> 74).
do if (dichy = 0).
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 't', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 't', 'p', 'LLCI', 'ULCI'/format =
F10.4.
else.
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 'Z', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.

```

```

print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 'Z', 'p', 'LLCI', 'ULCI' /format =
F10.4.
end if.
end if.
end if.
do if (ttt > 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
compute clbs = {modvnm2, 'Effect', 'SE', 't', 'p',
'LLCI', 'ULCI'}.
compute tratio = (directv&/sedir).
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
compute outp = {modvalsd, directv, sedir,
tratio, p}.
do if (dichy = 0).
compute temp1=directv-sqrt(jncrit)*sedir.
compute temp2=directv+sqrt(jncrit)*sedir.
compute outp = {outp, temp1, temp2}.
end if.
do if (dichy = 1).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute temp = directv-abs(xp2)*sedir.
compute outp = {outp, temp}.
compute temp = directv+abs(xp2)*sedir.
compute outp = {outp, temp}.
compute clbs = {modvnm2, 'Effect', 'SE', 'Z', 'p',
'LLCI', 'ULCI'}.
end if.
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
print outp/title = 'Conditional direct effect(s) of
X on Y at values of the moderator(s):'/format =
F10.4 /cnames = clbs.
end if.
do if (nmods > 0 and model <> 5).
do if (model < 4).

```

```

print/title =
*****
*****!

compute zmat(1,1) = 1*(1-yes).
compute cfse = make(nrow(modvals),1,0).
loop #m = 1 to nrow(modvals).
do if (model = 1).
compute zmat(2,1)=modvals(#m,1)**2.
compute zmat(5,1)=2*modvals(#m,1).
end if.
do if (model = 2 or model = 3).
compute zmat(2,1)=modvals(#m,2)**2.
compute zmat(3,1)=modvals(#m,1)**2.
compute
zmat(4,1)=(modvals(#m,1)**2)*(modvals(#m,2)
**2)*(1-yes2).
compute zmat(5,1)=2*modvals(#m,2)*(1-yes).
compute zmat(6,1)=2*modvals(#m,1)*(1-yes).
compute
zmat(7,1)=2*modvals(#m,1)*modvals(#m,2)*(1-
yes).
compute
zmat(8,1)=2*modvals(#m,1)*modvals(#m,2).
compute
zmat(9,1)=2*modvals(#m,1)*(modvals(#m,2)**
2)*(1-yes2).
compute
zmat(10,1)=2*(modvals(#m,1)**2)*modvals(#m
,2)*(1-yes2).
end if.
compute cfse(#m,1)=sqrt(csum(zmat&*cmat)).
end loop.
end if.
do if (nmods > 0).
compute clbs = {modvnm, 'Effect'}.
loop im = 1 to nmeds.
compute obs = t(indboot(1+(im-1)*(boot+1),:)).
compute outp = {modvals, obs}.
do if (model < 4).
compute tstat = obs&/cfse.

```

```

do if (dichy = 0).
compute pval = 2*(1-tcdf(abs(tstat), (n-
ncol(xy)))).
compute temp=obs-sqrt(jncrit)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp=obs+sqrt(jncrit)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 't', 'p', 'LLCI', 'ULCI'}.
compute jnclbs=clbs.
end if.
do if (dichy = 1).
compute pval = 2*(1-cdfnorm(abs(tstat))).
compute temp = obs-abs(xp2)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp = obs+abs(xp2)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute jnclbs=clbs.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indboot((1+(im-
1)*(boot+1)),:).
compute indboot2 = indboot((2+(im-
1)*(boot+1)):(1+(im-1)*(boot+1)+boot),:).
compute mnind = t(csum(indboot2)/boot).
compute stdind=t(sqrt((cssq(indboot2)-
((csum(indboot2)**2)/boot))/(boot-1))).
compute llci=make(1,ncol(indboot2),-999).
compute ulci=make(1,ncol(indboot2),-999).
loop #e = 1 to ncol(indboot2).
.
compute temp = indboot2(:,#e).
compute temp(GRADE( indboot2(:,#e) )) =
indboot2(:,#e).
compute badlo = 0.
compute badhi = 0.

```

```

do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.

```

```

do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {modvals, obs, stdind, t(llci),
t(ulci)}.
compute clbs = {modvnm, 'Effect', 'Boot SE',
'BootLLCI', 'BootULCI'}.
end if.
compute mtemp = mnames(1, im).
compute rlbs =
make(nrow(modvals),1,mnames(1,im)).
do if (model < 4).
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
do if (yes=0).
print outp/title = 'Conditional effect of X on Y at
values of the moderator(s):'/cnames =
clbs/format = F10.4.
else.
compute outp=outp(:,3:ncol(outp)).
compute clbs=clbs(:,3:ncol(clbs)).
compute clbs(1,1)='Contrast'.
print outp/title = 'Contrast of conditional effects
of X on Y'/cnames = clbs/format = F10.4.
end if.
end if.
do if (model > 5 and mod74dic <> 1).
do if (im = 1).
print/title = 'Conditional indirect effect(s) of X on
Y at values of the moderator(s):'.
end if.
print outp/title = 'Mediator'/rnames =
rlbs/cnames = clbs/format = F10.4.

```



```

else if (mod74dic = 1 and model = 74).
do if (im = 1).
print/title = 'Indirect effect(s) of X on Y:'.
compute clbs3=clbs(1,2:ncol(clbs)).
end if.
compute outp3=outp(1,2:ncol(outp)).
print outp3/title = 'Mediator'/rnames =
rlbs/cnames = clbs3/format = F10.4.
end if.
end loop.
loop i = notes to 1 by -1.
do if (note(i,1) = 4 and yes=0).
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 5 and yes=0).
print/title = 'Values for quantitative moderators
are the mean and plus/minus one SD from
mean.'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 14 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    below the mean was replaced
with the minimum because one SD below the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.
do if (note(i,1) = 15 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    above the mean was replaced
with the maximum because one SD above the
mean'/space=0.

```

```

print/title = '      is outside of the range of the
data.'/space=0.
end if.
end loop.
do if (model = 3 and yes=0).
compute jnvals=make(nrow(matw),7,0).
compute jnvals(:,1)=matw.
compute jnvals(:,2)=jnb1+jnb3*jnvals(:,1).
compute
jnvals(:,3)=sqrt(jnsb1+2*jnvals(:,1)*jnsb1b3+(jn
vals(:,1)*jnvals(:,1))*jnsb3).
compute jnvals(:,4)=jnvals(:,2)/jnvals(:,3).
do if (dichy = 0).
compute jnvals(:,5)=2*(1-tcdf(abs(jnvals(:,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(:,5)=2*(1-
cdfnorm(abs(jnvals(:,4)))).
end if.
compute jnvals(:,6)=jnvals(:,2)-
sqrt(jncrit)*jnvals(:,3).
compute
jnvals(:,7)=jnvals(:,2)+sqrt(jncrit)*jnvals(:,3).
compute clbs={clbs(:,1),clbs(:,3:ncol(clbs))}.
do if (coeffci = 0).
compute jnvals =jnvals(:,1:(ncol(jnvals)-2)).
end if.
print jnvals/title = 'Conditional effect of X*M
interaction at values of W:.'/cnames
=clbs/format = F10.4.
end if.
do if (jn = 1 and (model = 1 or model = 3) and
jndich = 0 and yes=0).
print/title = '***** JOHNSON-
NEYMAN TECHNIQUE
*****'.
compute ajn =(jncrit*jnsb3)-(jnb3*jnb3).
compute bjn = 2*((jncrit*jnsb1b3)-(jnb1*jnb3)).
compute cjn = (jncrit*jnsb1)-(jnb1*jnb1).

```

```

compute radarg = (bjn*bjn)-(4*ajn*cjn).
compute den = 2*ajn.
compute nrts = 0.
do if (radarg >= 0 and den <> 0).
compute x21 = (-bjn+sqrt(radarg))/den.
compute x22 = (-bjn-sqrt(radarg))/den.
compute roots = 0.
do if (x21 >= jnmin and x21 <= jnmax).
compute nrts = 1.
compute roots = {roots; x21}.
end if.
do if (x22 >= jnmin and x22 <= jnmax).
compute nrts = nrts + 1.
compute roots = {roots; x22}.
end if.
compute roots={roots,make(nrow(roots),2,0)}.
compute modtemp=m.
do if (model=3).
compute modtemp=w.
end if.
do if (nrts > 0).
compute roots = roots(2:nrow(roots),1:3).
compute roots(1,2)=(csum(modtemp <
roots(1,1))/n)*100.
compute roots(1,3)=(csum(modtemp >
roots(1,1))/n)*100.
do if (nrow(roots)=2).
compute roots(2,2)=(csum(modtemp <
roots(2,1))/n)*100.
compute roots(2,3)=(csum(modtemp >
roots(2,1))/n)*100.
end if.
print roots/title = 'Moderator value(s) defining
Johnson-Neyman significance
region(s)'/clabels = 'Value', '% below', '%
above'/format = F10.4.
compute jnvals=make((21+nrts),7,0).
loop i= 0 to 20.
compute jnvals((i+1),1)=jnmin+(i*((jnmax-
jnmin)/20)).

```

```

end loop.
loop i = 1 to nrts.
loop j = 2 to nrow(jnvals).
do if ((roots(i,1) > jnvals((j-1),1)) and (roots(i,1)
< jnvals(j,1))).
compute
jnvals((j+1):(21+i),1)=jnvals(j:(20+i),1).
compute jnvals(j,1)=roots(i,1).
end if.
end loop.
end loop.
loop i = 1 to nrow(jnvals).
compute jnvals(i,2)=jnb1+jnb3*jnvals(i,1).
compute
jnvals(i,3)=sqrt(jnsb1+2*jnvals(i,1)*jnsb1b3+(jn
vals(i,1)*jnvals(i,1))*jnsb3).
compute jnvals(i,4)=jnvals(i,2)/jnvals(i,3).
do if (dichy = 0).
compute jnvals(i,5)=2*(1-tcdf(abs(jnvals(i,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(i,5)=2*(1-
cdfnorm(abs(jnvals(i,4)))).
end if.
compute jnvals(i,6)=jnvals(i,2)-
sqrt(jncrit)*jnvals(i,3).
compute
jnvals(i,7)=jnvals(i,2)+sqrt(jncrit)*jnvals(i,3).
end loop.
do if (model = 1).
print jnvals/title = 'Conditional effect of X on Y
at values of the moderator (M)'/cnames
=jnclbs/format = F10.4.
end if.
do if (model = 3).
compute
jnclbs={jnclbs(:,1),jnclbs(:,3:ncol(jnclbs))}.

```

```

print jnvals/title = 'Conditional effect of X*M on
Y at values of the moderator (W)'/cnames
=jnclbs/format = F10.4.
end if.
end if.
do if (nrts = 0).
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
else.
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
end if.
end if.
do if (model < 4 and (plot = 1)).
compute dataplot =
make((nrow(modvals)*nrow(matx)),(ncol(modv
als)+1),0).
compute tmp = 1.
loop i = 1 to nrow(modvals).
loop j = 1 to nrow(matx).
compute dataplot(tmp,.)={matx(j,1),
modvals(i,:)}.
compute tmp=tmp+1.
end loop.
end loop.
compute dataplot = {dataplot,
make(nrow(dataplot),(1+dichy),0)}.
compute dataplo2 = make(nrow(dataplot),1,1).
do if (model = 1).
compute dataplo2 = {dataplo2, dataplot(:,2),
dataplot(:,1), (dataplot(:,1)*dataplot(:,2))}.
end if.
do if (model = 2 or model = 3).
compute dataplo2 = {dataplo2, dataplot(:,3),
dataplot(:,1), (dataplot(:,1)*dataplot(:,3)),
dataplot(:,2), (dataplot(:,1)*dataplot(:,2))}.

```

```

do if (model = 3).
compute dataplo2 = {dataplo2,
(dataplot(:,2)*dataplot(:,3)),
(dataplot(:,1)*dataplot(:,2)*dataplot(:,3))}.
end if.
end if.
loop i = 1 to nrow(dataplot).
compute tmp=dataplo2(i,:).
do if (ncovs > 0).
compute tmp = {tmp, covmeans}.
end if.
do if (cluster > 0).
compute tmp = {tmp, cldmeans}.
end if.
compute dataplot(i,(ncol(dataplot)-
(dichy)))=tmp*coeffplt.
do if (dichy = 1).
compute
dataplot(i,(ncol(dataplot)))=exp(tmp*coeffplt)/(1
+exp(tmp*coeffplt)).
end if.
end loop.
compute clbs = {xname, modvnm, 'yhat'}.
do if (dichy = 1).
compute clbs = {xname, modvnm, 'ln(odds)',
'prob'}.
end if.
print/title =
*****
*****!

```

```

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (model = 1).
do if (dichy = 0).
print/title = 'DATA LIST FREE/Opra_Pract
Technological_Advan
Operational_Performance.' /space=1.
else if (dichy = 1).
print/title = 'DATA LIST FREE/Opra_Pract
Technological_Advan logodds prob.'
/space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
Operational_Performance BY
Technological_Advan.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH Operational_Performance BY
Opra_Pract.' /space=0.
end if.
end if.

```

```

do if (dichy = 1).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
logodds BY Technological_Advan.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
prob BY Technological_Advan.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Opra_Pract.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Opra_Pract.' /space=0.
end if.
end if.
end if.
do if (model = 2 or model = 3).
do if (dichy = 0).
print/title = 'DATA LIST FREE/Opra_Pract
Technological_Advan
Operational_Performance.' /space=1.
end if.
do if (dichy = 1).
print/title = 'DATA LIST FREE/Opra_Pract
Technological_Advan logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' /format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
Operational_Performance BY
Technological_Advan/PANEL ROWVAR='
/space=0.
end if.

```



```

do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH Operational_Performance BY
Opra_Pract/PANEL ROWVAR=' /space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
logodds BY Technological_Advan/PANEL
ROWVAR=' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
prob BY Technological_Advan/PANEL
ROWVAR=' /space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Opra_Pract/PANEL
ROWVAR=' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Opra_Pract/PANEL
ROWVAR=' /space=0.
end if.
end if.
end if.
do if (ncovs > 0).
print/title = '* Estimates are based on setting
covariates to their sample means.'.
end if.
end if.
end if.
do if ((imm > 0) or (model = 74 and
mod74dic=0) or ((model = 58 or model = 59)
and wvdich=1))).
do if (imm > 4 and imm < 8).

```

```

compute
indbootp={indbootp(:,1:nmeds),indboop1,indbo
op2}.
end if.
compute obsprod = t(indbootp(1,:)).
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indbootp(1,:).
compute indbootp = indbootp(2:(boot+1),:).
compute mnindp = t(csum(indbootp)/boot).
compute stdindp=t(sqrt((cssq(indbootp)-
((csum(indbootp)&**2)/boot))/(boot-1))).
compute llcip=make(1,ncol(indbootp),-999).
compute ulcip=make(1,ncol(indbootp),-999).
loop #e = 1 to ncol(indbootp).

.
compute temp = indbootp(:,#e).
compute temp(GRADE( indbootp(:,#e) )) =
indbootp(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).

```

```

compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcip(1,#e)=llcit.
compute ulcip(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {obsprod, stdindp, t(llcip),
t(ulcip)}.
compute clbs = {'Effect', 'SE(Boot)', 'BootLLCI',
'BootULCI'}.
do if (model = 8 or model = 12).
print/title='-----'.
compute outpimm=outp(1:nmeds,:).

```

```

print/title = 'Indirect effect of highest order
product:.'/space=0.
print outpimm/title = 'Mediator'/cnames =
clbs/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
compute clbs3 = {'Index', 'SE(Boot)',
'BootLLCI', 'BootULCI'}.
do if (imm < 5).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=outp((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
do if (imm > 4 and imm < 8).

```

```

print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'.

compute outpimm=outp(1:nmeds,:).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
compute
outpimm=outp((nmeds+1):nrow(outp),:).
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.

loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),clbs3}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3(1,1)=modvnm(1,1).
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.

```

```

print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
end if.
do if (boot = 0).
do if (model = 8 or model = 12).
print/title='-----'.
compute obsimm=obsprod(1:nmeds,:).
print/title = 'Indirect effect of highest order
product.'/space=0.
print obsimm/title = 'Mediator'/clabels =
'Effect'/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
do if ((imm > 0 and imm < 5) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=obsprod((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
end loop.
end if.

```

```

do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****!
.
compute outpimm=obsprod(1:nmeds,1).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
compute
outpimm=obsprod((nmeds+1):nrow(obsprod),:)
.
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),'Index'}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3={modvnm(1,1),'Index'}.
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator: '/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
end if.
end if.

```

```

end if.
compute conmake=0.
compute concols = 0.
do if ((model > 3 and model < 7) and (contrast
> 0) and nmods = 0 and nmeds > 1 ).
compute concols =
((ncol(indboot)*(ncol(indboot)-1))/2).
compute indcon=make(nrow(indboot),concols,-
999).
compute conkey = { ' ', ' ', ' ' }.
compute temp=1.
compute conmake=1.
loop i = 1 to (ncol(indboot)-1).
loop j = (i+1) to (ncol(indboot)).
compute indcon(:,temp)=indboot(:,i)-
indboot(:,j).
do if (contrast = 2).
compute indcon(:,temp)=abs(indboot(:,i))-
abs(indboot(:,j)).
end if.
do if (model <= 6).
compute conkey={conkey; mnames(1,i),
'minus', mnames(1,j)}.
end if.
do if (model = 6).
compute conkey={conkey; indlbl2(i,1), 'minus',
indlbl2(j,1)}.
end if.
compute temp=temp+1.
end loop.
end loop.
end if.
do if (model = 4 or model = 5).
compute clbs = {'Effect'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obs = {csum(obs); obs}.
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.

```



```

compute rlbs = {rlbs;
cntname(1:ncol(indcon),1)}.
end if.
compute outp = obs.
compute outp2=outp.
do if (effsize = 1 and dichy = 0).
compute
pmeff(:,1)=rsum(pmeff(:,2:ncol(pmeff))).
compute
rmeff(:,1)=rsum(rmeff(:,2:ncol(rmeff))).
compute
abpseff(:,1)=rsum(abpseff(:,2:ncol(abpseff))).
compute
abcseff(:,1)=rsum(abcseff(:,2:ncol(abcseff))).
compute eff = {pmeff, rmeff, abpseff, abcseff}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute eff = {eff, r245}.
compute r245obs = {r245(1,1);r245(1,1)}.
compute kappa2ob = kappa2(1,1).
end if.
compute pmobs = t(pmeff(1,1:(nmeds+1))).
compute rmobs = t(rmeff(1,1:(nmeds+1))).
compute psobs = t(abpseff(1,1:(nmeds+1))).
compute csobs = t(abcseff(1,1:(nmeds+1))).
do if (contrast = 0).
compute outp2 = {obs, psobs, csobs, pmobs,
rmobs}.
end if.
do if (contrast > 0).
compute obs2=obs(1:nrow(psobs),:).
compute outp2 = {obs2, psobs, csobs, pmobs,
rmobs}.
end if.
compute clbs = {'ab', 'ab_ps', 'ab_cs', 'ab/c',
'ab/c'}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute outp2 = {outp2, r245obs}.
compute clbs = {clbs, 'R-sq_med'}.

```

```

do if (boot = 0 and mc = 0).
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
do if (boot = 0 and mc = 0).
do if (nmeds = 1).
compute outp2 = outp2(2,:).
compute rlbs = rlbs(2,1).
end if.
do if (model=5).
compute outp2=outp2(:,1:3).
end if.
print outp2/title = 'Indirect effect(s) of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1 and nmeds >
1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0 or mc > 0).
compute indboot = {rsum(indboot), indboot}.
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
compute ones = make(bootSZ,1,1).
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(bootSZ+1),:).
compute mnind = t(csum(indboot)/bootSZ).

```

```

compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/bootSZ))/(bootSZ-1))).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).

.

compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.

```

```

end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llici(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
do if (effsize = 1 and dichy = 0 and boot > 0).
compute estmte=eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).
.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.

```

```

compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).

```

```

compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
end if.
end if.
do if (boot > 0 or mc > 0).
compute outp = {obs, stdind, t(llci), t(ulci)}.
do if (nmeds = 1).
compute outp = outp(2,:).
compute rlbs = rlbs(2,1).
end if.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
do if (mc > 0).
compute clbs = {'Effect', 'MC SE', 'MC LLCI',
'MC ULCI'}.
end if.
print outp/title = 'Indirect effect of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (dichy = 0 and effsize = 1 and boot > 0).
compute outp = {psobs,
stdindf((2*(nmeds+1)+1):(3*(nmeds+1)),1),
t(llcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))),
t(ulcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Partially standardized indirect
effect of X on Y'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute outp = {csobs,
stdindf((3*(nmeds+1)+1):(4*(nmeds+1)),1),
t(llcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))),
t(ulcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.

```

```

print outp/title = 'Completely standardized
indirect effect of X on Y'/rnames = rlbs/cnames
= clbs/format = F10.4.
compute outp = {pmobs,
stdindf(1:(nmeds+1),1),t(llcif(1,1:(nmeds+1))),
t(ulcif(1,1:(nmeds+1))) }.
do if (model = 4).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
compute outp = {rmobs, stdindf(((nmeds+1)+1)
:(2*(nmeds+1)) ,1),
t(llcif(1,((nmeds+1)+1):(2*(nmeds+1)))),
t(ulcif(1,((nmeds+1)+1):(2*(nmeds+1))))}.
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
do if (nmeds = 1 and cluster = 0 and ncovs = 0
and model = 4).
compute r245obs = r245obs(1,1).
compute outp = {r245obs,
stdindf((4*(nmeds+1)+1):(4*(nmeds+1)+1),1),
t(llcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1))),
t(ulcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1)))}.
print outp/title = 'R-squared mediation effect
size (R-sq_med)'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
end if.
do if (normal = 1).

```

```

do if (nmeds = 1).
print sobel/title = 'Normal theory tests for
indirect effect'/clabels = 'Effect', 'se', 'Z',
'p'/format = F10.4.
end if.
do if (nmeds > 1).
compute rlbs2 = rlbs(2:nrow(rlbs),1).
print sobel/title = 'Normal theory tests for
specific indirect effects'/rnames = rlbs2/clabels
= 'Effect', 'se', 'Z', 'p'/format = F10.4.
end if.
end if.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
do if (model = 6).
compute clbs = {'eff'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obsfsum=csum(obs).
compute obs = {obsfsum; obs}.
compute indlbl = indlbl(1:nrow(obs),1).
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute indlbl = {indlbl;
cntname(1:ncol(indcon),1)}.
end if.
compute obs2=obs.
do if (boot = 0).
do if (dichy = 0 and effsize = 1).
compute clbs = {'eff', 'eff_ps', 'eff_cs', 'eff/c',
'eff/c'}.
do if (ncovs = 0).

```



```

compute obs = {obs, obs/stddevy,
(obs*stddevx/stddevy), obs/ctot,
obs/mmpaths(nrow(mmpaths),1)}.
end if.
do if (ncovs > 0 or clsdmy > 0).
compute obs = {obs, obs/sdycov,
(obs*sdxcov/sdycov),
obs/(obsfsum+mmpaths(nrow(mmpaths),1)),
obs/mmpaths(nrow(mmpaths),1)}.
end if.
compute obs2=obs.
do if (contrast > 0).
compute obs2 = obs(1:(nrow(obs)-concols),:).
end if.
end if.
print obs2/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute indboot = {rsum(indboot), indboot}.
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(boot+1),:).
compute mnind = t(csum(indboot)/boot).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/boot))/(boot-1))).
compute temp = nrow(indboot).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).

```

```

loop #e = 1 to ncol(indboot).

.

compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.

```

```

do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute obs = {obs, stdind, t(llci), t(ulci)}.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
print obs/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (effsize = 1 and dichy = 0).
compute indboot=indboot(:,1:(ncol(indboot)-
concols)).
compute eff = {rsum(abpseff), abpseff,
rsum(abcseff), abcseff, rsum(pmeff), pmeff,
rsum(rmeff), rmeff}.
compute effobs = eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).
.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).

```

```

compute badlo = 0.
compute badhi = 0.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(effobs(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.

```

```

compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute temp2 = stdindf(1:ncol(indboot),1).
compute temp3 = effobs(:,1:ncol(indboot)).
compute templo = llcif(1,1:ncol(indboot)).
compute temphi = ulcif(1,1:ncol(indboot)).
compute outp = {t(temp3), temp2,t(templo),
t(temphi)}.
print outp/title = 'Partially standardized indirect
effect of X on Y'/cnames = clbs/rnames =
indlbl/format = F10.4.
compute temp2 =
stdindf((ncol(indboot)+1):(2*ncol(indboot)),1).
compute temp3 =
effobs(:,(ncol(indboot)+1):(2*ncol(indboot))).
compute templo =
llcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute temphi =
ulcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templo),
t(temphi)}.
print outp/title = 'Completely standardized
indirect effect of X on Y'/cnames = clbs/rnames
= indlbl/format = F10.4.
compute temp2 =
stdindf((2*(ncol(indboot))+1):(3*ncol(indboot)),1
).
compute temp3 =
effobs(:,(2*(ncol(indboot))+1):(3*ncol(indboot)))
.

```

```

compute templow =
llcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute temphi =
ulcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
compute temp =
eff(:,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temp2 =
stdindf((3*(ncol(indboot))+1):(4*ncol(indboot)),1
).
compute temp3 =
effobs(:,(3*(ncol(indboot))+1):(4*ncol(indboot)))
.
compute templow =
llcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temphi =
ulcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
end if.
end if.
do if (nmeds = 2).
compute effkey = {xname, '->', mnames(1,1), '-
>', yname, '', ''}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, '', ''}.
compute effkey = {indlbl(2:4,1), effkey}.
end if.
do if (nmeds = 3).
compute effkey = {xname, '->', mnames(1,1), '-
>', yname, '', '', ''}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, '',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, '',' ',' ',' '}.
compute effkey = {indbl(2:8,1), effkey}.
end if.
do if (nmeds = 4).
compute effkey = {xname, '->' , mnames(1,1), '-
>', yname, '',' ',' ',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,4), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname, '',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,4), '->', yname, '',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, '',' '}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', mnames(1,4), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,4), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', mnames(1,4), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,4), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {indl(2:16,1), effkey}.
end if.
print effkey/title = 'Indirect effect key'/format =
A8.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
end if.
DO IF (medte2=1).
compute
name=t(mdlnms2(2:nrow(mdlnms2),1)).
compute tmpdat=datayed(:,2:ncol(datayed)).
compute dd=tmpdat(:,1).

```



```

compute
dd={dd,tmpdat(:,(2+nmeds)),tmpdat(:,2:(1+nmeds))}.
compute stratify=0.
do if (ncovs > 0).
compute cname = cnames.
compute
dd={dd,tmpdat(:,(3+nmeds):(2+nmeds+ncovs))}.
end if.
compute method = 1.
compute samples = boot.
do if (mc > 0).
compute samples = mc.
compute method=2.
end if.
compute omnibus = 1.
compute booterr = 0.
do if (mcx < 0 or mcx > 4).
compute mcx = 1.
end if.
compute ovals = ncol(design(dd(:,1))).
compute nte = notes.
compute criterr = 0.
compute xskip = 0.
compute dumok = 0.
compute badend = 0.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute priorlo = -9999999.
compute priorhi = 9999999.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute reps = samples.
do if (contrast = 1).
compute note(nte,1) = 26.
compute nte=nte+1.

```

```

end if.
do if (effsize = 1).
compute note(nte,1) = 27.
compute nte=nte+1.
end if.
do if (normal = 1).
compute note(nte,1) = 28.
compute nte=nte+1.
end if.
do if (method = 2).
do if (omnibus = 1).
compute note(nte,1) = 25.
compute nte=nte+1.
end if.
compute samples = 0.
do if (reps < 1000).
compute reps = 1000.
end if.
end if.
do if (method = 1).
compute samples = reps.
end if.
compute temp = ncol(dd).
compute n = nrow(dd).
compute ones = make(n,1,1).
compute temp=dd(:,2).
compute nxd = ncol(temp).
compute nx = nxd.
do if (nx = 1 and mcx > 0).
compute temp = dd.
compute temp(GRADE(dd(:,2)),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,2)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute mnvls = cmin(t(nnvls)).
do if (mnvls < 2).
compute errs = errs + 1.
compute runerrs(errs,1) = 37.
compute criterr = 1.

```

```

end if.
do if (nvls > 9).
compute errs = errs+1.
compute runerrs(errs,1) = 36.
compute criterr = 1.
end if.
do if (criterr = 0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,2).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,2) <> nnvls((temp-1),1)).
compute nnvls(temp,1)=dd(i,2).
compute temp = temp+1.
end if.
end loop.
do if (mcx > 0).
compute x = dummy(:,2:ncol(dummy)).
compute nx = ncol(x).
compute minus1 = make(1,ncol(x),-1).
compute xdes=make((nx+1),3,0).
compute xdes(1,1)=dd(1,2).
compute xdes(1,2)=1.
compute temp = 2.
loop k = 2 to n.
do if (dd(k,2) <> dd((k-1),2)).
compute xdes(temp,2) = k.
compute xdes(temp,1) = dd(k,2).
compute xdes((temp-1),3) = k-1.
compute temp=temp+1.
end if.
end loop.
compute xdes((temp-1),3)=n.
compute xdes = {xdes, (xdes(:,3)-xdes(:,2)+1)}.
do if (stratify <> 0).
compute note(nte,1) = 7.
compute nte=nte+1.
end if.
do if (mcx = 4).

```

```

loop k = 1 to n.
do if (rsum(x(k,:)) = 0).
compute x(k,:) = minus1.
end if.
end loop.
end if.
do if (mcx = 2 or mcx = 3).
compute note(nte,1)=4.
compute nte = nte+1.
loop k = 1 to n.
do if (rsum(x(k,:)) > 0).
loop i = 1 to ncol(x).
do if (x(k,i) = 0).
compute x(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcx = 3).
compute conmat1={-8,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1; 0,0,0,0,0,-
3,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.
loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,:)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute x(k,:)=conmat1((rsum(x(k,:))+1),:).
end loop.
end if.
end if.
compute xdata = x.
compute xskip = 1.
compute xname = ddd(1,1:nx).
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).

```

```

compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcx = 4).
compute dummat(1,:) = minus1.
end if.
do if (mcx = 2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcx = 3).
compute dummat=conmat1.
end if.
compute dummat={nnvls, dummat}.
compute xname2={name(1,2), xname}.
end if.
end if.
end if.
compute nm=nmeds.
compute r2b = make(nm,1,-999).
compute indeff = make((nx*nm),1,-999).
compute cov =
make((nx*nm+nm),(nx*nm+nm),0).
compute mns=make(nrow(cov),1,0).
compute ny = ncol(dd)-nm-nxd-ncovs.
compute y = dd(:,1).
compute ydata = y.
compute yname = name(1,1).
do if (xskip = 0).
compute x = dd(:,2:(1+nxd)).
compute xdata = x.
compute xname = name(1,2:(1+nxd)).
compute indlbs = t(xname).
end if.
compute m = dd(:,(2+nxd):(1+nxd+nm)).
compute mdata = m.

```

```

compute mlab = { ' M1 = ';' ' M2 = ';' ' M3 = ';' '
M4 = ';' ' M5 = ';' ' M6 = ';' ' M7 = ';' ' M8 = ';' '
M9 = ';' ' M10 = '}.
compute mlab = {mlab; ' M11 = ';' ' M12 = ';' '
M13 = ';' ' M14 = ';' ' M15 = '}.
compute mname =
name(1,(2+nxd):(1+nxd+nm)).
do if (ncovs > 0).
compute c =
dd(:,(2+nxd+nm):(1+nxd+nm+ncovs)).
compute x = {x, c}.
compute xdata = x.
compute cdata = c.
compute xname = {xname, cname}.
end if.
compute nms={yname; t(mname);
t(xname(1,1:(ncol(xname)-ncovs)))}.
do if (dumok = 1).
compute nms={yname; t(mname); name(1,2)}.
end if.
do if (nm < 16).
do if (dumok = 1).
print dummat/title = 'Coding of categorical X
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
compute
boots=make((samples+1),((nx+1)*nm),-999).
do if (errs = 0 and criterr = 0).
loop k = 1 to (samples+1).
do if (k = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef=0.
do if (k > 1).
compute u=trunc(uniform(n,1)*n)+1.
compute xdata = x(u,:).
compute mdata = m(u,:).

```

```

compute ydata = y(u,:).
do if (ncovs > 0).
compute cdata = c(u,:).
end if.
end if.
loop i = 1 to nm.
compute start = 1+(i-1)*nx.
compute xmat = {ones, xdata}.
do if (covmy=2).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = mdata(:,i).
compute amat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*amat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-(((1-r2)*(n-1))/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute r2cha = r2.
compute adjr2cha = adjr2.
do if (ncovs > 0 and covmy <> 2).
compute xmat = {ones, cdata}.
compute atmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*atmat)**2).
compute r2step1 = (sstotal-ssresid2)/sstotal.
compute r2cha = r2-r2step1.
compute adjr2cha = adjr2- (1-(((1-r2step1)*(n-
1))/(n-ncol(xmat)))).
compute xmat = {ones, xdata}.
end if.
compute bootcoef={bootcoef,t(amat)}.
compute r2b(i,1)=adjr2cha.
do if (k = 1).

```

```

compute hcinvtX=inv(t( xmat )' xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,1)*hcinvtX*t( xmat
(i3,1)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvtX*t( xmat )' xmat
*hcinvtX
.
compute tratio = amat&/sqrt(diag(covmat)).
compute rnms = mname(1,i).
print rnms/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.
compute lmat = ident(nrow(amat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*amat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*amat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/'format = F10.4
/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'.

```



```

compute op = {amat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*(xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute cov(start:(start+(nx-
1)),start:(start+(nx-
1)))=covmat(2:(ncol(covmat)-(ncovs*(covmy <>
2))),2:(ncol(covmat)-(ncovs*(covmy <> 2)))).
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns(start:(start+(nx-
1)),1)=amat(2:(1+nx)).
end loop.
compute xmat = {ones,mdata,xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (k = 1).
print yname/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.

```

```

do if (ovals = 2).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:'/cnames = nmsd/format = F9.2.
end if.
end if.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(yamat)/n)).
compute LL3 = yamat*&ln(pt2)+(1-yamat)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= yamat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).

```

```

compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute bootcoef={bootcoef,t(bmat)}.
compute LL1 = LL2.
compute covmat=varb.
end if.
do if (ovals <> 2).
compute bmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*bmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-(((1-r2)*(n-1))/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute bootcoef={bootcoef,t(bmat)}.
end if.
do if (k = 1).
do if (ovals = 2).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.

```

```

compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).
compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms2/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute
opdirsum=outp((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
end if.
do if (ovals <> 2).
.
compute hcinvXtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtXt*( xmat
(i3,:)).

```

```

end loop.
loop i3=1 to k3.
compute xmat(:,i3) = ( resid(:,ncol( resid
))&/(1-h))&* xmat(:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat(:,i3) = sqrt( mse )&* xmat(:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(bmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/(ncol(xmat)-1).
compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat((2+nmeds):(2+nmeds+(nx-1)),:)=
lmat2.
compute fratiodr =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/nx.
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute df2=(n-ncol(xmat)).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), df2, pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = bmat&/sqrt(diag(covmat)).
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute op = {bmat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat)))) }.

```

```

compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp))))*(xp2/(temp-
(2/3)+(.11/temp)))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms2/format =
F10.4.
compute
opdirsum=op((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
compute xmat = {ones,mdata}.
do if (ncovs > 0).
compute xmat = {xmat, cdata}.
end if.
compute btmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*btmat)**2).
compute r2direct = r2-((sstotal-
ssresid2)/sstotal).
compute pfr = 1-fcdf(fratiodr,(nx),(n-1-nx-nm-
ncovs)).
compute r2direct = {r2direct, fratiodr, nx, (n-1-
nx-nm-ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms2/cnames=rnms2/form
at= F10.4.
end if.
end if.

```

```

do if (k = 1 and toteff = 1).
print yname/title = '*****'
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:.'/format = A8.
compute xmat = {ones, xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(ymat)/n)).
compute LL3 = ymat*&ln(pt2)+(1-ymat*)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= ymat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.

```

```

compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute LL1 = LL2.
compute covmat=varb.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(bmat)-
1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms = {'Constant'; t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).

```



```

compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute optotsum=outp(2:(1+nx),:).
end if.
do if (ovals <> 2).
compute cmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*cmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).

.

compute hcinvXtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
)))/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).

```

```

loop i3=1 to k3.
compute xmat(:,i3) = sqrt( mse )&* xmat(:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(cmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = cmatrix/sqrt(diag(covmat)).
compute op = {cmatrix, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute optotsum=op(2:(1+nx),:).
compute ssresid2 = ssresid.
compute r2total = r2.

```

```

compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat(2:(1+nx),:) = lmat2.
compute fratio3 =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/nx.
do if (ncovs > 0).
compute xmat = {ones, cdata}.
compute ctmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*ctmat)**2).
compute r2total = r2-((sstotal-ssresid2)/sstotal).
end if.
compute pfr = 1-fcdf(fratio3,(nx),(n-1-nx-
ncovs)).
compute r2tot2 = {r2total, fratio3, nx, (n-1-nx-
ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns((nrow(mns)-
nm+1):nrow(mns),1)=bmat(2:(1+nm),1).
compute tmp = 1.
loop i = 1 to (nm*nx).
compute
boots(k,i)=mns(i,1)*mns(((nm*nx)+tmp),1).
do if ((i/nx)=trunc(i/nx)).
compute
boots(k,((nm*nx)+tmp))=r2b(tmp,1)*mns(((nm*
nx)+tmp),1).
compute tmp = tmp+1.
end if.
end loop.
do if (savboot = 1 and k > 1).

```

```

compute bootstrp((k-1),:)=bootcoef.
end if.
end loop.
do if (method = 1).
compute llci=make(1,ncol(boots),-999).
compute ulci=make(1,ncol(boots),-999).
compute indeff = t(boots(1,1:(nm*nx))).
compute indeff22=t(boots(1,:)).
compute omni = t(boots(1,((nm*nx)+1) :
ncol(boots))).
do if (nrow(boots) > 1).
compute boots = boots(2:nrow(boots),:).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
loop i = 1 to ncol(boots).
.
compute temp = boots(:,i).
compute temp(GRADE( boots(:,i) )) = boots(:,i).
compute badlo = 0.
compute badhi = 0.
do if ( indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(indeff22(i,1)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.

```

```

do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,i)=llcit.
compute ulci(1,i)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute llci = t(llci).
compute ulci= t(ulci).
compute llci={llci,ulci}.
end if.
end if.
do if (method = 2).
compute tmp=1.
loop i = 1 to (nm*nx).
compute
indeff(i,1)=mns(i,1)&*mns(((nm*nx)+tmp),1).

```

```

do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = sqrt(-
2*ln(uniform(reps,nrow(cov))))&*cos((2*3.1415
9265358979)*uniform(reps,nrow(cov))).
compute boots=boots*chol(cov).
loop i = 1 to ncol(boots).
compute boots(:,i)=boots(:,i)+mns(i,1).
end loop.
compute tmp=1.
loop i = 1 to (nm*nx).
compute
boots(:,i)=boots(:,i)&*boots(:,((nm*nx)+tmp)).
compute temp = boots(:,i).
compute temp(GRADE(boots(:,i))) = boots(:,i).
compute boots(:,i) = temp.
do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = boots(:,1:(nm*nx)).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
compute llci =
t(((boots(cilow,:));(boots(cihigh,:)))).
end if.
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
compute rnmssum=rnms(2:(1+nx),1).
do if (ovals <> 2).
print optotsum/title = 'Relative total effects of X
of Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.

```

```

print r2tot2/title = 'Omnibus test of total effect of
X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print optotsum/title = 'Relative total effects of X
on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/'space=0.
end if.
compute
rnmssum=rnms2((2+nmeds):(1+nmeds+nx),1).
do if (ovals <> 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.
print r2direct/title = 'Omnibus test of direct
effect of X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/'space=0.
do if (omnibus = 1 and (method = 1 or method
= 3)).
compute indlbs = {indlbs; 'Omnibus'}.
end if.
loop i = 1 to nm.
compute op = mname(1,i).
print op/title = 'Relative indirect effect(s) of X on
Y through:.'/format = A8/space=0.
do if (method = 2).
compute clbs = {'Effect', 'SE MC', 'MC LLCI',
'MC ULCI'}.
compute op = {indeff, bootse(1:(nm*nx)), lici}.
end if.

```

```

do if (method = 1).
compute clbs = {'Effect', 'SE(boot)', 'LLCI',
'ULCI'}.
compute op = indeff.
do if (samples > 0).
compute op = {indeff, bootse(1:(nm*nx),:),
llci(1:(nm*nx),:)}).
end if.
end if.
compute temp = op((((i-1)*nx)+1):(nx*i),:).
do if ((method = 1) and omnibus = 1).
do if (samples = 0).
compute temp={temp;omni(i,1)}.
end if.
do if (samples > 0).
compute temp = {temp; omni(i,1),
bootse(((nm*nx)+i),:), llci(((nm*nx)+i),:)}).
end if.
end if.
print temp/title = '/rnames = indlbs/cnames =
clbs/format = F10.4 /space=0.
print/title = '-----'.
end loop.
end if.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
END IF.
DO IF (mdpbe=1).
compute nm={yname,xname,mnames}.
do if (ncovs > 0).

```



```

compute
nm={yname,cnames,xname,mnames}.
end if.

compute tpx=xname.
compute x = dd(:,2:ncol(dd)).
compute tpy = yname.
compute tempy= dd(:,1).
compute n = nrow(dd).
compute nx=1.
compute criterr=0.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute ddd1 = {'int_1', 'int_2', 'int_3', 'int_4',
'int_5', 'int_6', 'int_7', 'int_8', 'int_9'}.
compute dumok = 0.
compute mcfoc=mcx.
compute mcmmod=mcm.
compute mcloc=(mcfoc>0).
compute con = make(n,1,1).
compute ncovs=ncol(x)-2.
do if (mcfoc > 0 or mcmmod > 0).
compute temp = dd.
compute temp(GRADE(dd(:,(ncol(dd)-
mcloc))),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,ncol(dd)-
mcloc)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute toosmall=rsum(nnvls < 2).
compute mnvls = cmin(t(nnvls)).
do if ((rsum(nnvls < 2)) > 0).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=37.
end if.
do if (nvls > 10).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=36.

```

```

end if.
do if (criterr=0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,ncol(dd)-mcloc).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,ncol(dd)-mcloc) <> nnvls((temp-
1),1)).
compute nnvls(temp,1)=dd(i,ncol(dd)-mcloc).
compute temp = temp+1.
end if.
end loop.
compute dummy = dummy(:,2:ncol(dummy)).
do if (mcfoc=4 or mcmmod=4).
compute minus1=make(1,ncol(dummy),-1).
loop k = 1 to n.
do if (rsum(dummy(k,:)) = 0).
compute dummy(k,:)=minus1.
end if.
end loop.
end if.
do if (mcfoc = 2 or mcfoc = 3) or (mcmmod = 2 or
mcmmod = 3)).
loop k = 1 to n.
do if (rsum(dummy(k,:)) > 0).
loop i = 1 to ncol(dummy).
do if (dummy(k,i) = 0).
compute dummy(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcfoc = 3 or mcmmod=3).
compute conmat1={-8,1,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1,1; 0,0,0,0,0,-
3,1,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.

```

```

loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,:)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute
dummy(k,:)=conmat1((rsum(dummy(k,:))+1),:).
end loop.
end if.
end if.
compute nx = ncol(dummy).
compute xname = ddd(1,1:nx).
compute xdata = dummy.
compute xname = ddd(1,1:nx).
compute xname2={nm(1,(ncol(nm)-mcloc)),
xname}.
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcfoc = 2 or mcmmod=2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcfoc = 3 or mcmmod=3).
compute dummat=conmat1.
end if.
do if (mcfoc = 4 or mcmmod = 4).
compute dummat(1,:)=minus1.
end if.
compute dummat={nnvls, dummat}.
end if.
end if.
do if (criterr=0).
compute y = dd(:,1).
compute ovals = ncol(design(dd(:,1))).

```

```

compute itprob = 0.
do if (ovals = 2).
compute omx = cmax(y(:,1)).
compute omn = cmin(y(:,1)).
compute y(:,1) = (y(:,1) = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute sstotal = csum((y-(csum(y)/n))&**2).
compute outv = t(nm(1,1)).
compute mdtr = nm(1,ncol(dd)).
compute fciv = nm(1,(ncol(dd)-1)).
do if (mcfoc <> 0 or mcmod <> 0).
compute inter=make(n,nx,0).
loop i = 1 to nx.
compute inter(:,i)=dummy(:,i)&*dd(:,ncol(dd)-
(1-mcloc)).
end loop.
do if (ncovs = 0).
compute x={con,dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
do if (ncovs > 0).
compute
x={con,dd(:,2:(1+ncovs)),dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={t(nm(1,2:(1+ncovs)));
nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
end if.
compute xmns = csum(x)/n.
compute focvals = {1,0,0}.
compute highwarn = 0.
compute lowwarn = 0.

```

```

do if (ncol(x) > (2*nx+2)).
compute covmns = {1, xmns(1,2:(ncol(x)-1-
(2*nx)))}.
compute focvals = {covmns,0,0}.
end if.
compute dfres = n-ncol(x).
do if (ovals = 2).
compute pt2 = make(n,1,(csum(y)/n)).
compute LL3 = y*&ln(pt2)+(1-y)&ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(x),1,0).
compute LL1 = 0.

.

compute pt1lp= pt1.
compute xlp= x.
compute ylp= y.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).

```

```

compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
else if (ovals <> 2).
compute invXtX=inv(t(x)*x).
compute b = invXtX*t(x)*y.
compute resid = y-(x*b).
compute ssresid=csum(resid*resid).
compute msresid=ssresid/dfres.
compute varb = msresid*invXtX.
compute
seint=sqrt(varb(nrow(varb),nrow(varb))).
compute k3 = nrow(b).
do if (hc3 = 1).
compute xhc=x.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)= xhc(i3,:)*invXtX*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).
end loop.
compute varb=(invXtX*t(xhc)*xhc*invXtX).
end if.
compute seb = sqrt(diag(varb)).
compute r2 = 1-(ssresid/sstotal).

```

```

do if (mcfoc <> 0 or mcmod <> 0).
compute ytp=y.
compute xtp=x(:,1:(ncol(x)-nx)).
compute btp = inv(t(xtp)*xtp)*t(xtp)*ytp.
compute residt = ytp-(xtp*btp).
compute ssresidt=csum(residt*residt).
compute r2noint = 1-(ssresidt/sstotal).
end if.
compute pr = ncol(x)-1.
compute lmat = ident(nrow(b)).
compute lmat = lmat(:,2:ncol(lmat)).
compute f =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
pr).
compute pf = 1-fcdf(f,pr,dfres).
compute pf = {sqrt(r2),r2,msresid,f,pr,dfres,pf}.
compute xd=abs(xp2).
end if.
do if (itprob = 0).
compute tstat = b&/seb.
do if (ovals <> 2).
compute p = 2*(1-tcdf(abs(tstat), dfres)).
else if (ovals = 2).
compute p = 2*(1-cdfnorm(abs(tstat))).
end if.
compute outp = {b,seb,tstat,p}.
compute nms = {'constant'; nms; 'interact'}.
do if (ovals <> 2).
compute tval = (dfres* (exp((dfres-
(5/6))*((xd/(dfres-(2/3)+(.11/dfres)))*(xd/(dfres-
(2/3)+(.11/dfres)))))-1)).
compute outp={outp,(b-
sqrt(tval)&*seb),(b+sqrt(tval)&*seb)}.
end if.
compute bb = tval.
do if (dumok = 1).
do if (mcfoc > 0).
print dummat/title = 'Coding of categorical X
variable for analysis:' /cnames = xname2/format
= F5.2.

```

```

else if (mcmod > 0).
print dummat/title = 'Coding of categorical M
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
print yname/title =
*****
*****'/rlabels = 'Outcome: '/format
= A8.
do if (ovals = 2).
compute nmsd = {outv, 'Analysis'}.
print rcd/title = 'Coding of binary Y for
analysis: '/cnames = nmsd/format = F10.4.
compute outp = {outp,(outp(:,1)-
sqrt(tval)&*outp(:,2)),(outp(:,1)+sqrt(tval)&*outp
(:,2))}.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
print outp/title = 'Logistic Regression
Model'/rnames = nms/clabels 'Coeff' 'se' 'Z' 'p'
'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
compute varbtmp=varb.
compute btmp=b.

```



```

compute LL2f=LL2.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=x(:,1:(ncol(x)-nx)).
compute bt1 = make(ncol(xlp),1,0).

.

compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)*&ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.

```

```

compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL2f.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(LLdiff,nx).
compute rcha={LLdiff,nx,pchi}.
end if.
do if (ovals <> 2).
print pf/title = 'Model Summary'/clabels = 'R' 'R-
sq' 'MSE' 'F' 'df1' 'df2' 'p'/format F10.4.
compute lmat=make(nrow(b),1,0).
compute lmat(nrow(lmat),1)=1.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
1).
compute
rcha=((b(nrow(b),1)/seint)*(b(nrow(b),1)/seint))*
(1-r2)/dfres.
compute rcha = {rcha, fcha, 1, dfres,
outp((pr+1),4)}.
print outp/title = 'Model'/rnames = nms/clabels
'coeff' 'se' 't' 'p' 'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
do if (mcfoc > 0 or mcmod > 0).
compute rcha=r2-r2noint.
compute lmat=make((nrow(b)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
nx).
compute pvalr2c=1-fcdf(fcha,nx,dfres).

```

```

compute rcha = {rcha, fcha, nx, dfres,pvalr2c}.
end if.
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute intkey = {'a', 'b', 'c', 'd', 'e'}.
loop i = 1 to nx.
compute intkey={intkey; ddd1(1,i), ' : ', ddd(1,i),
' X ', nm(1,ncol(dd)-(1-mcloc))}.
end loop.
compute intkey=intkey(2:nrow(intkey),:).
print intkey/title='Product terms key:'/format =
a8.
end if.
do if (ovals <> 2).
print rcha/title = 'R-square increase due to
interaction:.'/clabels 'R2-chng' 'F' 'df1' 'df2'
'p'/format F10.4.
else if (ovals = 2).
print rcha/title = 'Likelihood ratio test for
interaction:.'/clabels 'Chi-sq' 'df' 'p'/format F10.4.
end if.
print/title =
*****
*****'.
compute mdvar = x(:,(ncol(x)-1)).
do if (mcfoc = 0 and mcmmod = 0).
compute g1 = b((ncol(x)-2),1).
compute g3 = b(ncol(x),1).
compute vg1 = varb((ncol(x)-2),(ncol(x)-2)).
compute vg3 = varb(ncol(x),ncol(x)).
compute covg1g3 = varb((ncol(x)-2), ncol(x)).
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute mdvar = x(:,(ncol(x)-(2*nx))).
end if.
compute mdmin = cmin(mdvar).
compute mdmax = cmax(mdvar).
compute fvar = x(:,(ncol(x)-2)).
compute nval = ncol(design(mdvar)).
compute fvmin = cmin(fvar).

```

```

compute fvmax = cmax(fvar).
do if (mmodval = 999).
compute mnmd = csum(mdvar)/n.
compute tmp = make(n,1,mnmd).
compute sdmd = sqrt(csum(((mdvar-
tmp)**2))/(n-1)).
compute probeval = {mnmd-sdmd; mnmd;
mnmd+sdmd}.
do if (probeval(1,1) < mdmin).
compute lowwarn = 1.
compute probeval(1,1)=mdmin.
end if.
do if (probeval(nrow(probeval),1) > mdmax).
compute highwarn = 1.
compute probeval(nrow(probeval),1)=mdmax.
end if.
do if (quantile = 1).
compute tmp = mdvar.
compute tmp(GRADE(mdvar(:,1)),:) = mdvar.
compute mdvar = tmp.
compute
probeval={mdvar(trunc(0.1*n),1);mdvar(trunc(0
.25*n),1);mdvar(trunc(0.5*n),1);mdvar(trunc(0.7
5*n),1);mdvar(trunc(0.9*n),1)}.
end if.
end if.
do if (nval = 2).
compute probeval = make(2,1,0).
compute probeval(1,1)=cmin(mdvar).
loop i = 1 to n.
do if (mdvar(i,1) <> probeval(1,1)).
compute probeval(2,1) = mdvar(i,1).
BREAK.
end if.
end loop.
end if.
do if (mmodval <> 999).
compute probeval = mmodval.
end if.
compute outp = make(nrow(probeval),7,0).

```

```

do if (mcfoc > 0 or mcmod > 0).
compute focvals=make(1,ncol(x)+1,1).
do if (mcfoc > 0).
print/title = 'Conditional effect of X on Y at
values of the moderator:'.
print/title = ' '/space=0.
compute rnn2=mdtr.
compute matt=make(nx,6,0).
end if.
loop jj=1 to nrow(probeval).
do if (mcfoc > 0).
loop ii=1 to nx.
compute g1=b((ncol(x)-(2*nx)+ii),1).
compute g3=b((ncol(x)-nx+ii),1).
compute vg1=varb((ncol(x)-(2*nx)+ii),(ncol(x)-
(2*nx)+ii)).
compute vg3=varb((ncol(x)-nx+ii),(ncol(x)-
nx+ii)).
compute covg1g3=varb((ncol(x)-
(2*nx)+ii),(ncol(x)-nx+ii)).
compute x2 = probeval(jj,1).
compute w1 = g1+g3*x2.
compute varw1 =
vg1+(2*x2*covg1g3)+((x2*x2)*vg3).
compute sew1 = sqrt(varw1).
compute t1 = w1/sew1.
do if (ovals <> 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-tcdf(abs(t1), dfres)).
compute cnms = {'Coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)= {w1,sew1,t1,p,llci,ulci}.
else if (ovals = 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-cdfnorm(abs(t1))).
compute cnms = {'Coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)= {w1,sew1,t1,p,llci,ulci}.

```

```

end if.
end loop.
compute rnms=t(xname).
compute mdvalpr=probeval(jj,1).
print mdvalpr/title = 'Moderator
value: '/rnames=rnn2/format= F10.4 /space=0.
print matt/title='
'/cnames=cnms/rnames=rnms/format= F10.4
/space=0.
compute xprob=x(:,(ncol(x)-(2*nx)))-mdvalpr.
loop kk = 1 to nx.
compute xprob={xprob,
(xprob(:,1)&*x(:,(ncol(x)-(2*nx)+kk)))}.
end loop.
compute xprob={x(:,1:((ncol(x)-(2*nx))-
1)),xprob}.
do if (ovals <> 2).
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute residc = y-(xprob*bmultc).
compute ssresidc=csum(residc&*residc).
compute r2c = r2-(1-(ssresidc/sstotal)).
compute fcha2=(dfres*r2c)/(nx*(1-r2)).
do if (hc3 = 1).
loop kk = 1 to nx.
compute xprob={xprob, x(:,(ncol(x)-
(2*nx)+kk))}.
end loop.
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute k3 = nrow(bmultc).
compute xhc=xprob.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)=
xhc(i3,1)*inv(t(xprob)*xprob)*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).

```

```

end loop.
compute
varbc=(inv(t(xprob)*xprob)*t(xhc)*xhc*inv(t(xprob)*xprob)).
compute lmat=make((nrow(bmultc)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha2 =
(t(t(lmat)*bmultc)*inv(t(lmat)*varbc*lmat)*((t(lmat)*bmultc)))/nx).
end if.
compute pvalr2cc=1-fcdf(fcha2,nx,dfres).
compute rcha2 = {r2c, fcha2, nx,
dfres,pvalr2cc}.
print rcha2/title = 'Test of equality of conditional
means at this value of the moderator'/clabels
'R2-chng' 'F' 'df1' 'df2' 'p'/format F10.4.
else if (ovals = 2).
compute btmp=b.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=xprob.
compute bt1 = make(ncol(xlp),1,0).

.
compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.

```

```

end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)*&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*ulp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL4.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(lldiff,nx).
compute rcha2 = {LLdiff, nx,pchi}.
print rcha2/title = 'Test of equality of log odds
conditioned on this moderator value'/clabels
'Chi-sq' 'df' 'p'/format F10.4.
end if.
end if.
compute
tttt=make(nrow(dummat),1,probeval(jj,1)).
compute tttt={tttt,dummat(:,2:ncol(dummat))}.
loop kkk=1 to nx.
compute tttt={tttt,tttt(:,1)*tttt(:,1+kkk)}.
end loop.
compute ones=make(nrow(dummat),1,1).
do if (ncol(x) > (2*nx+2)).

```



```

compute
covmnmat=make(nrow(tttt),ncol(covmns),0).
loop kkk=1 to nrow(tttt).
compute covmnmat(kkk,:)=covmns.
end loop.
compute tttt={covmnmat,tttt}.
else.
compute tttt={ones,tttt}.
end if.
compute focvals={focvals;dummat(:,1),tttt}.
do if (mcfoc > 0).
compute yhat={dummat(:,1),(tttt*b)}.
compute cmnms={fciv, 'yhat'}.
do if (ovals <> 2).
print yhat/title = 'Estimated conditional means
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (ovals = 2).
print yhat/title = 'Estimated conditional log odds
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (jj <> nrow(probeval)).
print/title='-----'/space=0.
end if.
end if.
end loop.
compute focvals=focvals(2:nrow(focvals),:).
compute yhat=focvals(:,2:ncol(focvals))*b.
compute
focvals={focvals(:,1),focvals(:,(ncol(focvals)-
(2*nx))),yhat}.
compute cnms={fciv,mdtr,'yhat'}.
do if (mcmod > 0).
compute cnms={mdtr,fciv,'yhat'}.
end if.
end if.
do if (mcmod > 0).
compute outp=make((nx+1),7,0).

```

```

compute bcatm={b((ncol(x)-
(2*nx)),1);b((ncol(x)-nx+1):ncol(x),1)}.
compute outp(:,1)=dummat(:,1).
compute bcatcov=varb((ncol(x)-
nx):ncol(x),(ncol(x)-nx):ncol(x)).
compute bcatcov(1,1)=varb((ncol(x)-
(2*nx)),(ncol(x)-(2*nx))).
compute
bcatcov(2:nrow(bcatcov),1)=varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx)).
compute
bcatcov(1,2:nrow(bcatcov))=t(varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx))).
loop i = 1 to nrow(dummat).
compute
catmval={1,dummat(i,2:ncol(dummat))}.
compute condeff=catmval*bcatm.
compute
condse=sqrt(catmval*bcatcov*t(catmval)).
compute outp(i,2:3)={condeff,condse}.
end loop.
compute outp(:,4)=outp(:,2)&/outp(:,3).
compute outp(:,5) = 2*(1-tcdf(abs(outp(:,4)),
dfres)).
compute outp(:,6) = (outp(:,2)-
sqrt(tval)&*outp(:,3)).
compute outp(:,7) =
(outp(:,2)+sqrt(tval)&*outp(:,3)).
do if (ovals <> 2).
compute cnmms = {xname2(1,1), 'coeff', 'se',
't', 'p', 'LLCI', 'ULCI'}.
end if.
do if (ovals = 2).
compute outp(:,5) = 2*(1-
cdfnorm(abs(outp(:,4)))).
compute cnmms = {xname2(1,1), 'coeff', 'se',
'Z', 'p', 'LLCI', 'ULCI'}.
end if.

```

```

print outp/title = 'Conditional Effect of Focal
Predictor in Groups Defined by the Moderator
Variable:.'/cnames=cnmms/format = F10.4.
end if.
do if (nval > 2 and (mmodval = 999) and
mcmmod = 0).
do if (quantile <> 1).
print/title = 'Moderator values are the sample
mean and plus/minus one SD from mean'.
else.
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
end if.
do if (highwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD above the mean was'.
print/title = 'replaced with the maximum
because one SD above the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
do if (lowwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD below the mean was'.
print/title = 'replaced with the minimum
because one SD below the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
end if.
do if (nval = 2 and (mmodval = 999)).
print/title = 'The moderator variable is
dichotomous'.
end if.
do if (mmodval <> 999 and mcmmod = 0 and
((mmodval < mdmin) or (mmodval > mdmax))).
print/title = 'Warning: Moderator value specified
is outside of the range of the data'.
end if.

```

```

do if (plot = 1).
do if (mcfoc = 0 and mcmmod=0).
compute cnms = {t(nms((ncol(x)-2):(ncol(x)-
1),1)), 'yhat'}.
end if.
do if (ovals = 2).
compute prob =
exp(focvals(:,3))/(1+exp(focvals(:,3))).
compute focvals = {focvals, prob}.
compute cnms = {cnms, 'prob'}.
end if.
print/title =
*****
*****!

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (ovals = 2).
do if (mcfoc=0).
print/title = 'DATA LIST
FREE/Technological_Advan Opra_Pract
logodds prob.' /space=1.
end if.
do if (mcmmod=0).

```

```

print/title = 'DATA LIST FREE/Opra_Pract
Technological_Advan logodds prob.'
/space=1.
end if.
else if (ovals <> 2).
do if (mcfoc=0).
print/title = 'DATA LIST
FREE/Technological_Advan Opra_Pract
Operational_Performance.' /space=1.
end if.
do if (mcmod=0).
print/title = 'DATA LIST FREE/Opra_Pract
Technological_Advan
Operational_Performance.' /space=1.
end if.
end if.
print/title = 'BEGIN DATA.' /space=0.
print focvals/title = ' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (ovals <> 2 and mcmod=0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH Operational_Performance BY
Opra_Pract.' /space=0.
end if.
do if (ovals <> 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
Operational_Performance BY
Technological_Advan.' /space=0.
end if.
do if (ovals = 2 and mcmod=0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Opra_Pract.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Opra_Pract.' /space=0.
end if.
do if (ovals = 2 and mcfoc=0).

```

```

print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
logodds BY Technological_Advan.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
prob BY Technological_Advan.' /space=0.
end if.
do if (ncovs > 0).
print/title = 'NOTE: For data above, covariates
are set to their sample means.'.
end if.
end if.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
end if.
END IF.
end if.
do if (bad > 0).
compute note(notes,1) = 9.
compute notes = notes + 1.
end if.
print/title = '***** ANALYSIS
NOTES AND WARNINGS
*****'.
loop i = 1 to errs.
do if (runerrs(i,1) = 1).
print/title = 'ERROR: One of your declared
mediators is dichotomous. This procedure
can"t be used.'.
end if.
do if (runerrs(i,1) = 2).

```

```
print/title = 'ERROR: For model 6, this
procedure limits the number of mediators to
four.'.
end if.
do if (runerrs(i,1) = 3).
print/title = 'ERROR: For models 1, 2, and 3,
only a single variable can be listed in the M
list.'.
end if.
do if (runerrs(i,1) = 4).
print/title = 'ERROR: You requested a model
involving W but did not provide a valid W
variable name.'.
end if.
do if (runerrs(i,1) = 5).
print/title = 'ERROR: You requested a model
involving Z but did not provide a valid Z
variable name.'.
end if.
do if (runerrs(i,1) = 6).
print/title = 'ERROR: You requested a model
involving Q but did not provide a valid Q
variable name.'.
end if.
do if (runerrs(i,1) = 7).
print/title = 'ERROR: You requested a model
involving V but did not provide a valid V
variable name.'.
end if.
do if (runerrs(i,1) = 8).
print/title = 'ERROR: You specified a W
variable for a model that does not need it'.
end if.
do if (runerrs(i,1) = 9).
print/title = 'ERROR: You specified a Z variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 10).
print/title = 'ERROR: You specified a Q variable
for a model that does not need it'.
```

```

end if.
do if (runerrs(i,1) = 11).
print/title = 'ERROR: You specified a V variable
for a model that does not need it.'.
end if.
do if (runerrs(i,1) = 12).
print/title = 'ERROR: The variable specified for
W has already been assigned.'.
end if.
do if (runerrs(i,1) = 13).
print/title = 'ERROR: The variable specified for
Z has already been assigned.'.
end if.
do if (runerrs(i,1) = 14).
print/title = 'ERROR: The variable specified for
Q has already been assigned.'.
end if.
do if (runerrs(i,1) = 15).
print/title = 'ERROR: The variable specified for
V has already been assigned.'.
end if.
do if (runerrs(i,1) = 16).
print/title = 'ERROR: You did not provide a
valid Y variable name.'.
end if.
do if (runerrs(i,1) = 17).
print/title = 'ERROR: The variable specified for
Y has already been assigned.'.
end if.
do if (runerrs(i,1) = 18).
print/title = 'ERROR: Model 6 requires more
than one mediator.'.
end if.
do if (runerrs(i,1) = 19).
print/title = 'ERROR: You have not specified a
valid model number.'.
end if.
do if (runerrs(i,1) = 20).
print/title = 'ERROR: At least one and only one
variable must be listed for X.'.

```



```
end if.  
do if (runerrs(i,1) = 21).  
print/title = 'ERROR: At least one and only one  
variable must be listed for Y.'.  
end if.  
do if (runerrs(i,1) = 22).  
print/title = 'ERROR: Iteration didn"t converge  
to a solution. Interpret results with caution.'.  
end if.  
do if (runerrs(i,1) = 23).  
print/title = 'ERROR: You specified a clustering  
variable that does not exist in your variable  
list.'.  
end if.  
do if (runerrs(i,1) = 24).  
print/title = 'ERROR: You specified a clustering  
variable that has already been assigned.'.  
end if.  
do if (runerrs(i,1) = 25).  
print/title = 'ERROR: One or more of your M  
variables is not listed in the variables list.'.  
end if.  
do if (runerrs(i,1) = 26).  
print/title = 'ERROR: A maximum of 20 cluster  
units is allowed. Use multilevel modeling  
instead.'.  
end if.  
do if (runerrs(i,1) = 27).  
print/title = 'ERROR: One of the variables in  
your model is a constant.'.  
end if.  
do if (runerrs(i,1) = 28).  
print/title = 'ERROR: Dichotomous Y is not  
permitted with WS option.'.  
end if.  
do if (runerrs(i,1) = 29).  
print/title = 'ERROR: Insufficient number of  
variables in vars= list when using WS option.'.  
end if.  
do if (runerrs(i,1) = 30).
```

```
print/title = 'ERROR: Too many variables in
vars= list when using WS option. Covariates
not allowed.'.
end if.
do if (runerrs(i,1) = 31).
print/title = 'ERROR: mmodval and wmodval
can"t both be set to zero with contrast option.'.
end if.
do if (runerrs(i,1) = 32).
print/title = 'ERROR: You did not provide a
valid X variable name.'.
end if.
do if (runerrs(i,1) = 33).
print/title = 'ERROR: PROCESS requires all
variable names to be eight characters or
fewer.'.
print/title = '      Please shorten variable names
and reexecute.'/space=0.
end if.
do if (runerrs(i,1)=34).
print/title = 'ERROR: X and M cannot both be
specified as multicategorical.'.
end if.
do if (runerrs(i,1)=35).
print/title = 'ERROR: Multicategorical variable
specification available only for models 1 and
4.'.
end if.
do if (runerrs(i,1)=36).
print/title = 'ERROR: Categorical variables
cannot have more than 10 categories.'.
end if.
do if (runerrs(i,1)=37).
print/title = 'ERROR: Each group must have at
least two cases.'.
end if.
do if (runerrs(i,1)=38).
print/title = 'ERROR: Cluster option not
available with multicategorical variables.'.
end if.
```

```

do if (runerrs(i,1)=39).
print/title = 'ERROR: A maximum of 10
variables is allowed in the M= list.'.
end if.
do if (runerrs(i,1) = 41).
print/title = 'ERROR: MCM option not permitted
in model 4.'.
end if.
end loop.
do if (errs = 0).
do if (boot > 1 or mc > 0).
do if (bconoff = 1 and boot > 1).
print boot/title = 'Number of bootstrap samples
for bias corrected bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (bconoff = 0 and boot > 1).
print boot/title = 'Number of bootstrap samples
for percentile bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (mc > 1).
print mc/title = 'Number of samples for Monte
Carlo confidence intervals:'/format = F8.0.
end if.
do if (booterr = 1).
compute badend = badend(1,2:ncol(badend)).
print badend/title = 'WARNING: Bootstrap CI
endpoints below not trustworthy. Decrease
confidence or increase bootstraps'/format =
F10.4.
end if.
end if.
print conf/title = 'Level of confidence for all
confidence intervals in output:'/format = F8.2.
do if ((center = 1 or ws=1) and (ncol(centvar) >
1)).
compute centvar = centvar(1,2:ncol(centvar)).

```

```

print centvar/title = 'NOTE: The following
variables were mean centered prior to
analysis:.'/format = a8.
end if.
loop i = 1 to notes.
do if (note(i,1) = 1).
print/title = 'NOTE: Confidence level restricted
to between 50 and 99.9999%. 95% confidence
is provided in output'.
end if.
do if (note(i,1) = 2).
print/title = 'NOTE: Effect size measures for
indirect effects are not available for models
with dichotomous outcomes'.
end if.
do if (note(i,1) = 3).
print/title = 'NOTE: All standard errors for
continuous outcome models are based on the
HC3 estimator'.
end if.
do if (note(i,1) = 6).
print/title = 'NOTE: The number of bootstrap
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 7).
print/title = 'NOTE: The Johnson-Neyman
method is available only for Models 1 and 3'.
end if.
do if (note(i,1) = 8).
print/title = 'NOTE: The Johnson-Neyman
method cannot be used with a dichotomous
moderator'.
end if.
do if (note(i,1)=9).
print bad/title = 'NOTE: Some bootstrap
samples had to be replaced. The number of
such replacements was:'.
end if.
do if (note(i,1) = 11).

```

```

print nmiss/title = 'NOTE: Some cases were
deleted due to missing data. The number of
such cases was:'.
end if.
do if (note(i,1) = 12).
print /title = 'NOTE: Monte Carlo method
available only for models 4 and 5.
Bootstrapping was used instead.'.
end if.
do if (note(i,1) = 13).
print/title = 'NOTE: The number of Monte Carlo
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 19).
print/title = 'NOTE: Effect sizes not available for
within-subject analyses.'.
end if.
do if (note(i,1) = 16).
print/title = 'NOTE: Normal theory tests not
available for within-subject analyses.'.
end if.
do if (note(i,1) = 17).
print/title = 'NOTE: Monte Carlo confidence
intervals not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 18 and warnrep=0).
print/title = 'WARNING: You have requested
OLS estimation with a dichotomous criterion.'.
print/title = 'Interpret model coefficients and
inferential statistics with caution.'/space=0.
compute warnrep=1.
end if.
do if (note(i,1) = 20).
print/title = 'NOTE: Saving of bootstrap
estimates not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 22).

```

```
print/title = 'NOTE: Effect size option with
covariates requires covariates in models of M
and Y.'.
end if.
do if (note(i,1) = 23).
print/title = 'NOTE: Johnson-Neyman method
not available with multicategorical X'.
end if.
do if (note(i,1) = 24).
print/title = 'NOTE: MMODVAL option not
available with a multicategorical moderator'.
end if.
do if (note(i,1) = 25).
print/title = 'NOTE: Omnibus test for relative
indirect effects is not available with the MC
option.'.
end if.
do if (note(i,1) = 26).
print/title = 'NOTE: CONTRAST option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 27).
print/title = 'NOTE: EFFSIZE option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 28).
print/title = 'NOTE: NORMAL option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 31).
print/title = 'NOTE: The TOTAL option is not
available when Y is dichotomous.'.
end if.
do if (note(i,1) = 32).
print/title = 'NOTE: Kappa-squared is disabled
from output as of version 2.16.'.
end if.
do if (note(i,1) = 33).
print/title = 'NOTE: COVMY option ignored in
models 1, 2, and 3.'.
```

| | | | |
|-----------|----------------|-------------|-------------|
| | | end if. | |
| | | end loop. | |
| | | end if. | |
| | | end matrix. | |
| Resources | Processor Time | | 00:00:01.05 |
| | Elapsed Time | | 00:00:01.03 |

Run MATRIX procedure:

***** PROCESS Procedure for SPSS Release 2.16.3 *****

Written by Andrew F. Hayes, Ph.D. www.afhayes.com
Documentation available in Hayes (2013). www.guilford.com/p/hayes3

Model = 1
Y = Operatio
X = Opra_Pra
M = Technolo

Sample size
856

Outcome: Operatio

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|----------|--------|----------|-------|
| .8686 | .7545 | .2949 | 872.9914 | 3.0000 | 852.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|--------|-------|----------|-------|---------|--------|
| constant | -.9401 | .1472 | -6.3889 | .0000 | -1.2290 | -.6513 |
| Technolo | .8337 | .0589 | 14.1591 | .0000 | .7182 | .9493 |
| Opra_Pra | .9902 | .0571 | 17.3467 | .0000 | .8782 | 1.1023 |
| int_1 | -.1471 | .0142 | -10.3671 | .0000 | -.1750 | -.1193 |

Product terms key:

int_1 Opra_Pra X Technolo

R-square increase due to interaction(s):

| | R2-chng | F | df1 | df2 | p |
|-------|---------|----------|--------|----------|-------|
| int_1 | .0310 | 107.4775 | 1.0000 | 852.0000 | .0000 |

Conditional effect of X on Y at values of the moderator(s):

| Technolo | Effect | se | t | p | LLCI | ULCI |
|----------|--------|-------|---------|-------|-------|-------|
| 2.7372 | .5875 | .0350 | 16.8037 | .0000 | .5189 | .6561 |
| 3.8107 | .4295 | .0353 | 12.1600 | .0000 | .3602 | .4988 |
| 4.8843 | .2716 | .0417 | 6.5152 | .0000 | .1898 | .3534 |

Values for quantitative moderators are the mean and plus/minus one SD from mean.

Values for dichotomous moderators are the two values of the moderator.

***** JOHNSON-NEYMAN TECHNIQUE *****

There are no statistical significance transition points within the observed range of the moderator.

Data for visualizing conditional effect of X on Y

Paste text below into a SPSS syntax window and execute to produce plot.

DATA LIST FREE/Opra_Pract Technological_Advan Operational_Performance.
BEGIN DATA.

| | | |
|--------|--------|--------|
| 2.7750 | 2.7372 | 2.9722 |
| 3.9019 | 2.7372 | 3.6342 |
| 5.0288 | 2.7372 | 4.2962 |
| 2.7750 | 3.8107 | 3.4289 |
| 3.9019 | 3.8107 | 3.9129 |
| 5.0288 | 3.8107 | 4.3970 |
| 2.7750 | 4.8843 | 3.8856 |
| 3.9019 | 4.8843 | 4.1916 |
| 5.0288 | 4.8843 | 4.4977 |

END DATA.

GRAPH/SCATTERPLOT=Opra_Pract WITH Operational_Performance BY
Technological_Advan.

***** ANALYSIS NOTES AND WARNINGS *****

Level of confidence for all confidence intervals in output:
95.00

----- END MATRIX -----

restore.

Matrix

Notes

| | | |
|----------------|--------------------------------|----------------------|
| Output Created | | 20-OCT-2020 16:04:42 |
| Comments | | |
| Input | Active Dataset | DataSet1 |
| | Filter | <none> |
| | Weight | <none> |
| | Split File | <none> |
| | N of Rows in Working Data File | 856 |

Syntax

```
matrix.  
get dat/file = */variables =  
Operational_Performance Comm_Pract  
Technological_Advan /names =  
vnames/missing = 9999.  
compute ninit = nrow(dat).  
get dat/file = */variables =  
Operational_Performance Comm_Pract  
Technological_Advan /names =  
vnames/missing = omit.  
get tmp/file = */variables =  
Operational_Performance /names =  
ynames/missing = omit.  
get tmp2/file = */variables = Comm_Pract  
/names = xnames/missing = omit.  
get tmp3/file = */variables =  
Technological_Advan /names =  
mnames/missing = omit.  
get tmp4/file = */variables = w999999t z999999t  
v999999t q999999t.  
compute wname=tmp(1,1).  
do if (wname = ' ').  
compute wname = 'xxx'.  
end if.  
compute zname=tmp(1,2).  
do if (zname = ' ').  
compute zname = 'xxx'.  
end if.  
compute vname=tmp(1,3).  
do if (vname = ' ').  
compute vname = 'xxx'.  
end if.  
compute qname=tmp(1,4).  
do if (qname = ' ').  
compute qname = 'xxx'.  
end if.  
compute n = nrow(dat).  
compute p0=-.322232431088.  
compute p1 = -1.  
compute p2 = -.342242088547.
```

```

compute p3 = -.0204231210245.
compute p4 = -.0000453642210148.
compute q0 = .0993484626060.
compute q1 = .588581570495.
compute q2 = .531103462366.
compute q3 = .103537752850.
compute q4 = .0038560700634.
compute badend = 0.
compute priorlo = -9999999.
compute priorhi = 9999999.
compute criterr = 0.
compute cluster = 0.
compute clsdmy = 0.
compute jndich = 0.
compute wvdich=0.
compute zqdich=0.
compute mod74dic=0.
compute warnrep=0.
compute contrast = trunc( 0 ).
do if (contrast > 2 or contrast < 0).
compute contrast = 0.
end if.
compute booterr = 0.
compute effsize = ( 0 = 1).
compute note = make(10,1,0).
compute notes = 1.
compute iterr = 0.
compute clsmth = 0.
compute quantile = ( 0 = 1).
compute mmodval= 999.
compute jn = ( 1 = 1).
compute covcoeff=( 0 =1).
compute center = ( 0 = 1).
compute detail = ( 1 = 1).
compute coeffci = ( 1 = 1).
compute conf = 95.
compute longname = ( 1 =1).
compute bconoff=( 0 <> 1).
compute covmy = trunc( 0 ).
do if (covmy < 0 or covmy > 2).

```

```

compute covmy = 0.
end if.
do if (trunc(conf) >= 100 or (trunc(conf) <=
50)).
compute conf = 95.
compute note(notes,1) = 1.
compute notes = notes + 1.
end if.
do if (n < ninit).
compute nmiss = ninit-n.
compute note(notes,1) = 11.
compute notes = notes + 1.
end if.
compute errs = 0.
compute quantd = {0,0,0,0,0,0}.
compute quantc = {0,0,0,0,0,0}.
compute mcheck = 0.
compute ttt = 0.
compute plot = trunc( 1 ).
compute runerrs = make(50,1,0).
compute model = trunc( 1 ).
compute normal = 0.
compute ws= ( 0 =1).
compute olsdichm= ( 0 =1).
compute olsdichy= ( 0 =1).
compute mcx=trunc( 0 ).
compute mcm=trunc( 0 ).
do if (ws = 1).
do if (effsize = 1).
compute note(notes,1) = 19.
compute notes = notes + 1.
compute effsize=0.
end if.
do if (normal=1).
compute note(notes,1) = 16.
compute notes = notes + 1.
compute normal=0.
end if.
end if.
do if (model <> 4).

```

```

compute ws = 0.
end if.
do if (model > 0 and model < 4 and covmy <>
0).
compute covmy=0.
compute note(notes,1) = 33.
compute notes = notes + 1.
end if.
print/space=0.
do if (jn = 1 and model <> 1 and model <> 3).
compute note(notes,1) = 7.
compute notes = notes + 1.
end if.
do if ((model > 76) or (model < 1)).
compute model = 77.
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 19.
end if.
compute itprob=0.
do if (mcx < 0).
compute mcx=0.
end if.
do if (mcx > 4).
compute mcx=0.
end if.
do if (mcm < 0).
compute mcm=0.
end if.
do if (mcm > 4).
compute mcm=0.
end if.
do if (mcx > 0 and mcm > 0).
compute errs=errs+1.
compute runerrs(errs,1)=34.
compute criterr=1.
end if.
do if ((model <> 1 and model <> 4) and (mcx >
0 or mcm > 0)).
compute errs=errs+1.

```

```
compute runerrs(errs,1)=35.  
compute criterr=1.  
end if.  
do if ((model = 4) and (mcm > 0)).  
compute errs=errs+1.  
compute runerrs(errs,1)=41.  
compute criterr=1.  
end if.  
do if ((mcx > 0) and jn <> 0).  
compute jn=0.  
compute note(notes,1) = 23.  
compute notes = notes + 1.  
end if.  
do if (mmodval <> 999 and mcm > 0).  
compute note(notes,1) = 24.  
compute notes = notes + 1.  
end if.  
compute toteff = 0.  
compute toteff = (( 0 = 1)*( 1 = 4 or 1 = 6)).  
compute varorder = 2.  
compute hc3 = ( 0 <> 0).  
compute cname = 'xxx'.  
compute centvar = {'xxx'}.  
compute nmeds = ncol(mnames).  
do if (longname=0).
```

```
compute criterr = 1.  
compute errs = errs+1.  
compute runerrs(errs,1) = 33.
```

1

2

3

end if.

compute modelm =

```
{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,6;  
1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,7;  
1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,8;  
1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,9;  
1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,3,10;  
1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,5,11;  
1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,0,5,12;  
1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5,13;  
0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,2,14;  
0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,2,15;  
0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,4,16;
```

0,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,4,17;
0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,6,18;
0,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,6,19;
0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,6,20;
1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,7,21;
1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,7,22;
1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,23;
1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,0,24; 1,1,1,1

,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,25;
1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,0,0,26;
1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,0,0,27;
1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,7,28;
1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,0,7,29;
1,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,30;
1,1,0,1,0,0,1,1,0,1,0,0,0,0,0,0,0,31;
1,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,0,32;
1,1,1,1,0,0,1,1,1,1,0,0,0,0,0,0,0,33;
1,1,1,1,0,0,1,0,0,1,0,0,0,0,0,0,0,34;
1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,35;
1,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,36;
1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,37;
1,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,38;
1,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,39;
1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,0,40;
1,0,0,1,1,0,1,0,0,1,1,0,0,0,0,0,0,41;
1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,42;
1,0,0,1,1,1,1,0,0,1,1,1,0,0,0,0,0,43;
1,0,0,1,1,1,1,0,0,1,0,0,0,0,0,0,0,44;
1,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,45;
1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,46;
1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,47;
1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,48;
1,1,0,1,1,0,1,1,0,0,0,

0,0,0,0,0,0,0,0,49;
1,1,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,50;
1,1,1,1,1,0,1,1,1,0,0,0,0,0,0,0,0,51;
1,1,1,1,1,0,0,0,0,1,1,0,0,0,0,0,0,52;
1,1,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,53;

```

1,1,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,54;
1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,55;
1,1,1,1,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,56;
1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,57;
1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,58;
1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,59;
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,60;
1,1,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,61;
1,1,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,62;
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,0,0,0,0,63;
1,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,64;
1,0,0,1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,65;
1,0,0,1,0,0,0,0,0,0,1,0,0,1,0,0,0,0,0,66;
1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,67;
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,68;
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,69;
1,0,0,1,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,70;
1,0,0,1,0,0,1,0,0,1,0,0,1,1,1,0,0,0,0,71;
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,1,1,0,0,72;
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,1,1,0,0

,73; 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,74;
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,75;
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,1,0,0,0,76;
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,77}.
compute wm = modelm(model, 1).
compute zm = modelm(model, 2).
compute wzm = modelm(model, 3).
compute vy = modelm(model, 4).
compute qy = modelm(model, 5).
compute vqy = modelm(model, 6).
compute wy = modelm(model, 7).
compute zy = modelm(model, 8).
compute wzy = modelm(model, 9).
compute vxy = modelm(model, 10).
compute qxy = modelm(model, 11).
compute vqxy = modelm(model, 12).
compute wmy = modelm(model, 13).
compute wvmy = modelm(model, 14).
compute wvxy = modelm(model, 15).

```

```

compute zmy = modelm(model,16).
compute wzmy = modelm(model,17).
compute xmy = modelm(model,18).
compute imm=modelm(model,19).
do if (model = 74).
compute vdich=0.
end if.
do if (ncol(xname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 20.
compute criterr = 1.
end if.
do if (ncol(mnames) > 10).
compute errs = errs+1.
compute runerrs(errs,1) = 39.
compute criterr = 1.
end if.
do if (ncol(yname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 21.
compute criterr = 1.
end if.
do if (( 999 =0) and ( 999 =0) and (model=3 or
model=2) and (contrast=1)).
compute errs = errs+1.
compute runerrs(errs,1) = 31.
compute criterr = 1.
end if.
compute xlist = (wm or zm or wzm or wy or zy
or wzy or vxy or qxy or vqxy or wvxy or xmy).
compute mlist = (vy or qy or vqy or zmy or wmy
or wzmy or xmy or (model < 4)).
compute bad=0.
do if (criterr = 0).
compute werr = 0.
compute verr = 0.
compute qerr = 0.
compute zerr = 0.
compute yerr = 1.
compute xerr = 1.

```

```

compute wlist = (wm or wzm or wy or wzy or
wm or wvmy or wvmy or wvxy or wzmy).
do if (wlist = 1 and wname = 'xxx').
compute werr = 1.
compute wlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (wlist = 1 and ((wname = qname) or
(wname = vname) or (wname = zname) or
(wname = xname) or (wname = yname))).
compute werr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 12).
end if.
compute zlist = (zm or wzm or zy or wzy or
zmy or wzmy).
do if (zlist = 1 and zname = 'xxx').
compute zerr = 1.
compute zlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (zlist = 1 and ((zname = qname) or
(zname = vname) or (zname = wname) or
(zname = xname) or (zname = yname))).
compute zerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 13.
end if.
compute qlist = (qy or vqy or qxy or vqxy).
do if (qlist = 1 and qname = 'xxx').
compute qerr = 1.
compute qlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (qlist = 1 and ((qname = zname) or
(qname = vname) or (qname = wname) or
(qname = xname) or (qname = yname))).

```

```

compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 14.
end if.
compute vlist = (vy or vqy or vxy or vqxy or
wvmy or wvxy).
do if (vlist = 1 and vname = 'xxx').
compute verr = 1.
compute vlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (vlist = 1 and ((vname = zname) or
(vname = qname) or (vname = wname) or
(vname = xname) or (vname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 15.
end if.
do if (wlist = 0 and wname <> 'xxx').
compute werr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 8.
end if.
do if (zlist = 0 and zname <> 'xxx').
compute zerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 9.
end if.
do if (qlist = 0 and qname <> 'xxx').
compute qerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 10.
end if.
do if (vlist = 0 and vname <> 'xxx').
compute verr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 11.
end if.
do if (hc3 = 1).

```

```

compute note(notes,1) = 3.
compute notes = notes+1.
end if.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute cons = make(n,1,1).
compute temp = (n*sscp(dat))-
(t(csum(dat))*(csum(dat))).
compute temp = temp/(n*(n-1)).
compute temp = csum(((diag(temp))=0)).
compute temp2=1.
do if (ws = 1).
compute temp2 = (n*sscp(tmp2))-
(t(csum(tmp2))*(csum(tmp2))).
compute temp2 = temp2/(n*(n-1)).
end if.
do if ((temp > 0 and ws = 0) or (temp > 0 and
ws = 1 and temp2 > 0)).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 27.
end if.
compute nmeds = ncol(mnames).
compute mcmats=ident(nmeds*2).
compute mccoef=make(nmeds*2,1,0).
compute sobel = make(nmeds,4,-999).
do if (model = 6 and nmeds > 4).
compute errs = errs+1.
compute runerrs(errs,1)=2.
end if.
do if (model < 4 and nmeds > 1).
compute errs = errs+1.
compute runerrs(errs,1) = 3.
end if.
compute nmods = (model = 74).
compute bad = 0.
compute intcnt = 1.

```

```

compute modvals = 0.
compute modvalsd = 0.
compute yintemp = {'int_1', 'int_2', 'int_3',
'int_4', 'int_5', 'int_6', 'int_7', 'int_8', 'int_9',
'int_10', 'int_11', 'int_12', 'int_13', 'int_14',
'int_15'}.
compute yintemp = {yintemp, 'int_16', 'int_17',
'int_18', 'int_19', 'int_20', 'int_21', 'int_22',
'int_23', 'int_24', 'int_25', 'int_26', 'int_27',
'int_28'}.
compute yintemp = {yintemp, 'int_29', 'int_30',
'int_31', 'int_32', 'int_33', 'int_34', 'int_35',
'int_36', 'int_37', 'int_38', 'int_39', 'int_40'}.
compute yintemp = {yintemp, 'int_41', 'int_42',
'int_43', 'int_44', 'int_45', 'int_46', 'int_47',
'int_48', 'int_49', 'int_50', 'int_51', 'int_52'}.
compute yintemp = {yintemp, 'int_53', 'int_54',
'int_55', 'int_56', 'int_57', 'int_58', 'int_59',
'int_60', 'int_61', 'int_62', 'int_63', 'int_64'}.
compute
cntname={'(C1)';'(C2)';'(C3)';'(C4)';'(C5)';'(C6)';(
C7)';'(C8)';'(C9)';'(C10)';'(C11)';'(C12)';'(C13)';(
C14)';'(C15)';'(C16)';'(C17)'}.
compute
cntname={cntname;'(C18)';'(C19)';'(C20)';'(C21
)';'(C22)';'(C23)';'(C24)';'(C25)';'(C26)';'(C27)';(
C28)';'(C29)';'(C30)';'(C31)'}.
compute
cntname={cntname;'(C32)';'(C33)';'(C34)';'(C35
)';'(C36)';'(C37)';'(C38)';'(C39)';'(C40)';'(C41)';(
C42)';'(C43)';'(C44)';'(C45)'}.
compute
cntname={cntname;'(C46)';'(C47)';'(C48)';'(C49
)';'(C50)';'(C51)';'(C52)';'(C53)';'(C54)';'(C55)';(
C56)';'(C57)';'(C58)';'(C59)'}.
compute
cntname={cntname;'(C60)';'(C61)';'(C62)';'(C63
)';'(C64)';'(C65)';'(C66)';'(C67)';'(C68)';'(C69)';(
C70)';'(C71)';'(C72)';'(C73)'}.

```



```

compute
cntname={cntname;'(C74)';'(C75)';'(C76)';'(C77
)';'(C78)';'(C79)';'(C80)';'(C81)';'(C82)';'(C83)';'(
C84)';'(C85)';'(C86)';'(C87)}.
compute
cntname={cntname;'(C88)';'(C89)';'(C90)';'(C91
)';'(C92)';'(C93)';'(C94)';'(C95)';'(C96)';'(C97)';'(
C98)';'(C99)';'(C100)';'(C101)}.
compute
cntname={cntname;'(C102)';'(C103)';'(C104)';'(
C105)}.
compute apathnam={'a path'; 'a1 path'; 'a2
path'; 'a3 path'; 'a4 path'; 'a5 path'; 'a6 path';
'a7 path'; 'a8 path'; 'a9 path'; 'a10 path'}.
compute bpathnam={'b path'; 'b1 path'; 'b2
path'; 'b3 path'; 'b4path'; 'b5path'; 'b6 path'; 'b7
path'; 'b8 path'; 'b9 path'; 'b10 path'}.
compute modvnm = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute modvnm2 = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute mlab = { ' M1 ='; ' M2 ='; ' M3 ='; '
M4 ='; ' M5 ='; ' M6 ='; ' M7 ='; ' M8 ='; '
M9 ='; ' M10 ='}.
compute m = make(n,nmeds,1).
compute mmat = make(16,nmeds,0).
compute ymat = make(8,nmeds,0).
compute deco = make(10,1,0).
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
compute iterate = abs(trunc( 10000 )).
compute converge = abs( 0.00000001 ).
compute boot = abs(trunc( 5000 )).
do if (ws=1 and 0 > 0).
compute note(notes,1)=17.
compute notes=notes+1.
end if.
compute mc=(abs(trunc( 0 ))*(1-ws)).

```

```

do if (mc > 0 and model > 5).
do if (boot = 0).
compute boot=mc.
end if.
compute mc=0.
compute note(notes,1)=12.
compute notes = notes+1.
end if.
compute adjust = 0.
do if (mc > 0 and (model > 3 and model < 6)).
compute boot=0.
compute bconoff=0.
end if.
do if (boot > 0 and mc > 0).
compute mc = 0.
end if.
compute savboot = 0.
compute saveboot = ( 0 = 1).
do if (ws=1 and saveboot=1).
compute saveboot=0.
compute note(notes,1) = 20.
compute notes = notes + 1.
compute normal=0.
end if.
do if (saveboot =1 and boot > 0 and model >
3).
compute savboot = 1.
end if.
do if (boot <> 0 or mc <> 0).
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
loop.
compute cilow = rnd(bootsz*(1-(conf/100))/2).
compute cihigh =
trunc((bootsz*(conf/100)+(bootsz*(1-
(conf/100))/2)))+1.
do if (cilow < 1 or cihigh > bootsz).

```

```

compute
bootz=trunc((bootz+1000)/1000)*1000.
compute adjust = 1.
end if.
end loop if (cilow gt 0 and cihigh le bootz).
compute boot=bootz.
do if (mc > 0).
compute mc = bootz.
end if.
do if (boot > 0 and mc > 0 and (model > 3 and
model < 6)).
compute boot=0.
end if.
do if (adjust = 1 and boot > 0).
compute note(notes,1)=6.
compute notes = notes+1.
end if.
do if (adjust = 1 and mc > 0).
compute note(notes,1)=13.
compute notes = notes+1.
end if.
end if.
do if (model = 6 and nmeds > 1).
compute mmpaths =
make((nmeds+2),(nmeds+2),0).
do if (nmeds = 2).
compute indboot = make(boot+1, 3, 999).
else if (nmeds = 3).
compute indboot = make(boot+1, 7, 999).
else if (nmeds = 4).
compute indboot = make(boot+1, 15, 999).
end if.
compute indlbl = {'Total:;' 'Ind1 :;' 'Ind2 :;' 'Ind3
:;' 'Ind4 :;' 'Ind5 :;' 'Ind6 :;' 'Ind7 :;' 'Ind8 :;' 'Ind9
:;' 'Ind10:;' 'Ind11:;' 'Ind12:;' 'Ind13:;' 'Ind14:;'
'Ind15:'}.
compute indlbl2 = {'Ind1'; 'Ind2'; 'Ind3'; 'Ind4';
'Ind5'; 'Ind6'; 'Ind7'; 'Ind8'; 'Ind9'; 'Ind10';
'Ind11'; 'Ind12'; 'Ind13'; 'Ind14'; 'Ind15'}.
compute indces = make(boot+1, 4, 999).

```

```

end if.
do if (model < 4).
compute boot = 0.
compute cmat = make(10,1,0).
compute zmat = make(10,1,0).
end if.
compute nvarch = make(1,ncol(dat),0).
compute wmatch = 0.
compute zmatch = 0.
compute vmatch = 0.
compute qmatch = 0.
compute mmatch=0.
compute minprobe=0.
compute maxprobe=0.
loop i = 1 to ncol(vnames).
do if (vnames(:,i)=yname).
compute y = dat(:,i).
compute nvarch(1,i)=1.
compute yerr = 0.
do if ((yname = xname) or (yname = wname) or
(yname = zname) or (yname = vname) or
(yname = qname)).
compute errs = errs+1.
compute runerrs(errs,1)=17.
end if.
end if.
do if (vnames(:,i)=xname).
compute x = dat(:,i).
compute nvarch(1,i)=1.
compute xdich = 1.
compute xerr=0.
loop jj = 1 to n.
do if ((x(jj,1) <> cmin(x)) and (x(jj,1) <>
cmax(x))).
compute xdich = 0.
break.
end if.
end loop.
compute xmean = csum(x)/n.

```

```

do if (center = 1 and (model < 4 or xlist > 0)
and mcx = 0).
compute meanvec = make(n,1, xmean).
compute x = x-meanvec.
compute centvar = {centvar, xname}.
end if.
compute xmean = csum(x)/n.
compute tmp = x-(cons*xmean).
compute xsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
do if (xdich = 0).
compute quantc(1,6) = 1.
compute matx = {xmean-xsd; xmean;
xmean+xsd}.
do if ( 999 = 999 and quantile = 0).
do if ((matx(1,1) < cmin(x)) and model = 74).
compute matx(1,1)=cmin(x).
compute minprobe=1.
end if.
do if ((matx(3,1) > cmax(x)) and model = 74).
compute matx(3,1)=cmax(x).
compute maxprobe=1.
end if.
end if.
do if (quantile = 1).
compute quantd(1,6) = 1.
compute quantc(1,6) = 0.
compute tmp = x.
compute tmp(GRADE(x)) = x.
compute matx =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (xdich = 1).
compute matx = {cmin(x); cmax(x)}.
do if (model=74).
compute matx=cmin(x).
compute mod74dic=1.
end if.

```

```

end if.
do if ( 999 <> 999).
compute matx = 999.
compute quantd(1,6) = 0.
compute quantc(1,6) = 0.
end if.
end if.
do if (werr = 0 and wlist = 1).
do if (vnames(:,i)=wname).
compute werr = 0.
compute wmatch = 1.
compute w = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(w)/n).
compute w = w-meanvec.
compute centvar = {centvar, wname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute wmean = csum(w)/n.
compute tmp = w-(cons*wmean).
compute wsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute wdich = 1.
loop jj = 1 to n.
do if ((w(jj,1) <> cmin(w)) and (w(jj,1) <>
cmax(w))).
compute wdich = 0.
break.
end if.
end loop.
do if (model = 3).
compute jndich=wdich.
compute jnmin=cmin(w).
compute jnmax=cmax(w).
end if.
do if (wdich = 0).
compute matw = {wmean-wsd; wmean;
wmean+wsd}.
do if ( 999 = 999 and quantile = 0).
do if (matw(1,1) < cmin(w)).

```

```

compute matw(1,1)=cmin(w).
compute minprobe=1.
end if.
do if (matw(3,1) > cmax(w)).
compute matw(3,1)=cmax(w).
compute maxprobe=1.
end if.
end if.
compute quantc(1,1) = 1.
do if (quantile = 1).
compute quantd(1,1) = 1.
compute quantc(1,1) = 0.
compute tmp = w.
compute tmp(GRADE(w)) = w.
compute matw =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (wdich = 1).
compute matw = {cmin(w); cmax(w)}.
compute wvdich=1.
compute cmaxw=cmax(w).
compute cminw=cmin(w).
end if.
do if ( 999 <> 999).
compute matw = 999.
compute quantd(1,1) = 0.
compute quantc(1,1) = 0.
end if.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm(1,1)=wname.
compute modmatp(1,1) = 1.
end if.
end if.
do if (zerr = 0 and zlist = 1).
do if (vnames(:,i)=zname).
compute zerr = 0.

```

```

compute zmatch = 1.
compute z = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(z)/n).
compute z = z-meanvec.
compute centvar = {centvar, zname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute zmean = csum(z)/n.
compute tmp = z-(cons*zmean).
compute zsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute zdich = 1.
loop jj = 1 to n.
do if ((z(jj,1) <> cmin(z)) and (z(jj,1) <>
cmax(z))).
compute zdich = 0.
break.
end if.
end loop.
do if (zdich = 0).
compute matz = {zmean-zsd; zmean;
zmean+zsd}.
do if ( 999 = 999 and quantile = 0).
do if (matz(1,1) < cmin(z)).
compute matz(1,1)=cmin(z).
compute minprobe=1.
end if.
do if (matz(3,1) > cmax(z)).
compute matz(3,1)=cmax(z).
compute maxprobe=1.
end if.
end if.
compute quantc(1,2) = 1.
do if (quantile = 1).
compute quantd(1,2) = 1.
compute quantc(1,2) = 0.
compute tmp = z.
compute tmp(GRADE(z)) = z.

```



```

compute matz = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (zdich = 1).
compute matz = {cmin(z); cmax(z)}.
compute cmaxz=cmax(z).
compute cminz=cmin(z).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matz = 999.
compute quantd(1,2) = 0.
compute quantc(1,2) = 0.
end if.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm(1,2)=zname.
compute modmatp(1,2) = 1.
end if.
end if.
do if (verr = 0 and vlist = 1).
do if (vnames(:,i)=vname).
compute verr = 0.
compute vmatch = 1.
compute v = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(v)/n).
compute v = v-meanvec.
compute centvar = {centvar, vname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute vmean = csum(v)/n.
compute tmp = v-(cons*vmean).
compute vsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute vdich = 1.
loop jj = 1 to n.

```

```

do if ((v(jj,1) <> cmin(v)) and (v(jj,1) <>
cmax(v))).
compute vdich = 0.
break.
end if.
end loop.
do if (vdich = 0).
compute matv = {vmean-vsd; vmean;
vmean+vsd}.
do if ( 999 = 999 and quantile = 0).
do if (matv(1,1) < cmin(v)).
compute matv(1,1)=cmin(v).
compute minprobe=1.
end if.
do if (matv(3,1) > cmax(v)).
compute matv(3,1)=cmax(v).
compute maxprobe=1.
end if.
end if.
compute quantc(1,3) = 1.
do if (quantile = 1).
compute quantd(1,3) = 1.
compute quantc(1,3) = 0.
compute tmp = v.
compute tmp(GRADE(v)) = v.
compute matv = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (vdich = 1).
compute matv = {cmin(v); cmax(v)}.
compute wvdich=1.
compute cmaxv=cmax(v).
compute cminv=cmin(v).
end if.
do if ( 999 <> 999).
compute matv = 999.
compute quantd(1,3) = 0.
compute quantc(1,3) = 0.

```

```

end if.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm(1,3)=vname.
compute modmatp(1,3) = 1.
end if.
end if.
do if (qerr = 0 and qlist = 1).
do if (vnames(:,i)=qname).
compute qerr = 0.
compute qmatch = 1.
compute q = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(q)/n).
compute q = q-meanvec.
compute centvar = {centvar, qname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute qmean = csum(q)/n.
compute tmp = q-(cons*qmean).
compute qsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute qdich = 1.
loop jj = 1 to n.
do if ((q(jj,1) <> cmin(q)) and (q(jj,1) <>
cmax(q))).
compute qdich = 0.
break.
end if.
end loop.
do if (qdich = 0).
compute matq = {qmean-qsd; qmean;
qmean+qsd}.
do if ( 999 = 999 and quantile = 0).
do if (matq(1,1) < cmin(q)).
compute matq(1,1)=cmin(q).
compute minprobe=1.
end if.
do if (matq(3,1) > cmax(q)).
compute matq(3,1)=cmax(q).

```

```

compute maxprobe=1.
end if.
end if.
compute quantc(1,4) = 1.
do if (quantile = 1).
compute quantd(1,4) = 1.
compute quantc(1,4) = 0.
compute tmp = q.
compute tmp(GRADE(q)) = q.
compute matq = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (qdich = 1).
compute matq = {cmin(q); cmax(q)}.
compute cmaxq=cmax(q).
compute cminq=cmin(q).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matq = 999.
compute quantd(1,4) = 0.
compute quantc(1,4) = 0.
end if.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm(1,4)=qname.
compute modmatp(1,4) = 1.
end if.
end if.
do if (vnames(:,i)=cname).
compute cld = dat(:,i).
compute cvname = vnames(:,i).
compute nvarch(1,i)=1.
compute clsmtch = 1.
end if.
loop j = 1 to ncol(mnames).
do if (vnames(:,i)=mnames(1,j)).
compute mmatch = mmatch + 1.

```

```

compute m(:,j)=dat(:,i).
do if (center = 1 and nvarch(1,i) = 0 and mlist >
0 and mcm = 0).
compute meanvec = make(n,1,csum(m(:,j))/n).
compute m(:,j) = m(:,j)-meanvec.
compute centvar = {centvar, mnames(1,j)}.
end if.
compute nvarch(1,i)=1.
compute dichm = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute dichm = 0.
break.
end if.
end loop.
do if ((dichm = 1) and (olsdichm = 1)).
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichm = 1 and model > 3 and mcheck = 0
and oldsdichm = 0).
compute errs= errs+1.
compute runerrs(errs,1) = 1.
compute mcheck = 1.
end if.
do if ((model <= 3) and (ncol(mnames)=1)).
compute nmods = nmods + 1.
compute mmean = csum(m(:,j))/n.
compute tmp = m(:,j)-(cons*mmean).
compute msd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute mdich = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute mdich = 0.
break.
end if.
end loop.
do if (model = 1).

```

```

compute jndich=mdich.
compute jnmin=cmin(m(:,j)).
compute jnmax=cmax(m(:,j)).
end if.
do if (mdich = 0).
compute matm = {mmean-msd; mmean;
mmean+msd}.
do if ( 999 = 999 and quantile = 0).
do if (matm(1,1) < cmin(m(:,j))).
compute matm(1,1)=cmin(m(:,j)).
compute minprobe=1.
end if.
do if (matm(3,1) > cmax(m(:,j))).
compute matm(3,1)=cmax(m(:,j)).
compute maxprobe=1.
end if.
end if.
compute quantc(1,5) = 1.
do if (quantile = 1).
compute quantd(1,5) = 1.
compute quantc(1,5) = 0.
compute tmp = m(:,j).
compute tmp(GRADE(m(:,j))) = m(:,j).
compute matm = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (mdich = 1).
compute matm = {cmin(m); cmax(m)}.
end if.
do if ( 999 <> 999).
compute matm = 999.
compute quantd(1,5) = 0.
compute quantc(1,5) = 0.
end if.
compute modmatv(1,5)=nrow(matm).
compute modmat((1:nrow(matm)), 5) = matm.
compute modvnm(1,5)=mnames(1,j).
compute modmatp(1,5) = 1.

```

```

end if.
end if.
end loop.
end loop.
do if (minprobe=1).
compute note(notes,1) = 14.
compute notes = notes + 1.
end if.
do if (maxprobe=1).
compute note(notes,1) = 15.
compute notes = notes + 1.
end if.
do if (cname <> 'xxx' and clsmatch = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 23.
end if.
do if (cname <> 'xxx').
do if ((cname = zname) or (cname = vname)
or (cname = wname) or (cname = xname) or
(cname = yname) or (cname = qname)).
compute errs = errs+1.
compute runerrs(errs,1) = 24.
end if.
end if.
do if (wlist = 1 and werr = 0 and wmatch = 0).
compute werr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (zlist = 1 and zerr = 0 and zmatch = 0).
compute zerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (qlist = 1 and qerr = 0 and qmatch = 0).
compute qerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (vlist = 1 and verr = 0 and vmatch = 0).

```

```

compute verr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (yerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 16.
end if.
do if (xerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 32.
end if.
do if (model = 6 and nmeds < 2).
compute errs = errs+1.
compute runerrs(errs,1) = 18.
end if.
do if (mmatch < ncol(mnames)).
compute errs=errs+1.
compute runerrs(errs,1) = 25.
end if.
do if (cname <> 'xxx').
compute cld = design(cld).
compute cluster = ncol(cld).
compute cld = cld(:,2:ncol(cld)).
compute clsdmy = ncol(cld).
do if (clsdmy > 19).
compute errs = errs+1.
compute runerrs(errs,1) = 26.
end if.
do if (mcx > 0 or mcm > 0).
compute errs = errs+1.
compute runerrs(errs,1) = 38.
end if.
end if.
compute dichy = 1.
loop jj = 1 to n.
do if ((y(jj,1) <> cmin(y)) and (y(jj,1) <>
cmax(y))).
compute dichy = 0.
break.

```



```

end if.
end loop.
do if ((dichy = 1) and (olsdichy = 1)).
compute dichy=0.
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichy = 1).
do if (toteff=1).
compute toteff=0.
compute note(notes,1) = 31.
compute notes = notes + 1.
end if.
compute jncrit=xp2*xp2.
do if (ws = 1).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 28.
end if.
end if.
compute ncovs = ncol(dat)-rsum(nvarch).
do if (effsize = 1 and covmy <> 0 and ncovs >
0 and model > 3 and model < 7).
compute note(notes,1) = 22.
compute notes = notes + 1.
compute effsize = 0.
end if.
do if (ws = 1 and ncovs < nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 29.
end if.
do if (ws = 1 and ncovs > nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 30.
end if.
end if.
do if (errs = 0).
do if (rsum(quantd) > 0).

```

```

compute note(notes,1) = 4.
compute notes = notes+1.
end if.
do if (rsum(quantc) > 0).
compute note(notes,1) = 5.
compute notes = notes+1.
end if.
do if (ncovs > 0).
compute c = make(n,ncovs,0).
compute cnames = {'x'}.
compute j = 1.
loop i = 1 to ncol(vnames).
do if (nvarch(1,i)) = 0.
compute c(:,j) = dat(:,i).
compute nvarch(1,i)=1.
compute j=j+1.
compute cnames = {cnames, vnames(:,i)}.
end if.
end loop.
compute cnames = cnames(1,2:ncol(cnames)).
do if (ws = 1).
compute covmean=csum(c)/n.
loop i = 1 to ncovs.
compute meanvec = make(n,1, covmean(1,i)).
compute c(:,i) = c(:,i)-meanvec.
end loop.
compute centvar = {centvar, cnames}.
compute x=c(:,1:ncovs).
do if (ncovs=nmads).
compute ncovs=0.
end if.
end if.
end if.
compute names = {yname, xname, mnames,
wname, zname, vname, qname}.
do if (ncovs > 0).
compute names = {names, cnames}.
end if.
do if (dichy = 1 and effsize = 1).
compute note(notes,1) = 2.

```

```

compute notes = notes+1.
end if.
do if (model > 3 and model < 6).
compute indeff=make(nmeds,1,0).
compute indboot=make(boot+1,nmeds,999).
do if (mc > 0).
compute indboot=make(mc+1,nmeds,999).
end if.
do if (effsize = 1 and dichy = 0).
compute rmeff=make(boot+1,nmeds+1,999).
compute abpseff=make(boot+1,nmeds+1,999).
compute abcseff=make(boot+1,nmeds+1,999).
compute pmeff=make(boot+1,nmeds+1,999).
compute r245 = make(boot+1,1,999).
compute kappa2 = make(boot+1,1,999).
end if.
end if.
do if (model = 6 and effsize = 1 and dichy = 0).
compute
rmeff=make(boot+1,ncol(indboot),999).
compute
abpseff=make(boot+1,ncol(indboot),999).
compute
abcseff=make(boot+1,ncol(indboot),999).
compute
pmeff=make(boot+1,ncol(indboot),999).
end if.
do if (nmods > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm(1,tmp) = modvnm(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:nmods).
compute modvnm=modvnm(:,1:nmods).
compute modmatv=modmatv(:,1:nmods).

```

```

loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvals =
make((modmatv(1,1)*modprod(1,1)),
nmods,0).
loop i = 1 to nmods.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvals)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvals(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
do if (model = 74).
compute modvals=matx.
compute modvnm=xname.
end if.
compute vmat = make(8,nrow(modvals),0).
compute vmat(1,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute vmat(5,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute indeff=make(nrow(modvals),1,0).
do if (model <> 5).
compute indboot = make(((boot+1)*nmeds),
nrow(modvals), -99999999).
compute indbootp =
make(boot+1,(nmeds*nmods), -99999999).
do if (imm > 4 and imm < 8).
do if (imm = 7).

```

```

compute immmv1=matw.
compute immmv2=matv.
end if.
do if (imm = 5).
compute immmv1=matw.
compute immmv2=matz.
end if.
do if (imm = 6).
compute immmv1=matv.
compute immmv2=matq.
end if.
compute
indboop1=make((boot+1),(nmeds*(nrow(immmv2))),-9999999).
compute
indboop2=make((boot+1),(nmeds*(nrow(immmv1))),-9999999).
compute onesmv1=make(1,nrow(immmv1),1).
compute onesmv2=make(1,nrow(immmv2),1).
compute
nimvs=nrow(immmv1)+nrow(immmv2).
compute cmmval=0.
compute cmmlbs1={'xx'}.
compute cmmlbs2={'xx'}.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv2}.
compute
cmmlbt=make(nrow(immmv2),1,mnames(1,i)).
compute cmmlbs1={cmmlbs1;cmmlbt}.
end loop.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv1}.
compute
cmmlbt=make(nrow(immmv1),1,mnames(1,i)).
compute cmmlbs2={cmmlbs2;cmmlbt}.
end loop.
compute cmmval=cmmval(2:nrow(cmmval),:).
end if.
end if.
end if.

```

```

do if (nmods> 0).
loop i = 1 to ncol(modvals).
do if (modvnm(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm(1,i)=qname).
compute qcol = i.
end if.
end loop.
end if.
do if (dichy = 1).
compute omx = cmax(y).
compute omn = cmin(y).
compute y = (y = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute data = {cons,y,m,x}.
compute datamed = data.
compute datayed = data.
compute datanm = {'constant'; yname;
t(mnames); xname}.
compute datanmm = {'constant'; yname;
t(mnames); xname}.
compute datanmy = {'constant'; yname;
t(mnames); xname}.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
do if (model < 4 and errs = 0).
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' ', ' '}.
compute datayed = {datayed, x&*m}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).

```

```

compute vmat(1,i) = 1.
compute vmat(2,i) = modvals(i,1).
end loop.
compute mmat = make(16,nmeds,1).
end if.
do if (model = 2 or model = 3).
compute int1 = x&*w.
compute datayed = {datayed, w, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,2).
compute vmat(3,i) = modvals(i,1).
compute vmat(4,i) = modvals(i,1)*modvals(i,2).
end loop.
end if.
do if (model = 3).
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*m.
compute int2 = x&*w&*m.
compute datayed = {datayed, int1, int2}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' X', wname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (model = 4 or model = 5).
compute vmat = make(8,1,1).
end if.
compute yintkey2 = yintkey.
do if (wm = 1).
compute int1 = x&*w.

```

```

compute datamed = {datamed, w,int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmm = {datanmm; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,wcol).
end loop.
do if (zm = 1).
compute int1 = x&*z.
compute datamed = {datamed, z, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(3,i) = modvals(i,zcol).
end loop.
end if.
do if (wzm = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*z.
compute int2 = x&*w&*z.
compute datamed = {datamed, int1, int2}.
loop i = 1 to nrow(modvals).
compute vmat(4,i) =
modvals(i,wcol)*modvals(i,zcol).
end loop.
end if.

```



```

end if.
compute mdatacol=ncol(datamed).
compute mintkey = yintkey.
compute yintkey = {' ',' ',' ',' ',' ',' '}.
compute medints = intcnt-1.
do if (vy = 1 or xmy = 1).
compute mp = 1.
do if (xmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,1).
end loop.
end if.
do if (vy = 1).
compute datayed = {datayed, v}.
compute datanmy = {datanmy; vname}.
compute mmods = 1.
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,vcol).
end loop.
do if (qy = 1).
compute mp = 2.
compute datayed = {datayed,q}.
compute datanmy = {datanmy; qname}.
compute mmods = 2.
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,qcol).
end loop.
end if.
do if (vqy = 1).
compute mp = 3.
compute datayed = {datayed, v&*q}.
compute mmods = 3.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,vcol)*modvals(i,qcol).
end loop.
end if.
end if.
compute mints = make(n,(nmeds*mp),0).
loop i = 0 to (nmeds-1).

```

```

do if (i = 0 and vqy = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
vname, ' X', qname, '', ''}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, '', ''}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (xmy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))&*x.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', xname, '', ''}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (qy = 1).
compute mints(:,((i*mp)+2))=m(:,(i+1))&*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', qname, '', ''}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqy = 1).
compute mints(:,((i*mp)+3))=m(:,(i+1))&*v&*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end loop.

```

```

compute datayed = {datayed, mints}.
end if.
compute mp = 1.
do if (wvmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,wcol)*modvals(i,vcol).
end loop.
end if.
compute mints2 = make(n,(nmeds*mp),0).
do if (wmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,wcol).
end loop.
do if (wy = 0 and model > 3).
compute datayed = {datayed, w}.
compute datanmy = {datanmy; wname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wvmy = 1).
compute datayed = {datayed,w&*v}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints2(:,((i*mp)+1))= m(:,(i+1))&*w.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wvmy = 1).
compute mints2(:,((i*mp)+2))=
m(:,(i+1))&*w&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', vname}.

```

```

compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints2}.
end if.
compute mp = 1.
do if (zmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,zcol).
end loop.
do if (wzmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,zcol)&*modvals(i,wcol).
end loop.
end if.
end if.
do if (zmy = 1).
compute mints3 = make(n,(nmeds*mp),0).
do if (zy = 0).
compute datayed = {datayed, z}.
compute datanmy = {datanmy; zname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wzmy = 1 and wzy = 0).
compute datayed = {datayed,w&*z}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints3(:,((i*mp)+1))= m(:,(i+1))&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.

```

```

compute intcnt = intcnt+1.
do if (wzmy = 1).
compute mints3(:,(i*mp)+2))=
m(:,(i+1))*w*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', zname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints3}.
end if.
compute decoc = 1.
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
do if (wy = 1 and model > 3).
compute datayed = {datayed, w, x*w}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm2(1,1)=wname.
compute modmatp(1,1) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (zy = 1).
compute datayed = {datayed,z,x*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm2(1,2)=zname.
compute modmatp(1,2) = 1.

```

```

compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, '', ''}.
compute datanmy = {datanmy; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (wzy = 1).
compute datayed = {datayed,w&*z,x&*w&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, '', ''}.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vxy = 1).
compute datayed = {datayed, x&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm2(1,3)=vname.
compute modmatp(1,3) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, '', ''}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (qxy = 1).
compute datayed = {datayed, x&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.

```

```

compute modvnm2(1,4)=qname.
compute modmatp(1,4) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqxy = 1).
compute datayed = {datayed, x&*v&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, 'X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end if.
do if (wvxy = 1).
compute datayed = {datayed, x&*w&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, 'X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute modvalsd = 0.
compute ttt = rsum(modmatp).
do if (rsum(modmatp) > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm2(1,tmp) = modvnm2(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.

```

```

end loop.
compute modmat=modmat(:,1:ttt).
compute modvnm2=modvnm2(:,1:ttt).
compute modmatv=modmatv(:,1:ttt).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvalsd =
make((modmatv(1,1)*modprod(1,1)), ttt,0).
loop i = 1 to ttt.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvalsd)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvalsd(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
end if.
do if (ttt > 0).
loop i = 1 to ncol(modvalsd).
do if (modvnm2(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm2(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm2(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm2(1,i)=qname).
compute qcol = i.

```



```

end if.
end loop.
compute directv = make(nrow(modvalsd),1,1).
do if (wy = 1).
compute directv ={directv, modvalsd(:,wcol)}.
end if.
do if (zy = 1).
compute directv ={directv, modvalsd(:,zcol)}.
end if.
do if (wzy = 1).
compute directv ={directv,
(modvalsd(:,wcol)&*modvalsd(:,zcol))}.
end if.
do if (vxy = 1).
compute directv ={directv, modvalsd(:,vcol)}.
end if.
do if (qxy = 1).
compute directv ={directv, modvalsd(:,qcol)}.
end if.
do if (vqxy = 1).
compute directv ={directv,
(modvalsd(:,vcol)&*modvalsd(:,qcol))}.
end if.
do if (wvxy = 1).
compute directv ={directv,
(modvalsd(:,vcol)&*modvalsd(:,wcol))}.
end if.
end if.
compute ydatacol=ncol(datayed).
do if (ncovs > 0).
do if (covmy <> 2).
compute datamed = {datamed,c}.
end if.
do if (covmy <> 1).
compute datayed = {datayed,c}.
end if.
do if (covmy = 1 and model = 4 and mcx > 0).
compute datayed = {datayed,c}.
end if.
compute covmeans = csum(c)/n.

```

```

end if.
do if (cluster > 0).
compute datamed = {datamed,cld}.
compute datayed = {datayed, cld}.
compute cldmeans = csum(cld)/n.
end if.
compute mst = 3.
compute mnd = mst+nmeds-1.
compute ydatacol=ncol(datayed).
compute mdatacol=ncol(datamed).
do if (ncovs > 0).
compute datanmy = {datanmy; t(cnames)}.
do if (model > 3).
compute datanmm = {datanmm; t(cnames)}.
end if.
end if.
compute datanmy = {'constant';
datanmy(3:nrow(datanmy),1)}.
do if (model > 3).
compute datanmm = {'constant';
datanmm(3:nrow(datanmm),1)}.
end if.
compute amm = make(2,1,0).
compute abmm = make(2,1,0).
compute mnv = csum(datayed(:,2)/n).
compute mnv = make(n,1,mnv).
compute ssty = csum((datayed(:,2)-mnv)**2).
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
do if (ws=1).
compute stddevm=diag(sigma).
compute
stddevm=sqrt(stddevm(3:(nmeds+2),1)).
end if.
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute r2xy =
(sigma(2,(3+nmeds)))/(stddevy*stddevx)**2.

```

```

compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))**2.
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).
do if (model = 4 and nmeds = 1 and cluster = 0
and ncovs = 0).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
compute datatm = datamed.
compute dataty = datayed.
compute mdlrms2 = { '1' ; yname; xname}.
compute mdlrms = {'Model ='; ' Y ='; ' X ='
}.
do if (ws = 1).
compute mdlrms2 = { '1' ; yname}.
compute mdlrms = {'Model ='; ' Y ='}.
end if.

```

```

loop i = 1 to ncol(mnames).
compute mdlInms2 = {mdlInms2; mnames(1,i)}.
do if (i = 1 and ncol(mnames) = 1).
compute mdlInms = {mdlInms; '  M = '}.
else.
compute mdlInms = {mdlInms; mlab(i,1)}.
end if.
end loop.
do if (wname <> 'xxx').
compute mdlInms2 = {mdlInms2; wname}.
compute mdlInms = {mdlInms; '  W = '}.
end if.
do if (zname <> 'xxx').
compute mdlInms2 = {mdlInms2; zname}.
compute mdlInms = {mdlInms; '  Z = '}.
end if.
do if (vname <> 'xxx').
compute mdlInms2 = {mdlInms2; vname}.
compute mdlInms = {mdlInms; '  V = '}.
end if.
do if (qname <> 'xxx').
compute mdlInms2 = {mdlInms2; qname}.
compute mdlInms = {mdlInms; '  Q = '}.
end if.
do if (jn = 1 and model = 1 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
do if (jn = 1 and model = 3 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
compute mdpbe=0.
compute medte2=0.
do if ( 1=4 and mcx > 0).
compute medte2=1.
end if.
do if (((mcx > 0) or (mcm > 0)) AND model =
1).
compute mdpbe=1.

```

```

compute dd={datayed(:,4),datayed(:,3)}.
do if (ncovs > 0).
compute dd={datayed(:,6:(5+ncovs)),dd}.
end if.
compute dd={datayed(:,2),dd}.
end if.
compute yes = ((nrow(modvals)=1) and
(contrast = 1) and (model = 3 or model = 2)).
compute yes2 = ((nrow(modvals)=1) and
(contrast = 1) and (model = 2)).
print/title = '***** PROCESS Procedure
for SPSS Release 2.16.3 *****'.
print/title = '      Written by Andrew F. Hayes,
Ph.D.    www.afhayes.com'.
print/title = '  Documentation available in
Hayes (2013).
www.guilford.com/p/hayes3/space=0.
print mdlnms2/title =
*****
*****'/rnames = mdlnms/format =
a8.
do if (ncovs > 0).
print cnames/title = 'Statistical Controls:'/'rlabels
= 'CONTROL='/'format a8.
end if.
print n/title = 'Sample size'/'format F10.0.
do if (cluster > 0).
print cluster/rnames = cvname/title =
'Clustering Variable and Number of Clusters'.
end if.
do if (model > 3 and ( 'random' <> 'random')).
compute seedt= 'random'.
print seedt/title='Custom'/'format=A12/'rlabel =
'Seed:'.
end if.
do if (mdpbe=0 and medte2 = 0).
loop bt = 1 to (boot+1).
do if (bt = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)),-999).

```

```

end if.
compute bootcoef = 0.
do if (bt > 1).
loop.
compute v=trunc(uniform(n,1)*n)+1.
compute datayed = dataty(v,:).
do if (ws=1).
compute
bmeans=csum(datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed)))/n.
compute cntmat=make(n,nmeds,1).
compute bmeans2=bmeans.
do if (nmeds > 1).
compute bmeans2=mdia(bmeans).
end if.
compute cntmat=cntmat*bmeans2.
compute datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed))=datayed(:,(ncol(data
yed)-nmeds+1):ncol(datayed))-cntmat.
end if.
compute detcheck=det(t(datayed)*datayed).
compute rk=(detcheck <> 0).
do if (model > 3).
compute datamed = datatm(v,:).
compute detcheck=det(t(datamed)*datamed).
do if (rk=1).
compute rk=(detcheck <> 0).
end if.
end if.
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
compute temp=diag(sigma).
compute bad = bad+(1-rk).
compute false = 1.
end loop if (rk = 1).
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).

```

```

compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).
do if (model = 4 and nmeds = 1 and ncovs = 0
and cluster = 0).
compute r2xy =
(sigma(2,4)/(stddevy*stddevx))&**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))&**2.
compute sstot = sigma(2,2)*(n-1).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
end if.
do if (model > 3).
loop im = 1 to nmeds.
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.

```

```

compute xnm = {'constant';
datanmm((2+nmeds):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
do if (model = 6).
do if (im = 1).
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
end if.
do if (im > 1).
compute xm={cons, datamed(:,3:(im+1)),
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant'; datanmm(2:im,1);
datanmm((mnd):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
compute
mmpaths((im+1),(2:im))=t(coeff(2:im,1)).
end if.
end if.
do if (ws = 1).
compute
coeff(1,1)=csum(datamed(:,(2+im)))/n.
end if.
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
do if (bt = 1).
compute resid=datamed(:,(2+im))-xm*coeff.
compute sse =cssq(resid).
compute mse = sse/(n-ncol(xm)).
compute mnv = csum(data(:,(2+im)))/n.
compute mnv = make(n,1,mnv).
compute sstm = csum((data(:,(2+im))-
mnv)**2).

```



```

compute k3 = nrow(coeff).
do if (hc3 = 1).
compute h = xm(:,1).
loop i3=1 to n.
compute h(i3,1)= xm(i3,:)*invXtX*t(xm(i3,:)).
end loop.
loop i3=1 to k3.
compute xm(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xm(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xm(:,i3) = sqrt(mse)&*xm(:,i3).
end loop.
end if.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute hccov=invXtX*t(xm)*xm*invXtX.
compute mcmats(im,im)=hccov(2,2).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*hccov*lmat)*((t(lmat)
)*coeff))/dfnum.
compute coeff = coeff(1:(nrow(coeff)-
clsdmy),1).
compute mccoef(im,1)=coeff(2,1).
compute standerr =
sqrt(diag(invXtX*t(xm)*xm*invXtX)).
compute standerr = standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1).
compute standerr(1,1)=stddevm(im,1)/sqrt(n).
end if.
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-
ncol(xm)))).
compute temp=(n-ncol(xm)).
do if (ws=1).

```

```

compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute temp=(n-1).
end if.
compute xd = abs(xp2).
compute temp = (temp* (exp((temp-
(5/6))*((xd/(temp-(2/3)+(.11/temp)))*(xd/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = coeff-
sqrt(abs(temp))*standerr.
compute temp2 =
coeff+sqrt(abs(temp))*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (ws=1).
compute op=op(1,:).
compute xnmn =
apathnam(((nmeds>1)+1+(im-1)),1).
end if.
compute sobel(im,1) = coeff(2,1).
compute sobel(im,2)=standerr(2,1).
compute temp = mnames(1,im).
compute r2full = 1-(sse/sstm).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute summ = {sqrt(r2full), r2full, mse,
Fratio, dfnum, dfden, pfr}.
do if (detail = 1).
print temp/title =
*****
*****'/rlabels = 'Outcome:'/format
= A8.
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels 'R',
'R-sq', 'MSE','F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
print op/title = 'Model'/rnames = xnmn/clabels
'coeff', 'se', 't', 'p', 'LLCI', 'ULCI'/format = F10.4.
do if (covcoeff=1 and ws <> 1).

```

```

compute
hccovtmp=hccov(1:nrow(op),1:nrow(op)).
compute cnamestp=t(xmnm).
print hccovtmp/title='Covariance matrix of
regression parameter
estimates'/rnames=xmnm/cnames=cnamestp/f
ormat= F10.4.
end if.
do if (nmods > 0 and nrow(mintkey) > 1).
print mintkey/title = 'Product terms key:.'/format
= A8.
end if.
do if (ws = 1).
compute wsdf=n-1.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
end if.
end if.
compute ymat(1,im) = coeff(2,1).
do if (ws = 1).
compute ymat(1,im)=coeff(1,1).
end if.
do if (wm = 1).
compute ymat(2,im) = coeff(4,1).
do if (zm = 1).
compute ymat(3,im) = coeff(6,1).
do if (wzm = 1).
compute ymat(4, im) = coeff(8,1).
end if.
end if.
end if.
do if (model = 6).
compute mmpaths((im+1),1)=coeff((im+1),1).
end if.
end loop.
end if.
loop totlp = 1 to (1+(toteff*(bt = 1))).
do if (toteff = 1 and totlp =2).
compute xy =
{cons,datayed(:,(3+nmeds):ydatacol)}.

```

```

else.
compute xy={cons, datayed(:,3:ydatacol)}.
end if.
do if (dichy = 1).
compute pt2 =
make(nrow(datayed(:,2)),1,(csum(datayed(:,2))
/n)).
compute LL3 = datayed(:,2)*ln(pt2)+(1-
datayed(:,2))*ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xy),1,0).
compute LL1 = 0.
compute xy22=xy.
loop jjj = 1 to iterate.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.
compute coeff =
bt1+inv(t(xy22)*xy)*t(xy)*(datayed(:,2)-pt1).
compute pt1 = 1/(1+exp(-(xy*coeff))).
compute itprob = csum((pt1 <
.00000000000001) or (pt1 >
.99999999999999)).
do if (itprob = 0).
compute LL = datayed(:,2)*ln(pt1)+(1-
datayed(:,2))*ln(1-pt1).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
break.
end if.
compute bt1 = coeff.
compute LL1 = LL2.
end loop.
do if (jjj >= iterate and iterr = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
computer iterr = 1.
end if.

```

```

loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)&*pt1&*(1-pt1).
end loop.
compute covmat = inv(t(xy22)*xy).
release xy22.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
end if.
do if (dichy = 0).
compute invXtX = inv(t(xy)*xy).
compute coeff =invXtX*t(xy)*datayed(:,2).
do if (effsize=1 and model > 3 and model < 7
and (ncovs > 0 or clsdmy > 0)).
compute xysdy =
{cons,datayed(:,(4+nmeds):ydatacol)}.
compute coeffsd
=inv(t(xysdy)*xysdy)*t(xysdy)*datayed(:,2).
compute resid=datayed(:,2)-(xysdy*coeffsd).
compute sdy cov =sqrt(csum(resid&*resid)/(n-
ncol(xysdy))).
compute xvaron=datayed(:,(3+nmeds)).
compute
coeffx2=inv(t(xysdy)*xysdy)*t(xysdy)*xvaron.
compute residx2=xvaron-(xysdy*coeffx2).
compute sdx cov
=sqrt(csum(residx2&*residx2)/(n-ncol(xysdy))).
end if.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4 and bt > 1).
compute resid=datayed(:,2)-xy*coeff.
compute sse = cssq(resid).
compute r2full = 1-(sse/sstot).

```

```

end if.
do if (bt = 1).
compute resid=data(:,2)-xy*coeff.
compute k3 = nrow(coeff).
compute sse = cssq(resid).
compute mse = sse/(n-ncol(xy)).
compute
hcnott=coeff&/(sqrt(diag(mse*invXtX))).
do if (hc3 = 1).
compute h = xy(:,1).
loop i3=1 to n.
compute h(i3,1)= xy(i3,:)*invXtX*t(xy(i3,:)).
end loop.
loop i3=1 to k3.
compute xy(:,i3) = (resid(:,ncol(resid))&/(1-
h))*xy(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xy(:,i3) = sqrt(mse)*xy(:,i3).
end loop.
end if.
compute covmat = (invXtX*t(xy)*xy*invXtX).
end if.
end if.
do if (bt = 1).
do if (model = 2).
compute xy2={cons, datayed(:,3:ydatacol)}.
compute temp = ncol(xy2).
do if (temp > 6).
compute xy3=xy2(:,7:temp)}.
end if.
compute xy2={xy2(:,1:3), xy2(:,5)}.
do if (temp > 6).
compute xy2={xy2, xy3}.
release xy3.
end if.
compute invXtX = inv(t(xy2)*xy2).
compute coeff2 =invXtX*t(xy2)*datayed(:,2).

```

```

compute ssem2=cssq(datayed(:,2)-xy2*coeff2).
release xy2.
end if.
compute standerr = sqrt(diag(covmat)).
do if (totlp =1).
compute
mcmats((nmeds+1):ncol(mcmats),(nmeds+1):n
col(mcmats))=covmat(2:(1+nmeds),2:(1+nmed
s)).
end if.
compute standerr=standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1 and totp=2).
compute standerr(1,1)=stddevy/sqrt(n).
end if.
compute coeffplt = coeff.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*coeff)))/dfnum).
compute coeff=coeff(1:(nrow(coeff)-clsdmy),1).
do if (totlp=1).
compute
mccoeff((nmeds+1):nrow(mccoeff))=coeff(2:(1+
nmeds),1).
end if.
compute bbbb=coeff(2,1).
do if totp = 1.
compute deco(1,1)=2+nmeds.
compute deco = deco(1:decoc,1).
compute covdirt = make((nrow(covmat)-
clsdmy),(ncol(covmat)-clsdmy),0).
compute covdirt = covmat(deco,:).
compute covdir =
make(nrow(covdirt),ncol(covdirt),0).
compute covdir = covdirt(:,t(deco)).
compute deco=coeff(deco,1).

```

```

do if (ttt > 0).
compute sedir =
sqrt(diag(directv*covdir*t(directv))).
compute directv = directv*deco.
end if.
compute sobel(:,3)=coeff(2:(1+nmeds),1).
compute sobel(:,4) = standerr(2:(1+nmeds),1).
compute sobel2 = sobel&*sobel.
do if (varorder <> 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)&*sobel2(:,4)+sobel2(:,3)&*sob
el2(:,2)).
end if.
do if (varorder = 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)&*sobel2(:,4)+sobel2(:,3)&*sob
el2(:,2)+sobel2(:,2)&*sobel2(:,4)).
end if.
compute sobel(:,1) = sobel(:,1)&*sobel(:,3).
compute sobel(:,3) = sobel(:,1)&/sobel(:,2).
compute sobel(:,4) = 2*(1-
cdfnorm(abs(sobel(:,3)))).
end if.
do if (dichy = 0).
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
do if (ws= 1 and totlp=2).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute dfden=n-1.
end if.
compute cnms = {'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute op = {coeff, standerr, tratio, p}.
end if.
do if (dichy = 1).
compute tratio = (coeff&/standerr).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute wald = tratio&*tratio.
compute cnms = {'coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.

```



```

compute temp=coeff-abs(xp2)*standerr.
compute op = {coeff, standerr, tratio, p, temp}.
compute temp=coeff+abs(xp2)*standerr.
compute op = {op, temp}.
end if.
do if (detail = 1).
do if (totlp = 2).
print yname/title = '*****'
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:'/'format = A8.
end if.
do if (totlp <> 2).
print yname/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.
end if.
end if.
do if (dichy = 1 and bt = 1 and totp = 1).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:'/'cnames = nmsd/'format = F9.2.
end if.
do if (dichy = 0).
compute r2full = 1-(sse/ssty).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute jndf=dfden.
compute wsdf=dfden.
compute xd = abs(xp2).
compute jncrit = (dfden* (exp((dfden-
(5/6))*((xd/(dfden-
(2/3)+(.11/dfden)))*(xd/(dfden-
(2/3)+(.11/dfden)))))-1)).
compute summ = {sqrt(r2full), r2full, mse,
fratio, dfnum, dfden, pfr}.
compute temp1=coeff-sqrt(jncrit)*standerr.
compute temp2=coeff+sqrt(jncrit)*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.

```

```

do if (detail = 1).
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels =
'R', 'R-sq', 'MSE' 'F', 'df1', 'df2', 'p'/format =
F10.4.
end if.
end if.
end if.
do if (dichy = 1).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(coeff)-
1)).
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
do if (detail = 1).
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format =
F10.4.
end if.
end if.
do if (totlp = 2).
compute datanmy={'constant';
datanmy((nmeds+2):nrow(datanmy),1)}.
end if.
do if (detail = 1).
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
do if (ws=1).
compute op2=op(1:(1+nmeds),:).
compute datanmy2 = {'c" path'; bpathnam(
(1+(nmeds>1)): ((1+(nmeds>1))+(nmeds-
1)),1)}.
do if (totlp=2).
compute datanmy2={'c path'}.
compute op2=op(1,:).

```

```

compute wsdf=n-1.
end if.
print op2/title = 'Model'/rnames =
datanmy2/cnames = cnms/format = F10.4.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
do if (ws <> 1).
print op/title = 'Model'/rnames =
datanmy/cnames = cnms/format = F10.4.
do if (covcoeff=1).
compute
covmattp=covmat(1:nrow(op),1:nrow(op)).
compute cnamestp=t(datanmy).
print covmattp/title='Covariance matrix of
regression parameter
estimates'/rnames=datanmy/cnames=cnamest
p/format= F10.4.
end if.
end if.
end if.
do if (ttt = 0 and totlp = 1).
compute deco = op((nmeds+2),:).
do if (ws = 1).
compute deco=op(1,:).
end if.
end if.
do if (ttt = 0 and totlp = 2).
compute decotot = op(2,:).
do if (ws = 1).
compute decotot=op(1,:).
end if.
end if.
do if (nmods > 0 and model > 4 and detail = 1
and nrow(yintkey) > 1)).
print yintkey/title = 'Product terms key:'/format
= A8.
end if.
do if (nmods > 0 and model < 4 and detail = 1).
print yintkey2/title = 'Product terms key:'/format
= A8.

```

```

do if ((model = 1 or model = 2) and dichy = 0).
compute temp={((op(4,3)**2)*(1-r2full))/dfden,
op(4,3)**2, 1, dfden, op(4,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(4,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(2,1).
do if (model = 2).
compute temp={temp;((op(6,3)**2)*(1-
r2full))/dfden, op(6,3)**2, 1, dfden, op(6,4)}.
do if (hc3=1).
compute temp(2,1)=((hcnot(6,1)**2)*(1-
r2full))/dfden.
end if.
compute frat2=(dfden*(r2full-(1-
(ssem2/ssty))))/(2*(1-r2full)).
do if (hc3=1).
compute lmat=make(nrow(coeffplt),2,0).
compute lmat(4,1)=1.
compute lmat(6,2)=1.
compute frat2 =
(t(t(lmat)*coeffplt)*inv(t(lmat)*covmat*lmat)*((t(l
mat)*coeffplt))/2).
end if.
compute temp={temp;(r2full-(1-
(ssem2/ssty))),frat2,2,dfden,1-
fcdf(frat2,2,dfden)}.
compute rnms={rnms; yintkey2(3,1),'Both'}.
end if.
print temp/title = 'R-square increase due to
interaction(s):'/rnames=rnms/clabels = 'R2-
chng', 'F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (model = 3 and dichy = 0).
compute temp={((op(8,3)**2)*(1-r2full))/dfden,
op(8,3)**2, dfden, op(8,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(8,1)**2)*(1-
r2full))/dfden.

```

```

end if.
compute rnms=yintkey2(5,1).
print temp/title = 'R-square increase due to
three-way interaction:.'/rnames=rnms/clabels =
'R2-chng', 'F(1,df2)', 'df2', 'p'/format = F10.4.
end if.
end if.
end if.
do if (model = 6 and totlp = 1).
compute
mmpaths(nrow(mmpaths),1)=coeff(nrow(mmpa
ths),1).
compute
mmpaths(nrow(mmpaths),(2:(nmeds+1)))=t(co
eff(2:(nmeds+1),1)).
end if.
do if (totlp = 1).
loop im = 1 to nmeds.
do if (model < 4).
compute ymat(1,im) = coeff(3,1)*(1-yes).
compute ymat(2,im) = coeff(4,1).
compute cmat(1,im) = covmat(3,3).
compute cmat(2,im) = covmat(4,4).
compute cmat(5,im) = covmat(3,4).
compute jnb1=coeff(3,1).
compute jnb3=coeff(4,1).
compute jnsb1=covmat(3,3).
compute jnsb3=covmat(4,4).
compute jnsb1b3=covmat(3,4).
do if (model = 2 or model = 3).
compute ymat(3,im) = coeff(6,1).
compute cmat(3,im) = covmat(6,6).
compute cmat(6,im) = covmat(3,6).
compute cmat(8,im) = covmat(4,6).
end if.
do if (model = 3).
compute ymat(4,im) = coeff(8,1).
compute cmat(4,im) = covmat(8,8).
compute cmat(7,im) = covmat(3,8).
compute cmat(9,im) = covmat(4,8).

```

```

compute cmat(10,im) = covmat(6,8).
compute jnb1=coeff(4,1).
compute jnb3=coeff(8,1).
compute jnsb1=covmat(4,4).
compute jnsb3=covmat(8,8).
compute jnsb1b3=covmat(4,8).
end if.
end if.
do if (model > 3).
compute ymat(5,im) = coeff((1+im),1).
end if.
do if (xmy = 1).
compute ymat(6,im) = coeff((2+nmeds+im),1).
end if.
do if (vy = 1).
compute ymat(6,im) = coeff((3+nmeds+im),1).
end if.
do if (qy = 1 and vy = 1).
compute ymat(6,im) = coeff((5+nmeds+((im-1)*2)),1).
compute ymat(7,im) = coeff((6+nmeds+((im-1)*2)),1).
end if.
do if (vqy = 1).
compute ymat(6,im) = coeff((6+nmeds+((im-1)*3)),1).
compute ymat(7,im) = coeff((7+nmeds+((im-1)*3)),1).
compute ymat(8,im) = coeff((8+nmeds+((im-1)*3)),1).
end if.
do if (wmy = 1).
compute ymat(7,im) = coeff((3+nmeds+im-wy),1).
end if.
do if (wmy = 1 and vy = 1).
compute ymat(7,im) =coeff((4+(nmeds*2)+im-wy),1).
end if.
do if (wmy = 1 and vy = 1 and wvmy = 1).

```

```

compute ymat(7,im) =
coeff((6+(nmeds*2)+((im-1)*2)-wy),1).
compute ymat(8,im) =
coeff((7+(nmeds*2)+((im-1)*2)-wy),1).
end if.
do if (wmy = 1 and zmy = 1).
compute ymat(6,im) = coeff((6-(wzy*3)+(wzm-
1)+(nmeds*2)+((im-1)*2)*wzm)+((im-1)*(1-
wzm))-((zy-wzy)*2),1).
compute ymat(7,im) = coeff((3-
zy+im+nmeds),1).
do if (wzmy = 1).
compute ymat(8,im) = coeff((7-
(wzy*3)+(nmeds*2)+((im-1)*2)),1).
end if.
end if.
do if (nmods > 0 and model <> 5).
loop indlp = 1 to nrow(modvals).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp)).
do if (model > 6).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp))*csum(ym
at(5:8,im)*vmat(5:8,indlp)).
end if.
end loop.
compute indboot((bt+(im-
1)*(boot+1)),:)=t(indeff).
do if (imm = 1 or imm = 3).
compute
indbootp(bt,im)=ymat(2,im)*ymat(5,im).
do if (imm=3).
compute
indbootp(bt,(im+nmeds))=ymat(3,im)*ymat(5,i
m).
do if (zdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxz-cminz).
end if.

```

```

end if.
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
end if.
end if.
do if ((imm= 2 or imm = 4) or model = 74).
compute
indbootp(bt,im)=ymat(1,im)*ymat(6,im).
do if (imm=4).
compute
indbootp(bt,(im+nmeds))=ymat(1,im)*ymat(7,i
m).
do if (qdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxq-cminq).
end if.
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
end if.
end if.
do if (imm=7).
compute
indbootp(bt,im)=ymat(2,im)*ymat(6,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(2,im)*y
mat(6,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(6,im)*onesmv1)+(ymat(2,im)*y
mat(6,im)*t(immmv1)).
do if (wdich=1).

```



```

compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxv-cminv).
end if.
end if.
do if (imm=5).
compute
indbootp(bt,im)=ymat(4,im)*ymat(5,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(4,im)*y
mat(5,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(3,im)*ymat(5,im)*onesmv1)+(ymat(4,im)*y
mat(5,im)*t(immmv1)).
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).

```

```

end if.
do if (zdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxz-
cminz).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxz-cminz).
end if.
end if.
do if (imm=6).
compute
indbootp(bt,im)=ymat(1,im)*ymat(8,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(1,im)*ymat(6,im)*onesmv2)+(ymat(1,im)*y
mat(8,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(7,im)*onesmv1)+(ymat(1,im)*y
mat(8,im)*t(immmv1)).
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxv-cminv).
end if.
do if (qdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxq-
cminq).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-

```

```

1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxq-cminq).
end if.
end if.
do if ((model = 58 or model = 59) and
wvdich=1).
compute
indbootp(bt,im)=(ymat(1,im)*ymat(7,im)*(cmax
w-cminw))+(ymat(2,im)*ymat(5,im)*(cmaxw-
cminw))+(ymat(2,im)*ymat(7,im)*((cmaxw*cma
xw)-(cminw*cminw))).
end if.
end if.
do if (model = 4 or model = 5).
compute
indboot(bt,im)=csum(ymat(1:4,im)*vmat(1:4,1
))*csum(ymat(5:8,im)*vmat(5:8,1)).
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.000000000000001.
end if.
compute sumind=rsum(indboot(bt,:)).
do if (im=nmeds).
compute
pmeff(bt,2:(im+1))=indboot(bt,:)/(sumind+coeff(
(2+nmeds),1)).
end if.
compute
rmeff(bt,(im+1))=indboot(bt,im)/coeff((2+nmeds
),1).
compute
abpseff(bt,(im+1))=indboot(bt,im)/stddevy.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*stddevx.
do if (ncovs > 0 or clsdmy > 0).
compute
abpseff(bt,(im+1))=indboot(bt,im)/sdy cov.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*sdx cov.
end if.

```

```

do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,1) = r2my-(r2full-r2xy).
compute abmmr = 1.
do if (indboot(bt,im) < 0).
compute abmmr = cmin(abmm).
end if.
do if (indboot(bt,im) > 0).
compute abmmr = cmax(abmm).
end if.
compute kappa2(bt,1)=indboot(bt,im)/abmmr.
compute tmp = indboot(bt,im)/abmmr.
end if.
end if.
end if.
do if (model = 6).
do if (nmeds = 2).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(4,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3).
compute
indboot(bt,3)=mmpaths(3,1)*mmpaths(4,3).
end if.
do if (nmeds = 3).
compute indboot(bt,1) =
mmpaths(2,1)*mmpaths(5,2).
compute indboot(bt,2) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3).
compute indboot(bt,3) =
mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4).
compute indboot(bt,4) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4).
compute indboot(bt,5) =
mmpaths(3,1)*mmpaths(5,3).
compute indboot(bt,6) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4).

```

```

compute indboot(bt,7) =
mmpaths(4,1)*mmpaths(5,4).
end if.
do if (nmeds = 4).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(6,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(6,3).
compute
indboot(bt,3)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(6,4).
compute
indboot(bt,4)=mmpaths(2,1)*mmpaths(5,2)*mmpaths(6,5).
compute
indboot(bt,5)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(6,4).
compute
indboot(bt,6)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3)*mmpaths(6,5).
compute
indboot(bt,7)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,8)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,9)=mmpaths(3,1)*mmpaths(6,3).
compute
indboot(bt,10)=mmpaths(3,1)*mmpaths(4,3)*mmpaths(6,4).
compute indboot(bt,11)
=mmpaths(3,1)*mmpaths(5,3)*mmpaths(6,5).
compute indboot(bt,12) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,13) =
mmpaths(4,1)*mmpaths(6,4).

```

```

compute indboot(bt,14) =
mmpaths(4,1)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,15) =
mmpaths(5,1)*mmpaths(6,5).
end if.
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.00000000000001.
end if.
compute
pmeff(bt,:)=indboot(bt,:)/(rsum(indboot(bt,:))+m
mpaths(nrow(mmpaths),1)).
compute rmeff(bt,:) =
indboot(bt,:)/mmpaths(nrow(mmpaths),1).
compute abpseff(bt,:)=indboot(bt,:)/stddevy.
compute
abcseff(bt,:)=(stddevx*indboot(bt,:))/stddevy.
do if (ncovs > 0 or clsdmy > 0).
compute abpseff(bt,:)=indboot(bt,:)/sdyconv.
compute
abcseff(bt,:)=(sdxcov*indboot(bt,:))/sdyconv.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,:) = r2my-(r2full-r2xy).
end if.
end if.
end if.
end loop.
end if.
end loop.
do if (savboot = 1 and bt > 1).
compute bootstrp((bt-1),:)=bootcoef.
end if.
end loop.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.

```

```

release datayed, dat, datamed, data, y, x,
datatm, dataty, mints2, cons, mnv, tmp.
do if (mc > 0).
compute x1 = sqrt(-
2*ln(uniform(mc,nrow(mcmats))))&*cos((2*3.14
159265358979)*uniform(mc,nrow(mcmats))).
compute x1=x1*chol(mcmats).
loop i=1 to nrow(x1).
compute x1(i,:)=x1(i,:)+t(mccoeff).
end loop.
loop i = 1 to nmeds.
compute
indboot(2:nrow(indboot),i)=x1(:,i)&*x1(:,(i+nme
ds)).
end loop.
end if.
do if (ttt = 0 and model > 3).
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
end if.
do if (model <> 74).
do if (dichy = 0).
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 't', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 't', 'p', 'LLCI', 'ULCI'/format =
F10.4.
else.
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 'Z', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.

```

```

print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 'Z', 'p', 'LLCI', 'ULCI' /format =
F10.4.
end if.
end if.
end if.
do if (ttt > 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
compute clbs = {modvnm2, 'Effect', 'SE', 't', 'p',
'LLCI', 'ULCI'}.
compute tratio = (directv&/sedir).
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy))))).
compute outp = {modvalsd, directv, sedir,
tratio, p}.
do if (dichy = 0).
compute temp1=directv-sqrt(jncrit)*sedir.
compute temp2=directv+sqrt(jncrit)*sedir.
compute outp = {outp, temp1, temp2}.
end if.
do if (dichy = 1).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute temp = directv-abs(xp2)*sedir.
compute outp = {outp, temp}.
compute temp = directv+abs(xp2)*sedir.
compute outp = {outp, temp}.
compute clbs = {modvnm2, 'Effect', 'SE', 'Z', 'p',
'LLCI', 'ULCI'}.
end if.
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
print outp/title = 'Conditional direct effect(s) of
X on Y at values of the moderator(s):'/format =
F10.4 /cnames = clbs.
end if.
do if (nmods > 0 and model <> 5).
do if (model < 4).

```



```

print/title =
*****
*****!

compute zmat(1,1) = 1*(1-yes).
compute cfse = make(nrow(modvals),1,0).
loop #m = 1 to nrow(modvals).
do if (model = 1).
compute zmat(2,1)=modvals(#m,1)**2.
compute zmat(5,1)=2*modvals(#m,1).
end if.
do if (model = 2 or model = 3).
compute zmat(2,1)=modvals(#m,2)**2.
compute zmat(3,1)=modvals(#m,1)**2.
compute
zmat(4,1)=(modvals(#m,1)**2)*(modvals(#m,2)
**2)*(1-yes2).
compute zmat(5,1)=2*modvals(#m,2)*(1-yes).
compute zmat(6,1)=2*modvals(#m,1)*(1-yes).
compute
zmat(7,1)=2*modvals(#m,1)*modvals(#m,2)*(1-
yes).
compute
zmat(8,1)=2*modvals(#m,1)*modvals(#m,2).
compute
zmat(9,1)=2*modvals(#m,1)*(modvals(#m,2)**
2)*(1-yes2).
compute
zmat(10,1)=2*(modvals(#m,1)**2)*modvals(#m
,2)*(1-yes2).
end if.
compute cfse(#m,1)=sqrt(csum(zmat&*cmat)).
end loop.
end if.
do if (nmods > 0).
compute clbs = {modvnm, 'Effect'}.
loop im = 1 to nmeds.
compute obs = t(indboot(1+(im-1)*(boot+1),:)).
compute outp = {modvals, obs}.
do if (model < 4).
compute tstat = obs&/cfse.

```

```

do if (dichy = 0).
compute pval = 2*(1-tcdf(abs(tstat), (n-
ncol(xy)))).
compute temp=obs-sqrt(jncrit)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp=obs+sqrt(jncrit)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 't', 'p', 'LLCI', 'ULCI'}.
compute jnclbs=clbs.
end if.
do if (dichy = 1).
compute pval = 2*(1-cdfnorm(abs(tstat))).
compute temp = obs-abs(xp2)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp = obs+abs(xp2)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute jnclbs=clbs.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indboot((1+(im-
1)*(boot+1)),:).
compute indboot2 = indboot((2+(im-
1)*(boot+1)):(1+(im-1)*(boot+1)+boot),:).
compute mnind = t(csum(indboot2)/boot).
compute stdind=t(sqrt((cssq(indboot2)-
((csum(indboot2)**2)/boot))/(boot-1))).
compute llci=make(1,ncol(indboot2),-999).
compute ulci=make(1,ncol(indboot2),-999).
loop #e = 1 to ncol(indboot2).
.
compute temp = indboot2(:,#e).
compute temp(GRADE( indboot2(:,#e) )) =
indboot2(:,#e).
compute badlo = 0.
compute badhi = 0.

```

```

do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.

```

```

do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {modvals, obs, stdind, t(llci),
t(ulci)}.
compute clbs = {modvnm, 'Effect', 'Boot SE',
'BootLLCI', 'BootULCI'}.
end if.
compute mtemp = mnames(1, im).
compute rlbs =
make(nrow(modvals),1,mnames(1,im)).
do if (model < 4).
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
do if (yes=0).
print outp/title = 'Conditional effect of X on Y at
values of the moderator(s):'/cnames =
clbs/format = F10.4.
else.
compute outp=outp(:,3:ncol(outp)).
compute clbs=clbs(:,3:ncol(clbs)).
compute clbs(1,1)='Contrast'.
print outp/title = 'Contrast of conditional effects
of X on Y'/cnames = clbs/format = F10.4.
end if.
end if.
do if (model > 5 and mod74dic <> 1).
do if (im = 1).
print/title = 'Conditional indirect effect(s) of X on
Y at values of the moderator(s):'.
end if.
print outp/title = 'Mediator'/rnames =
rlbs/cnames = clbs/format = F10.4.

```

```

else if (mod74dic = 1 and model = 74).
do if (im = 1).
print/title = 'Indirect effect(s) of X on Y:'.
compute clbs3=clbs(1,2:ncol(clbs)).
end if.
compute outp3=outp(1,2:ncol(outp)).
print outp3/title = 'Mediator'/rnames =
rlbs/cnames = clbs3/format = F10.4.
end if.
end loop.
loop i = notes to 1 by -1.
do if (note(i,1) = 4 and yes=0).
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 5 and yes=0).
print/title = 'Values for quantitative moderators
are the mean and plus/minus one SD from
mean.'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 14 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    below the mean was replaced
with the minimum because one SD below the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.
do if (note(i,1) = 15 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    above the mean was replaced
with the maximum because one SD above the
mean'/space=0.

```

```

print/title = '      is outside of the range of the
data.'/space=0.
end if.
end loop.
do if (model = 3 and yes=0).
compute jnvals=make(nrow(matw),7,0).
compute jnvals(:,1)=matw.
compute jnvals(:,2)=jnb1+jnb3*jnvals(:,1).
compute
jnvals(:,3)=sqrt(jnsb1+2*jnvals(:,1)*jnsb1b3+(jn
vals(:,1)*jnvals(:,1))*jnsb3).
compute jnvals(:,4)=jnvals(:,2)/jnvals(:,3).
do if (dichy = 0).
compute jnvals(:,5)=2*(1-tcdf(abs(jnvals(:,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(:,5)=2*(1-
cdfnorm(abs(jnvals(:,4)))).
end if.
compute jnvals(:,6)=jnvals(:,2)-
sqrt(jncrit)*jnvals(:,3).
compute
jnvals(:,7)=jnvals(:,2)+sqrt(jncrit)*jnvals(:,3).
compute clbs={clbs(:,1),clbs(:,3:ncol(clbs))}.
do if (coeffci = 0).
compute jnvals =jnvals(:,1:(ncol(jnvals)-2)).
end if.
print jnvals/title = 'Conditional effect of X*M
interaction at values of W:.'/cnames
=clbs/format = F10.4.
end if.
do if (jn = 1 and (model = 1 or model = 3) and
jndich = 0 and yes=0).
print/title = '***** JOHNSON-
NEYMAN TECHNIQUE
*****'.
compute ajn =(jncrit*jnsb3)-(jnb3*jnb3).
compute bjn = 2*((jncrit*jnsb1b3)-(jnb1*jnb3)).
compute cjn = (jncrit*jnsb1)-(jnb1*jnb1).

```

```

compute radarg = (bjn*bjn)-(4*ajn*cjn).
compute den = 2*ajn.
compute nrts = 0.
do if (radarg >= 0 and den <> 0).
compute x21 = (-bjn+sqrt(radarg))/den.
compute x22 = (-bjn-sqrt(radarg))/den.
compute roots = 0.
do if (x21 >= jnmin and x21 <= jnmax).
compute nrts = 1.
compute roots = {roots; x21}.
end if.
do if (x22 >= jnmin and x22 <= jnmax).
compute nrts = nrts + 1.
compute roots = {roots; x22}.
end if.
compute roots={roots,make(nrow(roots),2,0)}.
compute modtemp=m.
do if (model=3).
compute modtemp=w.
end if.
do if (nrts > 0).
compute roots = roots(2:nrow(roots),1:3).
compute roots(1,2)=(csum(modtemp <
roots(1,1))/n)*100.
compute roots(1,3)=(csum(modtemp >
roots(1,1))/n)*100.
do if (nrow(roots)=2).
compute roots(2,2)=(csum(modtemp <
roots(2,1))/n)*100.
compute roots(2,3)=(csum(modtemp >
roots(2,1))/n)*100.
end if.
print roots/title = 'Moderator value(s) defining
Johnson-Neyman significance
region(s)'/clabels = 'Value', '% below', '%
above'/format = F10.4.
compute jnvals=make((21+nrts),7,0).
loop i= 0 to 20.
compute jnvals((i+1),1)=jnmin+(i*((jnmax-
jnmin)/20)).

```

```

end loop.
loop i = 1 to nrts.
loop j = 2 to nrow(jnvals).
do if ((roots(i,1) > jnvals((j-1),1)) and (roots(i,1)
< jnvals(j,1))).
compute
jnvals((j+1):(21+i),1)=jnvals(j:(20+i),1).
compute jnvals(j,1)=roots(i,1).
end if.
end loop.
end loop.
loop i = 1 to nrow(jnvals).
compute jnvals(i,2)=jnb1+jnb3*jnvals(i,1).
compute
jnvals(i,3)=sqrt(jnsb1+2*jnvals(i,1)*jnsb1b3+(jn
vals(i,1)*jnvals(i,1))*jnsb3).
compute jnvals(i,4)=jnvals(i,2)/jnvals(i,3).
do if (dichy = 0).
compute jnvals(i,5)=2*(1-tcdf(abs(jnvals(i,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(i,5)=2*(1-
cdfnorm(abs(jnvals(i,4)))).
end if.
compute jnvals(i,6)=jnvals(i,2)-
sqrt(jncrit)*jnvals(i,3).
compute
jnvals(i,7)=jnvals(i,2)+sqrt(jncrit)*jnvals(i,3).
end loop.
do if (model = 1).
print jnvals/title = 'Conditional effect of X on Y
at values of the moderator (M)'/cnames
=jnclbs/format = F10.4.
end if.
do if (model = 3).
compute
jnclbs={jnclbs(:,1),jnclbs(:,3:ncol(jnclbs))}.

```



```

print jnvals/title = 'Conditional effect of X*M on
Y at values of the moderator (W)'/cnames
=jnclbs/format = F10.4.
end if.
end if.
do if (nrts = 0).
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
else.
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
end if.
end if.
do if (model < 4 and (plot = 1)).
compute dataplot =
make((nrow(modvals)*nrow(matx)),(ncol(modv
als)+1),0).
compute tmp = 1.
loop i = 1 to nrow(modvals).
loop j = 1 to nrow(matx).
compute dataplot(tmp,.)={matx(j,1),
modvals(i,:)}.
compute tmp=tmp+1.
end loop.
end loop.
compute dataplot = {dataplot,
make(nrow(dataplot),(1+dichy),0)}.
compute dataplo2 = make(nrow(dataplot),1,1).
do if (model = 1).
compute dataplo2 = {dataplo2, dataplot(:,2),
dataplot(:,1), (dataplot(:,1)*dataplot(:,2))}.
end if.
do if (model = 2 or model = 3).
compute dataplo2 = {dataplo2, dataplot(:,3),
dataplot(:,1), (dataplot(:,1)*dataplot(:,3)),
dataplot(:,2), (dataplot(:,1)*dataplot(:,2))}.

```

```

do if (model = 3).
compute dataplo2 = {dataplo2,
(dataplot(:,2)*dataplot(:,3)),
(dataplot(:,1)*dataplot(:,2)*dataplot(:,3))}.
end if.
end if.
loop i = 1 to nrow(dataplot).
compute tmp=dataplo2(i,:).
do if (ncovs > 0).
compute tmp = {tmp, covmeans}.
end if.
do if (cluster > 0).
compute tmp = {tmp, cldmeans}.
end if.
compute dataplot(i,(ncol(dataplot)-
(dichy)))=tmp*coeffplt.
do if (dichy = 1).
compute
dataplot(i,(ncol(dataplot)))=exp(tmp*coeffplt)/(1
+exp(tmp*coeffplt)).
end if.
end loop.
compute clbs = {xname, modvnm, 'yhat'}.
do if (dichy = 1).
compute clbs = {xname, modvnm, 'ln(odds)',
'prob'}.
end if.
print/title =
*****
*****!

```

```

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (model = 1).
do if (dichy = 0).
print/title = 'DATA LIST FREE/Comm_Pract
Technological_Advan
Operational_Performance.' /space=1.
else if (dichy = 1).
print/title = 'DATA LIST FREE/Comm_Pract
Technological_Advan logodds prob.'
/space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
Operational_Performance BY
Technological_Advan.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH Operational_Performance BY
Comm_Pract.' /space=0.
end if.
end if.

```

```

do if (dichy = 1).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
logodds BY Technological_Advan.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
prob BY Technological_Advan.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Comm_Pract.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Comm_Pract.' /space=0.
end if.
end if.
end if.
do if (model = 2 or model = 3).
do if (dichy = 0).
print/title = 'DATA LIST FREE/Comm_Pract
Technological_Advan
Operational_Performance.' /space=1.
end if.
do if (dichy = 1).
print/title = 'DATA LIST FREE/Comm_Pract
Technological_Advan logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' /format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
Operational_Performance BY
Technological_Advan/PANEL ROWVAR='
/space=0.
end if.

```

```

do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH Operational_Performance BY
Comm_Pract/PANEL ROWVAR=' /space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
logodds BY Technological_Advan/PANEL
ROWVAR=' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
prob BY Technological_Advan/PANEL
ROWVAR=' /space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Comm_Pract/PANEL
ROWVAR=' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Comm_Pract/PANEL
ROWVAR=' /space=0.
end if.
end if.
end if.
do if (ncovs > 0).
print/title = '* Estimates are based on setting
covariates to their sample means.'.
end if.
end if.
end if.
do if ((imm > 0) or (model = 74 and
mod74dic=0) or ((model = 58 or model = 59)
and wvdich=1))).
do if (imm > 4 and imm < 8).

```

```

compute
indbootp={indbootp(:,1:nmeds),indboop1,indbo
op2}.
end if.
compute obsprod = t(indbootp(1,:)).
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indbootp(1,:).
compute indbootp = indbootp(2:(boot+1),:).
compute mnindp = t(csum(indbootp)/boot).
compute stdindp=t(sqrt((cssq(indbootp)-
((csum(indbootp)&**2)/boot))/(boot-1))).
compute llcip=make(1,ncol(indbootp),-999).
compute ulcip=make(1,ncol(indbootp),-999).
loop #e = 1 to ncol(indbootp).

.
compute temp = indbootp(:,#e).
compute temp(GRADE( indbootp(:,#e) )) =
indbootp(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).

```

```

compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcip(1,#e)=llcit.
compute ulcip(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {obsprod, stdindp, t(llcip),
t(ulcip)}.
compute clbs = {'Effect', 'SE(Boot)', 'BootLLCI',
'BootULCI'}.
do if (model = 8 or model = 12).
print/title='-----'.
compute outpimm=outp(1:nmeds,:).

```

```

print/title = 'Indirect effect of highest order
product:.'/space=0.
print outpimm/title = 'Mediator'/cnames =
clbs/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
compute clbs3 = {'Index', 'SE(Boot)',
'BootLLCI', 'BootULCI'}.
do if (imm < 5).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=outp((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
do if (imm > 4 and imm < 8).

```



```

print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'.

compute outpimm=outp(1:nmeds,:).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
compute
outpimm=outp((nmeds+1):nrow(outp),:).
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.

loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),clbs3}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3(1,1)=modvnm(1,1).
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.

```

```

print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
end if.
do if (boot = 0).
do if (model = 8 or model = 12).
print/title='-----'.
compute obsimm=obsprod(1:nmeds,:).
print/title = 'Indirect effect of highest order
product.'/space=0.
print obsimm/title = 'Mediator'/clabels =
'Effect'/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
do if ((imm > 0 and imm < 5) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=obsprod((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
end loop.
end if.

```

```

do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****!
.
compute outpimm=obsprod(1:nmeds,1).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
compute
outpimm=obsprod((nmeds+1):nrow(obsprod),:)
.
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),'Index'}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3={modvnm(1,1),'Index'}.
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator: '/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
end if.
end if.

```

```

end if.
compute conmake=0.
compute concols = 0.
do if ((model > 3 and model < 7) and (contrast
> 0) and nmods = 0 and nmeds > 1 ).
compute concols =
((ncol(indboot)*(ncol(indboot)-1))/2).
compute indcon=make(nrow(indboot),concols,-
999).
compute conkey = { ' ', ' ', ' ' }.
compute temp=1.
compute conmake=1.
loop i = 1 to (ncol(indboot)-1).
loop j = (i+1) to (ncol(indboot)).
compute indcon(:,temp)=indboot(:,i)-
indboot(:,j).
do if (contrast = 2).
compute indcon(:,temp)=abs(indboot(:,i))-
abs(indboot(:,j)).
end if.
do if (model <= 6).
compute conkey={conkey; mnames(1,i),
'minus', mnames(1,j)}.
end if.
do if (model = 6).
compute conkey={conkey; indlbl2(i,1), 'minus',
indlbl2(j,1)}.
end if.
compute temp=temp+1.
end loop.
end loop.
end if.
do if (model = 4 or model = 5).
compute clbs = {'Effect'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obs = {csum(obs); obs}.
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.

```

```

compute rlbs = {rlbs;
cntname(1:ncol(indcon),1)}.
end if.
compute outp = obs.
compute outp2=outp.
do if (effsize = 1 and dichy = 0).
compute
pmeff(:,1)=rsum(pmeff(:,2:ncol(pmeff))).
compute
rmeff(:,1)=rsum(rmeff(:,2:ncol(rmeff))).
compute
abpseff(:,1)=rsum(abpseff(:,2:ncol(abpseff))).
compute
abcseff(:,1)=rsum(abcseff(:,2:ncol(abcseff))).
compute eff = {pmeff, rmeff, abpseff, abcseff}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute eff = {eff, r245}.
compute r245obs = {r245(1,1);r245(1,1)}.
compute kappa2ob = kappa2(1,1).
end if.
compute pmobs = t(pmeff(1,1:(nmeds+1))).
compute rmobs = t(rmeff(1,1:(nmeds+1))).
compute psobs = t(abpseff(1,1:(nmeds+1))).
compute csobs = t(abcseff(1,1:(nmeds+1))).
do if (contrast = 0).
compute outp2 = {obs, psobs, csobs, pmobs,
rmobs}.
end if.
do if (contrast > 0).
compute obs2=obs(1:nrow(psobs),:).
compute outp2 = {obs2, psobs, csobs, pmobs,
rmobs}.
end if.
compute clbs = {'ab', 'ab_ps', 'ab_cs', 'ab/c',
'ab/c'}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute outp2 = {outp2, r245obs}.
compute clbs = {clbs, 'R-sq_med'}.

```

```

do if (boot = 0 and mc = 0).
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
do if (boot = 0 and mc = 0).
do if (nmeds = 1).
compute outp2 = outp2(2,:).
compute rlbs = rlbs(2,1).
end if.
do if (model=5).
compute outp2=outp2(:,1:3).
end if.
print outp2/title = 'Indirect effect(s) of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1 and nmeds >
1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0 or mc > 0).
compute indboot = {rsum(indboot), indboot}.
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
compute ones = make(bootSZ,1,1).
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(bootSZ+1),:).
compute mnind = t(csum(indboot)/bootSZ).

```

```

compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/bootSZ))/(bootSZ-1))).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).

.
compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.

```

```

end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llici(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
do if (effsize = 1 and dichy = 0 and boot > 0).
compute estmte=eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).
.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.

```



```

compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).

```

```

compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
end if.
end if.
do if (boot > 0 or mc > 0).
compute outp = {obs, stdind, t(llci), t(ulci)}.
do if (nmeds = 1).
compute outp = outp(2,:).
compute rlbs = rlbs(2,1).
end if.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
do if (mc > 0).
compute clbs = {'Effect', 'MC SE', 'MC LLCI',
'MC ULCI'}.
end if.
print outp/title = 'Indirect effect of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (dichy = 0 and effsize = 1 and boot > 0).
compute outp = {psobs,
stdindf((2*(nmeds+1)+1):(3*(nmeds+1)),1),
t(llcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))),
t(ulcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Partially standardized indirect
effect of X on Y'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute outp = {csobs,
stdindf((3*(nmeds+1)+1):(4*(nmeds+1)),1),
t(llcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))),
t(ulcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.

```

```

print outp/title = 'Completely standardized
indirect effect of X on Y'/rnames = rlbs/cnames
= clbs/format = F10.4.
compute outp = {pmobs,
stdindf(1:(nmeds+1),1),t(llcif(1,1:(nmeds+1))),
t(ulcif(1,1:(nmeds+1))) }.
do if (model = 4).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
compute outp = {rmobs, stdindf(((nmeds+1)+1)
:(2*(nmeds+1)) ,1),
t(llcif(1,((nmeds+1)+1):(2*(nmeds+1)))),
t(ulcif(1,((nmeds+1)+1):(2*(nmeds+1))))}.
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
do if (nmeds = 1 and cluster = 0 and ncovs = 0
and model = 4).
compute r245obs = r245obs(1,1).
compute outp = {r245obs,
stdindf((4*(nmeds+1)+1):(4*(nmeds+1)+1),1),
t(llcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1))),
t(ulcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1)))}.
print outp/title = 'R-squared mediation effect
size (R-sq_med)'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
end if.
do if (normal = 1).

```

```

do if (nmeds = 1).
print sobel/title = 'Normal theory tests for
indirect effect'/clabels = 'Effect', 'se', 'Z',
'p'/format = F10.4.
end if.
do if (nmeds > 1).
compute rlbs2 = rlbs(2:nrow(rlbs),1).
print sobel/title = 'Normal theory tests for
specific indirect effects'/rnames = rlbs2/clabels
= 'Effect', 'se', 'Z', 'p'/format = F10.4.
end if.
end if.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
do if (model = 6).
compute clbs = {'eff'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obsfsum=csum(obs).
compute obs = {obsfsum; obs}.
compute indlbl = indlbl(1:nrow(obs),1).
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute indlbl = {indlbl;
cntname(1:ncol(indcon),1)}.
end if.
compute obs2=obs.
do if (boot = 0).
do if (dichy = 0 and effsize = 1).
compute clbs = {'eff', 'eff_ps', 'eff_cs', 'eff/c',
'eff/c'}.
do if (ncovs = 0).

```

```

compute obs = {obs, obs/stddevy,
(obs*stddevx/stddevy), obs/ctot,
obs/mmpaths(nrow(mmpaths),1)}.
end if.
do if (ncovs > 0 or clsdmy > 0).
compute obs = {obs, obs/sdycov,
(obs*sdxcov/sdycov),
obs/(obsfsum+mmpaths(nrow(mmpaths),1)),
obs/mmpaths(nrow(mmpaths),1)}.
end if.
compute obs2=obs.
do if (contrast > 0).
compute obs2 = obs(1:(nrow(obs)-concols),:).
end if.
end if.
print obs2/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute indboot = {rsum(indboot), indboot}.
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(boot+1),:).
compute mnind = t(csum(indboot)/boot).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/boot))/(boot-1))).
compute temp = nrow(indboot).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).

```

```

loop #e = 1 to ncol(indboot).

.

compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.

```

```

do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute obs = {obs, stdind, t(llci), t(ulci)}.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
print obs/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (effsize = 1 and dichy = 0).
compute indboot=indboot(:,1:(ncol(indboot)-
concols)).
compute eff = {rsum(abpseff), abpseff,
rsum(abcseff), abcseff, rsum(pmeff), pmeff,
rsum(rmeff), rmeff}.
compute effobs = eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).
.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).

```

```

compute badlo = 0.
compute badhi = 0.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(effobs(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.

```



```

compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute temp2 = stdindf(1:ncol(indboot),1).
compute temp3 = effobs(:,1:ncol(indboot)).
compute templo = llcif(1,1:ncol(indboot)).
compute temphi = ulcif(1,1:ncol(indboot)).
compute outp = {t(temp3), temp2,t(templo),
t(temphi)}.
print outp/title = 'Partially standardized indirect
effect of X on Y'/cnames = clbs/rnames =
indlbl/format = F10.4.
compute temp2 =
stdindf((ncol(indboot)+1):(2*ncol(indboot)),1).
compute temp3 =
effobs(:,(ncol(indboot)+1):(2*ncol(indboot))).
compute templo =
llcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute temphi =
ulcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templo),
t(temphi)}.
print outp/title = 'Completely standardized
indirect effect of X on Y'/cnames = clbs/rnames
= indlbl/format = F10.4.
compute temp2 =
stdindf((2*(ncol(indboot))+1):(3*ncol(indboot)),1
).
compute temp3 =
effobs(:,(2*(ncol(indboot))+1):(3*ncol(indboot)))
.

```

```

compute templow =
  llcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute temphi =
  ulcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
  t(temphi)}.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
compute temp =
  eff(:,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temp2 =
  stdindf((3*(ncol(indboot))+1):(4*ncol(indboot)),1
  ).
compute temp3 =
  effobs(:,(3*(ncol(indboot))+1):(4*ncol(indboot)))
  .
compute templow =
  llcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temphi =
  ulcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
  t(temphi)}.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
end if.
end if.
do if (nmeds = 2).
compute effkey = {xname, '->', mnames(1,1), '-
->', yname, '', ''}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, '', ''}.
compute effkey = {indlbl(2:4,1), effkey}.
end if.
do if (nmeds = 3).
compute effkey = {xname, '->' , mnames(1,1), '-
->', yname, '', '', ''}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, '',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, '',' ',' ',' '}.
compute effkey = {indbl(2:8,1), effkey}.
end if.
do if (nmeds = 4).
compute effkey = {xname, '->' , mnames(1,1), '-
>', yname, '',' ',' ',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,4), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname, '',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,4), '->', yname, '',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, '',' '}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', mnames(1,4), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,4), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', mnames(1,4), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,4), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {indl(2:16,1), effkey}.
end if.
print effkey/title = 'Indirect effect key'/format =
A8.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
end if.
DO IF (medte2=1).
compute
name=t(mdlms2(2:nrow(mdlms2),1)).
compute tmpdat=datayed(:,2:ncol(datayed)).
compute dd=tmpdat(:,1).

```

```

compute
dd={dd,tmpdat(:,(2+nmeds)),tmpdat(:,2:(1+nmeds))}.
compute stratify=0.
do if (ncovs > 0).
compute cname = cnames.
compute
dd={dd,tmpdat(:,(3+nmeds):(2+nmeds+ncovs))}.
end if.
compute method = 1.
compute samples = boot.
do if (mc > 0).
compute samples = mc.
compute method=2.
end if.
compute omnibus = 1.
compute booterr = 0.
do if (mcx < 0 or mcx > 4).
compute mcx = 1.
end if.
compute ovals = ncol(design(dd(:,1))).
compute nte = notes.
compute criterr = 0.
compute xskip = 0.
compute dumok = 0.
compute badend = 0.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute priorlo = -9999999.
compute priorhi = 9999999.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute reps = samples.
do if (contrast = 1).
compute note(nte,1) = 26.
compute nte=nte+1.

```

```

end if.
do if (effsize = 1).
compute note(nte,1) = 27.
compute nte=nte+1.
end if.
do if (normal = 1).
compute note(nte,1) = 28.
compute nte=nte+1.
end if.
do if (method = 2).
do if (omnibus = 1).
compute note(nte,1) = 25.
compute nte=nte+1.
end if.
compute samples = 0.
do if (reps < 1000).
compute reps = 1000.
end if.
end if.
do if (method = 1).
compute samples = reps.
end if.
compute temp = ncol(dd).
compute n = nrow(dd).
compute ones = make(n,1,1).
compute temp=dd(:,2).
compute nxd = ncol(temp).
compute nx = nxd.
do if (nx = 1 and mcx > 0).
compute temp = dd.
compute temp(GRADE(dd(:,2)),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,2)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute mnvls = cmin(t(nnvls)).
do if (mnvls < 2).
compute errs = errs + 1.
compute runerrs(errs,1) = 37.
compute criterr = 1.

```

```

end if.
do if (nvls > 9).
compute errs = errs+1.
compute runerrs(errs,1) = 36.
compute criterr = 1.
end if.
do if (criterr = 0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,2).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,2) <> nnvls((temp-1),1)).
compute nnvls(temp,1)=dd(i,2).
compute temp = temp+1.
end if.
end loop.
do if (mcx > 0).
compute x = dummy(:,2:ncol(dummy)).
compute nx = ncol(x).
compute minus1 = make(1,ncol(x),-1).
compute xdes=make((nx+1),3,0).
compute xdes(1,1)=dd(1,2).
compute xdes(1,2)=1.
compute temp = 2.
loop k = 2 to n.
do if (dd(k,2) <> dd((k-1),2)).
compute xdes(temp,2) = k.
compute xdes(temp,1) = dd(k,2).
compute xdes((temp-1),3) = k-1.
compute temp=temp+1.
end if.
end loop.
compute xdes((temp-1),3)=n.
compute xdes = {xdes, (xdes(:,3)-xdes(:,2)+1)}.
do if (stratify <> 0).
compute note(nte,1) = 7.
compute nte=nte+1.
end if.
do if (mcx = 4).

```

```

loop k = 1 to n.
do if (rsum(x(k,:)) = 0).
compute x(k,:) = minus1.
end if.
end loop.
end if.
do if (mcx = 2 or mcx = 3).
compute note(nte,1)=4.
compute nte = nte+1.
loop k = 1 to n.
do if (rsum(x(k,:)) > 0).
loop i = 1 to ncol(x).
do if (x(k,i) = 0).
compute x(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcx = 3).
compute conmat1={-8,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1; 0,0,0,0,0,-
3,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.
loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,:)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute x(k,:)=conmat1((rsum(x(k,:))+1),:).
end loop.
end if.
end if.
compute xdata = x.
compute xskip = 1.
compute xname = ddd(1,1:nx).
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).

```



```

compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcx = 4).
compute dummat(1,:) = minus1.
end if.
do if (mcx = 2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcx = 3).
compute dummat=conmat1.
end if.
compute dummat={nnvls, dummat}.
compute xname2={name(1,2), xname}.
end if.
end if.
end if.
compute nm=nmeds.
compute r2b = make(nm,1,-999).
compute indeff = make((nx*nm),1,-999).
compute cov =
make((nx*nm+nm),(nx*nm+nm),0).
compute mns=make(nrow(cov),1,0).
compute ny = ncol(dd)-nm-nxd-ncovs.
compute y = dd(:,1).
compute ydata = y.
compute yname = name(1,1).
do if (xskip = 0).
compute x = dd(:,2:(1+nxd)).
compute xdata = x.
compute xname = name(1,2:(1+nxd)).
compute indlbs = t(xname).
end if.
compute m = dd(:,(2+nxd):(1+nxd+nm)).
compute mdata = m.

```

```

compute mlab = { ' M1 = ';' ' M2 = ';' ' M3 = ';' '
M4 = ';' ' M5 = ';' ' M6 = ';' ' M7 = ';' ' M8 = ';' '
M9 = ';' ' M10 = '}.
compute mlab = {mlab; ' M11 = ';' ' M12 = ';' '
M13 = ';' ' M14 = ';' ' M15 = '}.
compute mname =
name(1,(2+nxd):(1+nxd+nm)).
do if (ncovs > 0).
compute c =
dd(:,(2+nxd+nm):(1+nxd+nm+ncovs)).
compute x = {x, c}.
compute xdata = x.
compute cdata = c.
compute xname = {xname, cname}.
end if.
compute nms={yname; t(mname);
t(xname(1,1:(ncol(xname)-ncovs)))}.
do if (dumok = 1).
compute nms={yname; t(mname); name(1,2)}.
end if.
do if (nm < 16).
do if (dumok = 1).
print dummat/title = 'Coding of categorical X
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
compute
boots=make((samples+1),((nx+1)*nm),-999).
do if (errs = 0 and criterr = 0).
loop k = 1 to (samples+1).
do if (k = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef=0.
do if (k > 1).
compute u=trunc(uniform(n,1)*n)+1.
compute xdata = x(u,:).
compute mdata = m(u,:).

```

```

compute ydata = y(u,:).
do if (ncovs > 0).
compute cdata = c(u,:).
end if.
end if.
loop i = 1 to nm.
compute start = 1+(i-1)*nx.
compute xmat = {ones, xdata}.
do if (covmy=2).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = mdata(:,i).
compute amat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*amat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-(((1-r2)*(n-1))/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute r2cha = r2.
compute adjr2cha = adjr2.
do if (ncovs > 0 and covmy <> 2).
compute xmat = {ones, cdata}.
compute atmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*atmat)**2).
compute r2step1 = (sstotal-ssresid2)/sstotal.
compute r2cha = r2-r2step1.
compute adjr2cha = adjr2- (1-(((1-r2step1)*(n-
1))/(n-ncol(xmat)))).
compute xmat = {ones, xdata}.
end if.
compute bootcoef={bootcoef,t(amat)}.
compute r2b(i,1)=adjr2cha.
do if (k = 1).

```

```

compute hcinvtX=inv(t( xmat )' xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,1)*hcinvtX*t( xmat
(i3,1)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvtX*t( xmat )' xmat
*hcinvtX
.
compute tratio = amat&/sqrt(diag(covmat)).
compute rnms = mname(1,i).
print rnms/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.
compute lmat = ident(nrow(amat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*amat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*amat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/'format = F10.4
/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'.

```

```

compute op = {amat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*(xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute cov(start:(start+(nx-
1)),start:(start+(nx-
1)))=covmat(2:(ncol(covmat)-(ncovs*(covmy <>
2))),2:(ncol(covmat)-(ncovs*(covmy <> 2)))).
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns(start:(start+(nx-
1)),1)=amat(2:(1+nx)).
end loop.
compute xmat = {ones,mdata,xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (k = 1).
print yname/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.

```

```

do if (ovals = 2).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis: '/cnames = nmsd/format = F9.2.
end if.
end if.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(yamat)/n)).
compute LL3 = yamat*&ln(pt2)+(1-yamat)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xamat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xamat.
compute ylp= yamat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).

```

```

compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute bootcoef={bootcoef,t(bmat)}.
compute LL1 = LL2.
compute covmat=varb.
end if.
do if (ovals <> 2).
compute bmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*bmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-(((1-r2)*(n-1))/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute bootcoef={bootcoef,t(bmat)}.
end if.
do if (k = 1).
do if (ovals = 2).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.

```

```

compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).
compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms2/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute
opdirsum=outp((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
end if.
do if (ovals <> 2).
.
compute hcinvXtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtXt*( xmat
(i3,:)).

```



```

end loop.
loop i3=1 to k3.
compute xmat(:,i3) = ( resid(:,ncol( resid
))&/(1-h))&* xmat(:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat(:,i3) = sqrt( mse )&* xmat(:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(bmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/(ncol(xmat)-1).
compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat((2+nmeds):(2+nmeds+(nx-1)),:)=
lmat2.
compute fratiodr =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/nx.
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute df2=(n-ncol(xmat)).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), df2, pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = bmat&/sqrt(diag(covmat)).
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute op = {bmat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat)))) }.

```

```

compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp))))*(xp2/(temp-
(2/3)+(.11/temp)))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms2/format =
F10.4.
compute
opdirsum=op((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
compute xmat = {ones,mdata}.
do if (ncovs > 0).
compute xmat = {xmat, cdata}.
end if.
compute btmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*btmat)**2).
compute r2direct = r2-((sstotal-
ssresid2)/sstotal).
compute pfr = 1-fcdf(fratiodr,(nx),(n-1-nx-nm-
ncovs)).
compute r2direct = {r2direct, fratiodr, nx, (n-1-
nx-nm-ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms2/cnames=rnms2/form
at= F10.4.
end if.
end if.

```

```

do if (k = 1 and toteff = 1).
print yname/title = '*****'
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:.'/format = A8.
compute xmat = {ones, xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(ymat)/n)).
compute LL3 = ymat*&ln(pt2)+(1-ymat)*&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= ymat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*ulp)*t(xlp)*(ulp-
pt1lp).
compute pt1lp = 1/(1+exp(-(ulp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.

```

```

compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute LL1 = LL2.
compute covmat=varb.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(bmat)-
1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms = {'Constant'; t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).

```

```

compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute optotsum=outp(2:(1+nx),:).
end if.
do if (ovals <> 2).
compute cmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*cmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).

.

compute hcinvXtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).

```

```

loop i3=1 to k3.
compute xmat(:,i3) = sqrt( mse )&* xmat(:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(cmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = cmatrix/sqrt(diag(covmat)).
compute op = {cmatrix, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute optotsum=op(2:(1+nx),:).
compute ssresid2 = ssresid.
compute r2total = r2.

```

```

compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat(2:(1+nx),:) = lmat2.
compute fratio3 =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/nx.
do if (ncovs > 0).
compute xmat = {ones, cdata}.
compute ctmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*ctmat)**2).
compute r2total = r2-((sstotal-ssresid2)/sstotal).
end if.
compute pfr = 1-fcdf(fratio3,(nx),(n-1-nx-
ncovs)).
compute r2tot2 = {r2total, fratio3, nx, (n-1-nx-
ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns((nrow(mns)-
nm+1):nrow(mns),1)=bmat(2:(1+nm),1).
compute tmp = 1.
loop i = 1 to (nm*nx).
compute
boots(k,i)=mns(i,1)*mns(((nm*nx)+tmp),1).
do if ((i/nx)=trunc(i/nx)).
compute
boots(k,((nm*nx)+tmp))=r2b(tmp,1)*mns(((nm*
nx)+tmp),1).
compute tmp = tmp+1.
end if.
end loop.
do if (savboot = 1 and k > 1).

```

```

compute bootstrp((k-1),:)=bootcoef.
end if.
end loop.
do if (method = 1).
compute llci=make(1,ncol(boots),-999).
compute ulci=make(1,ncol(boots),-999).
compute indeff = t(boots(1,1:(nm*nx))).
compute indeff22=t(boots(1,:)).
compute omni = t(boots(1,((nm*nx)+1) :
ncol(boots))).
do if (nrow(boots) > 1).
compute boots = boots(2:nrow(boots),:).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
loop i = 1 to ncol(boots).
.
compute temp = boots(:,i).
compute temp(GRADE( boots(:,i) )) = boots(:,i).
compute badlo = 0.
compute badhi = 0.
do if ( indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(indeff22(i,1)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.

```



```

do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,i)=llcit.
compute ulci(1,i)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute llci = t(llci).
compute ulci= t(ulci).
compute llci={llci,ulci}.
end if.
end if.
do if (method = 2).
compute tmp=1.
loop i = 1 to (nm*nx).
compute
indeff(i,1)=mns(i,1)&*mns(((nm*nx)+tmp),1).

```

```

do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = sqrt(-
2*ln(uniform(reps,nrow(cov))))&*cos((2*3.1415
9265358979)*uniform(reps,nrow(cov))).
compute boots=boots*chol(cov).
loop i = 1 to ncol(boots).
compute boots(:,i)=boots(:,i)+mns(i,1).
end loop.
compute tmp=1.
loop i = 1 to (nm*nx).
compute
boots(:,i)=boots(:,i)&*boots(:,((nm*nx)+tmp)).
compute temp = boots(:,i).
compute temp(GRADE(boots(:,i))) = boots(:,i).
compute boots(:,i) = temp.
do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = boots(:,1:(nm*nx)).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
compute llci =
t(({boots(cilow,:)};{boots(cihigh,:)})).
end if.
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
compute rnmssum=rnms(2:(1+nx),1).
do if (ovals <> 2).
print optotsum/title = 'Relative total effects of X
of Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.

```

```

print r2tot2/title = 'Omnibus test of total effect of
X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print optotsum/title = 'Relative total effects of X
on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
end if.
compute
rnmssum=rnms2((2+nmeds):(1+nmeds+nx),1).
do if (ovals <> 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.
print r2direct/title = 'Omnibus test of direct
effect of X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
do if (omnibus = 1 and (method = 1 or method
= 3)).
compute indlbs = {indlbs; 'Omnibus'}.
end if.
loop i = 1 to nm.
compute op = mname(1,i).
print op/title = 'Relative indirect effect(s) of X on
Y through:.'/format = A8/space=0.
do if (method = 2).
compute clbs = {'Effect', 'SE MC', 'MC LLCI',
'MC ULCI'}.
compute op = {indeff, bootse(1:(nm*nx)), lici}.
end if.

```

```

do if (method = 1).
compute clbs = {'Effect', 'SE(boot)', 'LLCI',
'ULCI'}.
compute op = indeff.
do if (samples > 0).
compute op = {indeff, bootse(1:(nm*nx),:),
llci(1:(nm*nx),:)}).
end if.
end if.
compute temp = op((((i-1)*nx)+1):(nx*i),:).
do if ((method = 1) and omnibus = 1).
do if (samples = 0).
compute temp={temp;omni(i,1)}.
end if.
do if (samples > 0).
compute temp = {temp; omni(i,1),
bootse(((nm*nx)+i),:), llci(((nm*nx)+i),:)}).
end if.
end if.
print temp/title = '/rnames = indlbs/cnames =
clbs/format = F10.4 /space=0.
print/title = '-----'.
end loop.
end if.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
END IF.
DO IF (mdpbe=1).
compute nm={yname,xname,mnames}.
do if (ncovs > 0).

```

```

compute
nm={yname,cnames,xname,mnames}.
end if.

compute tpx=xname.
compute x = dd(:,2:ncol(dd)).
compute tpy = yname.
compute tempy= dd(:,1).
compute n = nrow(dd).
compute nx=1.
compute criterr=0.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute ddd1 = {'int_1', 'int_2', 'int_3', 'int_4',
'int_5', 'int_6', 'int_7', 'int_8', 'int_9'}.
compute dumok = 0.
compute mcfoc=mcx.
compute mcmmod=mcm.
compute mcloc=(mcfoc>0).
compute con = make(n,1,1).
compute ncovs=ncol(x)-2.
do if (mcfoc > 0 or mcmmod > 0).
compute temp = dd.
compute temp(GRADE(dd(:,(ncol(dd)-
mcloc))),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,ncol(dd)-
mcloc)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute toosmall=rsum(nnvls < 2).
compute mnvls = cmin(t(nnvls)).
do if ((rsum(nnvls < 2)) > 0).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=37.
end if.
do if (nvls > 10).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=36.

```

```

end if.
do if (criterr=0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,ncol(dd)-mcloc).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,ncol(dd)-mcloc) <> nnvls((temp-
1),1)).
compute nnvls(temp,1)=dd(i,ncol(dd)-mcloc).
compute temp = temp+1.
end if.
end loop.
compute dummy = dummy(:,2:ncol(dummy)).
do if (mcfoc=4 or mcmmod=4).
compute minus1=make(1,ncol(dummy),-1).
loop k = 1 to n.
do if (rsum(dummy(k,:)) = 0).
compute dummy(k,:)=minus1.
end if.
end loop.
end if.
do if (mcfoc = 2 or mcfoc = 3) or (mcmmod = 2 or
mcmmod = 3)).
loop k = 1 to n.
do if (rsum(dummy(k,:)) > 0).
loop i = 1 to ncol(dummy).
do if (dummy(k,i) = 0).
compute dummy(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcfoc = 3 or mcmmod=3).
compute conmat1={-8,1,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1,1; 0,0,0,0,0,-
3,1,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.

```

```

loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,:)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute
dummy(k,:)=conmat1((rsum(dummy(k,:))+1),:).
end loop.
end if.
end if.
compute nx = ncol(dummy).
compute xname = ddd(1,1:nx).
compute xdata = dummy.
compute xname = ddd(1,1:nx).
compute xname2={nm(1,(ncol(nm)-mcloc)),
xname}.
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcfoc = 2 or mcmmod=2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcfoc = 3 or mcmmod=3).
compute dummat=conmat1.
end if.
do if (mcfoc = 4 or mcmmod = 4).
compute dummat(1,:)=minus1.
end if.
compute dummat={nnvls, dummat}.
end if.
end if.
do if (criterr=0).
compute y = dd(:,1).
compute ovals = ncol(design(dd(:,1))).

```

```

compute itprob = 0.
do if (ovals = 2).
compute omx = cmax(y(:,1)).
compute omn = cmin(y(:,1)).
compute y(:,1) = (y(:,1) = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute sstotal = csum((y-(csum(y)/n))&**2).
compute outv = t(nm(1,1)).
compute mdtr = nm(1,ncol(dd)).
compute fciv = nm(1,(ncol(dd)-1)).
do if (mcfoc <> 0 or mcmod <> 0).
compute inter=make(n,nx,0).
loop i = 1 to nx.
compute inter(:,i)=dummy(:,i)&*dd(:,ncol(dd)-
(1-mcloc)).
end loop.
do if (ncovs = 0).
compute x={con,dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
do if (ncovs > 0).
compute
x={con,dd(:,2:(1+ncovs)),dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={t(nm(1,2:(1+ncovs)));
nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
end if.
compute xmns = csum(x)&/n.
compute focvals = {1,0,0}.
compute highwarn = 0.
compute lowwarn = 0.

```



```

do if (ncol(x) > (2*nx+2)).
compute covmns = {1, xmns(1,2:(ncol(x)-1-
(2*nx)))}.
compute focvals = {covmns,0,0}.
end if.
compute dfres = n-ncol(x).
do if (ovals = 2).
compute pt2 = make(n,1,(csum(y)/n)).
compute LL3 = y*&ln(pt2)+(1-y)&ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(x),1,0).
compute LL1 = 0.

.

compute pt1lp= pt1.
compute xlp= x.
compute ylp= y.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).

```

```

compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
else if (ovals <> 2).
compute invXtX=inv(t(x)*x).
compute b = invXtX*t(x)*y.
compute resid = y-(x*b).
compute ssresid=csum(resid&*resid).
compute msresid=ssresid/dfres.
compute varb = msresid*invXtX.
compute
seint=sqrt(varb(nrow(varb),nrow(varb))).
compute k3 = nrow(b).
do if (hc3 = 1).
compute xhc=x.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)= xhc(i3,:)*invXtX*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).
end loop.
compute varb=(invXtX*t(xhc)*xhc*invXtX).
end if.
compute seb = sqrt(diag(varb)).
compute r2 = 1-(ssresid/sstotal).

```

```

do if (mcfoc <> 0 or mcmod <> 0).
compute ytp=y.
compute xtp=x(:,1:(ncol(x)-nx)).
compute btp = inv(t(xtp)*xtp)*t(xtp)*ytp.
compute residt = ytp-(xtp*btp).
compute ssresidt=csum(residt*residt).
compute r2noint = 1-(ssresidt/sstotal).
end if.
compute pr = ncol(x)-1.
compute lmat = ident(nrow(b)).
compute lmat = lmat(:,2:ncol(lmat)).
compute f =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
pr).
compute pf = 1-fcdf(f,pr,dfres).
compute pf = {sqrt(r2),r2,msresid,f,pr,dfres,pf}.
compute xd=abs(xp2).
end if.
do if (itprob = 0).
compute tstat = b&/seb.
do if (ovals <> 2).
compute p = 2*(1-tcdf(abs(tstat), dfres)).
else if (ovals = 2).
compute p = 2*(1-cdfnorm(abs(tstat))).
end if.
compute outp = {b,seb,tstat,p}.
compute nms = {'constant'; nms; 'interact'}.
do if (ovals <> 2).
compute tval = (dfres* (exp((dfres-
(5/6))*((xd/(dfres-(2/3)+(.11/dfres)))*(xd/(dfres-
(2/3)+(.11/dfres)))))-1)).
compute outp={outp,(b-
sqrt(tval)&*seb),(b+sqrt(tval)&*seb)}.
end if.
compute bb = tval.
do if (dumok = 1).
do if (mcfoc > 0).
print dummat/title = 'Coding of categorical X
variable for analysis:'/cnames = xname2/format
= F5.2.

```

```

else if (mcmod > 0).
print dummat/title = 'Coding of categorical M
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
print yname/title =
*****
*****'/rlabels = 'Outcome: '/format
= A8.
do if (ovals = 2).
compute nmsd = {outv, 'Analysis'}.
print rcd/title = 'Coding of binary Y for
analysis: '/cnames = nmsd/format = F10.4.
compute outp = {outp,(outp(:,1)-
sqrt(tval)&*outp(:,2)),(outp(:,1)+sqrt(tval)&*outp
(:,2))}.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
print outp/title = 'Logistic Regression
Model'/rnames = nms/clabels 'Coeff' 'se' 'Z' 'p'
'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
compute varbtmp=varb.
compute btmp=b.

```

```

compute LL2f=LL2.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=x(:,1:(ncol(x)-nx)).
compute bt1 = make(ncol(xlp),1,0).

.

compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)*&ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.

```

```

compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL2f.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(LLdiff,nx).
compute rcha={LLdiff,nx,pchi}.
end if.
do if (ovals <> 2).
print pf/title = 'Model Summary'/clabels = 'R' 'R-
sq' 'MSE' 'F' 'df1' 'df2' 'p'/format F10.4.
compute lmat=make(nrow(b),1,0).
compute lmat(nrow(lmat),1)=1.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
1).
compute
rcha=((b(nrow(b),1)/seint)*(b(nrow(b),1)/seint))*
(1-r2)/dfres.
compute rcha = {rcha, fcha, 1, dfres,
outp((pr+1),4)}.
print outp/title = 'Model'/rnames = nms/clabels
'coeff' 'se' 't' 'p' 'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
do if (mcfoc > 0 or mcmod > 0).
compute rcha=r2-r2noint.
compute lmat=make((nrow(b)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
nx).
compute pvalr2c=1-fcdf(fcha,nx,dfres).

```

```

compute rcha = {rcha, fcha, nx, dfres,pvalr2c}.
end if.
end if.
do if (mcfoc > 0 or mcmod > 0).
compute intkey = {'a', 'b', 'c', 'd', 'e'}.
loop i = 1 to nx.
compute intkey={intkey; ddd1(1,i), ' : ', ddd(1,i),
' X ', nm(1,ncol(dd)-(1-mcloc))}.
end loop.
compute intkey=intkey(2:nrow(intkey),:).
print intkey/title='Product terms key:'/format =
a8.
end if.
do if (ovals <> 2).
print rcha/title = 'R-square increase due to
interaction:.'/clabels 'R2-chng' 'F' 'df1' 'df2'
'p'/format F10.4.
else if (ovals = 2).
print rcha/title = 'Likelihood ratio test for
interaction:.'/clabels 'Chi-sq' 'df' 'p'/format F10.4.
end if.
print/title =
*****
*****'.
compute mdvar = x(:,(ncol(x)-1)).
do if (mcfoc = 0 and mcmod = 0).
compute g1 = b((ncol(x)-2),1).
compute g3 = b(ncol(x),1).
compute vg1 = varb((ncol(x)-2),(ncol(x)-2)).
compute vg3 = varb(ncol(x),ncol(x)).
compute covg1g3 = varb((ncol(x)-2), ncol(x)).
end if.
do if (mcfoc > 0 or mcmod > 0).
compute mdvar = x(:,(ncol(x)-(2*nx))).
end if.
compute mdmin = cmin(mdvar).
compute mdmax = cmax(mdvar).
compute fvar = x(:,(ncol(x)-2)).
compute nval = ncol(design(mdvar)).
compute fvmin = cmin(fvar).

```

```

compute fvmax = cmax(fvar).
do if (mmodval = 999).
compute mnmd = csum(mdvar)/n.
compute tmp = make(n,1,mnmd).
compute sdmd = sqrt(csum(((mdvar-
tmp)**2))/(n-1)).
compute probeval = {mnmd-sdmd; mnmd;
mnmd+sdmd}.
do if (probeval(1,1) < mdmin).
compute lowwarn = 1.
compute probeval(1,1)=mdmin.
end if.
do if (probeval(nrow(probeval),1) > mdmax).
compute highwarn = 1.
compute probeval(nrow(probeval),1)=mdmax.
end if.
do if (quantile = 1).
compute tmp = mdvar.
compute tmp(GRADE(mdvar(:,1)),:)= mdvar.
compute mdvar = tmp.
compute
probeval={mdvar(trunc(0.1*n),1);mdvar(trunc(0
.25*n),1);mdvar(trunc(0.5*n),1);mdvar(trunc(0.7
5*n),1);mdvar(trunc(0.9*n),1)}.
end if.
end if.
do if (nval = 2).
compute probeval = make(2,1,0).
compute probeval(1,1)=cmin(mdvar).
loop i = 1 to n.
do if (mdvar(i,1) <> probeval(1,1)).
compute probeval(2,1) = mdvar(i,1).
BREAK.
end if.
end loop.
end if.
do if (mmodval <> 999).
compute probeval = mmodval.
end if.
compute outp = make(nrow(probeval),7,0).

```



```

do if (mcfoc > 0 or mcmmod > 0).
compute focvals=make(1,ncol(x)+1,1).
do if (mcfoc > 0).
print/title = 'Conditional effect of X on Y at
values of the moderator:'.
print/title = ' '/space=0.
compute rnn2=mdtr.
compute matt=make(nx,6,0).
end if.
loop jj=1 to nrow(probeval).
do if (mcfoc > 0).
loop ii=1 to nx.
compute g1=b((ncol(x)-(2*nx)+ii),1).
compute g3=b((ncol(x)-nx+ii),1).
compute vg1=varb((ncol(x)-(2*nx)+ii),(ncol(x)-
(2*nx)+ii)).
compute vg3=varb((ncol(x)-nx+ii),(ncol(x)-
nx+ii)).
compute covg1g3=varb((ncol(x)-
(2*nx)+ii),(ncol(x)-nx+ii)).
compute x2 = probeval(jj,1).
compute w1 = g1+g3*x2.
compute varw1 =
vg1+(2*x2*covg1g3)+((x2*x2)*vg3).
compute sew1 = sqrt(varw1).
compute t1 = w1/sew1.
do if (ovals <> 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-tcdf(abs(t1), dfres)).
compute cnms = {'Coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)= {w1,sew1,t1,p,llci,ulci}.
else if (ovals = 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-cdfnorm(abs(t1))).
compute cnms = {'Coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)= {w1,sew1,t1,p,llci,ulci}.

```

```

end if.
end loop.
compute rnms=t(xname).
compute mdvalpr=probeval(jj,1).
print mdvalpr/title = 'Moderator
value: '/rnames=rnn2/format= F10.4 /space=0.
print matt/title='
'/cnames=cnms/rnames=rnms/format= F10.4
/space=0.
compute xprob=x(:,(ncol(x)-(2*nx)))-mdvalpr.
loop kk = 1 to nx.
compute xprob={xprob,
(xprob(:,1)&*x(:,(ncol(x)-(2*nx)+kk)))}.
end loop.
compute xprob={x(:,1:((ncol(x)-(2*nx))-
1)),xprob}.
do if (ovals <> 2).
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute residc = y-(xprob*bmultc).
compute ssresidc=csum(residc&*residc).
compute r2c = r2-(1-(ssresidc/sstotal)).
compute fcha2=(dfres*r2c)/(nx*(1-r2)).
do if (hc3 = 1).
loop kk = 1 to nx.
compute xprob={xprob, x(:,(ncol(x)-
(2*nx)+kk))}.
end loop.
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute k3 = nrow(bmultc).
compute xhc=xprob.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)=
xhc(i3,:)*inv(t(xprob)*xprob)*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).

```

```

end loop.
compute
varbc=(inv(t(xprob)*xprob)*t(xhc)*xhc*inv(t(xprob)*xprob)).
compute lmat=make((nrow(bmultc)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha2 =
(t(t(lmat)*bmultc)*inv(t(lmat)*varbc*lmat)*((t(lmat)*bmultc)))/nx).
end if.
compute pvalr2cc=1-fcdf(fcha2,nx,dfres).
compute rcha2 = {r2c, fcha2, nx,
dfres,pvalr2cc}.
print rcha2/title = 'Test of equality of conditional
means at this value of the moderator'/clabels
'R2-chng' 'F' 'df1' 'df2' 'p'/format F10.4.
else if (ovals = 2).
compute btmp=b.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=xprob.
compute bt1 = make(ncol(xlp),1,0).

.
compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.

```

```

end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL4.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(lldiff,nx).
compute rcha2 = {LLdiff, nx,pchi}.
print rcha2/title = 'Test of equality of log odds
conditioned on this moderator value'/clabels
'Chi-sq' 'df' 'p'/format F10.4.
end if.
end if.
compute
tttt=make(nrow(dummat),1,probeval(jj,1)).
compute tttt={tttt,dummat(:,2:ncol(dummat))}.
loop kkk=1 to nx.
compute tttt={tttt,tttt(:,1)*tttt(:,1+kkk)}.
end loop.
compute ones=make(nrow(dummat),1,1).
do if (ncol(x) > (2*nx+2)).

```

```

compute
covmnm=make(nrow(tttt),ncol(covmns),0).
loop kkk=1 to nrow(tttt).
compute covmnm(kkk,:)=covmns.
end loop.
compute tttt={covmnm,tttt}.
else.
compute tttt={ones,tttt}.
end if.
compute focvals={focvals;dummat(:,1),tttt}.
do if (mcfoc > 0).
compute yhat={dummat(:,1),(tttt*b)}.
compute cmnms={fciv, 'yhat'}.
do if (ovals <> 2).
print yhat/title = 'Estimated conditional means
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (ovals = 2).
print yhat/title = 'Estimated conditional log odds
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (jj <> nrow(probeval)).
print/title='-----'/space=0.
end if.
end if.
end loop.
compute focvals=focvals(2:nrow(focvals),:).
compute yhat=focvals(:,2:ncol(focvals))*b.
compute
focvals={focvals(:,1),focvals(:,(ncol(focvals)-
(2*nx))),yhat}.
compute cnms={fciv,mdtr,'yhat'}.
do if (mcmod > 0).
compute cnms={mdtr,fciv,'yhat'}.
end if.
end if.
do if (mcmod > 0).
compute outp=make((nx+1),7,0).

```

```

compute bcatm={b((ncol(x)-
(2*nx)),1);b((ncol(x)-nx+1):ncol(x),1)}.
compute outp(:,1)=dummat(:,1).
compute bcatcov=varb((ncol(x)-
nx):ncol(x),(ncol(x)-nx):ncol(x)).
compute bcatcov(1,1)=varb((ncol(x)-
(2*nx)),(ncol(x)-(2*nx))).
compute
bcatcov(2:nrow(bcatcov),1)=varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx)).
compute
bcatcov(1,2:nrow(bcatcov))=t(varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx))).
loop i = 1 to nrow(dummat).
compute
catmval={1,dummat(i,2:ncol(dummat))}.
compute condeff=catmval*bcatm.
compute
condse=sqrt(catmval*bcatcov*t(catmval)).
compute outp(i,2:3)={condeff,condse}.
end loop.
compute outp(:,4)=outp(:,2)&/outp(:,3).
compute outp(:,5) = 2*(1-tcdf(abs(outp(:,4)),
dfres)).
compute outp(:,6) = (outp(:,2)-
sqrt(tval)&*outp(:,3)).
compute outp(:,7) =
(outp(:,2)+sqrt(tval)&*outp(:,3)).
do if (ovals <> 2).
compute cnmms = {xname2(1,1), 'coeff', 'se',
't', 'p', 'LLCI', 'ULCI'}.
end if.
do if (ovals = 2).
compute outp(:,5) = 2*(1-
cdfnorm(abs(outp(:,4)))).
compute cnmms = {xname2(1,1), 'coeff', 'se',
'Z', 'p', 'LLCI', 'ULCI'}.
end if.

```

```

print outp/title = 'Conditional Effect of Focal
Predictor in Groups Defined by the Moderator
Variable:.'/cnames=cnmms/format = F10.4.
end if.
do if (nval > 2 and (mmodval = 999) and
mcmmod = 0).
do if (quantile <> 1).
print/title = 'Moderator values are the sample
mean and plus/minus one SD from mean'.
else.
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
end if.
do if (highwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD above the mean was'.
print/title = 'replaced with the maximum
because one SD above the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
do if (lowwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD below the mean was'.
print/title = 'replaced with the minimum
because one SD below the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
end if.
do if (nval = 2 and (mmodval = 999)).
print/title = 'The moderator variable is
dichotomous'.
end if.
do if (mmodval <> 999 and mcmmod = 0 and
((mmodval < mdmin) or (mmodval > mdmax))).
print/title = 'Warning: Moderator value specified
is outside of the range of the data'.
end if.

```

```

do if (plot = 1).
do if (mcfoc = 0 and mcmmod=0).
compute cnms = {t(nms((ncol(x)-2):(ncol(x)-
1),1)), 'yhat'}.
end if.
do if (ovals = 2).
compute prob =
exp(focvals(:,3))/(1+exp(focvals(:,3))).
compute focvals = {focvals, prob}.
compute cnms = {cnms, 'prob'}.
end if.
print/title =
*****
*****!

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (ovals = 2).
do if (mcfoc=0).
print/title = 'DATA LIST
FREE/Technological_Advan Comm_Pract
logodds prob.' /space=1.
end if.
do if (mcmmod=0).

```



```

print/title = 'DATA LIST FREE/Comm_Pract
Technological_Advan logodds prob.'
/space=1.
end if.
else if (ovals <> 2).
do if (mcfoc=0).
print/title = 'DATA LIST
FREE/Technological_Advan Comm_Pract
Operational_Performance.' /space=1.
end if.
do if (mcmod=0).
print/title = 'DATA LIST FREE/Comm_Pract
Technological_Advan
Operational_Performance.' /space=1.
end if.
end if.
print/title = 'BEGIN DATA.' /space=0.
print focvals/title = ' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (ovals <> 2 and mcmod=0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH Operational_Performance BY
Comm_Pract.' /space=0.
end if.
do if (ovals <> 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
Operational_Performance BY
Technological_Advan.' /space=0.
end if.
do if (ovals = 2 and mcmod=0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Comm_Pract.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Comm_Pract.' /space=0.
end if.
do if (ovals = 2 and mcfoc=0).

```

```

print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
logodds BY Technological_Advan.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
prob BY Technological_Advan.' /space=0.
end if.
do if (ncovs > 0).
print/title = 'NOTE: For data above, covariates
are set to their sample means.'.
end if.
end if.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
end if.
END IF.
end if.
do if (bad > 0).
compute note(notes,1) = 9.
compute notes = notes + 1.
end if.
print/title = '***** ANALYSIS
NOTES AND WARNINGS
*****'.
loop i = 1 to errs.
do if (runerrs(i,1) = 1).
print/title = 'ERROR: One of your declared
mediators is dichotomous. This procedure
can"t be used.'.
end if.
do if (runerrs(i,1) = 2).

```

```
print/title = 'ERROR: For model 6, this
procedure limits the number of mediators to
four.'.
end if.
do if (runerrs(i,1) = 3).
print/title = 'ERROR: For models 1, 2, and 3,
only a single variable can be listed in the M
list.'.
end if.
do if (runerrs(i,1) = 4).
print/title = 'ERROR: You requested a model
involving W but did not provide a valid W
variable name.'.
end if.
do if (runerrs(i,1) = 5).
print/title = 'ERROR: You requested a model
involving Z but did not provide a valid Z
variable name.'.
end if.
do if (runerrs(i,1) = 6).
print/title = 'ERROR: You requested a model
involving Q but did not provide a valid Q
variable name.'.
end if.
do if (runerrs(i,1) = 7).
print/title = 'ERROR: You requested a model
involving V but did not provide a valid V
variable name.'.
end if.
do if (runerrs(i,1) = 8).
print/title = 'ERROR: You specified a W
variable for a model that does not need it'.
end if.
do if (runerrs(i,1) = 9).
print/title = 'ERROR: You specified a Z variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 10).
print/title = 'ERROR: You specified a Q variable
for a model that does not need it'.
```

```

end if.
do if (runerrs(i,1) = 11).
print/title = 'ERROR: You specified a V variable
for a model that does not need it.'.
end if.
do if (runerrs(i,1) = 12).
print/title = 'ERROR: The variable specified for
W has already been assigned.'.
end if.
do if (runerrs(i,1) = 13).
print/title = 'ERROR: The variable specified for
Z has already been assigned.'.
end if.
do if (runerrs(i,1) = 14).
print/title = 'ERROR: The variable specified for
Q has already been assigned.'.
end if.
do if (runerrs(i,1) = 15).
print/title = 'ERROR: The variable specified for
V has already been assigned.'.
end if.
do if (runerrs(i,1) = 16).
print/title = 'ERROR: You did not provide a
valid Y variable name.'.
end if.
do if (runerrs(i,1) = 17).
print/title = 'ERROR: The variable specified for
Y has already been assigned.'.
end if.
do if (runerrs(i,1) = 18).
print/title = 'ERROR: Model 6 requires more
than one mediator.'.
end if.
do if (runerrs(i,1) = 19).
print/title = 'ERROR: You have not specified a
valid model number.'.
end if.
do if (runerrs(i,1) = 20).
print/title = 'ERROR: At least one and only one
variable must be listed for X.'.

```

```
end if.  
do if (runerrs(i,1) = 21).  
print/title = 'ERROR: At least one and only one  
variable must be listed for Y.'.  
end if.  
do if (runerrs(i,1) = 22).  
print/title = 'ERROR: Iteration didn"t converge  
to a solution. Interpret results with caution.'.  
end if.  
do if (runerrs(i,1) = 23).  
print/title = 'ERROR: You specified a clustering  
variable that does not exist in your variable  
list.'.  
end if.  
do if (runerrs(i,1) = 24).  
print/title = 'ERROR: You specified a clustering  
variable that has already been assigned.'.  
end if.  
do if (runerrs(i,1) = 25).  
print/title = 'ERROR: One or more of your M  
variables is not listed in the variables list.'.  
end if.  
do if (runerrs(i,1) = 26).  
print/title = 'ERROR: A maximum of 20 cluster  
units is allowed. Use multilevel modeling  
instead.'.  
end if.  
do if (runerrs(i,1) = 27).  
print/title = 'ERROR: One of the variables in  
your model is a constant.'.  
end if.  
do if (runerrs(i,1) = 28).  
print/title = 'ERROR: Dichotomous Y is not  
permitted with WS option.'.  
end if.  
do if (runerrs(i,1) = 29).  
print/title = 'ERROR: Insufficient number of  
variables in vars= list when using WS option.'.  
end if.  
do if (runerrs(i,1) = 30).
```

```

print/title = 'ERROR: Too many variables in
vars= list when using WS option. Covariates
not allowed.'.
end if.
do if (runerrs(i,1) = 31).
print/title = 'ERROR: mmodval and wmodval
can"t both be set to zero with contrast option.'.
end if.
do if (runerrs(i,1) = 32).
print/title = 'ERROR: You did not provide a
valid X variable name.'.
end if.
do if (runerrs(i,1) = 33).
print/title = 'ERROR: PROCESS requires all
variable names to be eight characters or
fewer.'.
print/title = '      Please shorten variable names
and reexecute.'/space=0.
end if.
do if (runerrs(i,1)=34).
print/title = 'ERROR: X and M cannot both be
specified as multicategorical.'.
end if.
do if (runerrs(i,1)=35).
print/title = 'ERROR: Multicategorical variable
specification available only for models 1 and
4.'.
end if.
do if (runerrs(i,1)=36).
print/title = 'ERROR: Categorical variables
cannot have more than 10 categories.'.
end if.
do if (runerrs(i,1)=37).
print/title = 'ERROR: Each group must have at
least two cases.'.
end if.
do if (runerrs(i,1)=38).
print/title = 'ERROR: Cluster option not
available with multicategorical variables.'.
end if.

```

```

do if (runerrs(i,1)=39).
print/title = 'ERROR: A maximum of 10
variables is allowed in the M= list.'.
end if.
do if (runerrs(i,1) = 41).
print/title = 'ERROR: MCM option not permitted
in model 4.'.
end if.
end loop.
do if (errs = 0).
do if (boot > 1 or mc > 0).
do if (bconoff = 1 and boot > 1).
print boot/title = 'Number of bootstrap samples
for bias corrected bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (bconoff = 0 and boot > 1).
print boot/title = 'Number of bootstrap samples
for percentile bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (mc > 1).
print mc/title = 'Number of samples for Monte
Carlo confidence intervals:'/format = F8.0.
end if.
do if (booterr = 1).
compute badend = badend(1,2:ncol(badend)).
print badend/title = 'WARNING: Bootstrap CI
endpoints below not trustworthy. Decrease
confidence or increase bootstraps'/format =
F10.4.
end if.
end if.
print conf/title = 'Level of confidence for all
confidence intervals in output:'/format = F8.2.
do if ((center = 1 or ws=1) and (ncol(centvar) >
1)).
compute centvar = centvar(1,2:ncol(centvar)).

```

```

print centvar/title = 'NOTE: The following
variables were mean centered prior to
analysis:.'/format = a8.
end if.
loop i = 1 to notes.
do if (note(i,1) = 1).
print/title = 'NOTE: Confidence level restricted
to between 50 and 99.9999%. 95% confidence
is provided in output'.
end if.
do if (note(i,1) = 2).
print/title = 'NOTE: Effect size measures for
indirect effects are not available for models
with dichotomous outcomes'.
end if.
do if (note(i,1) = 3).
print/title = 'NOTE: All standard errors for
continuous outcome models are based on the
HC3 estimator'.
end if.
do if (note(i,1) = 6).
print/title = 'NOTE: The number of bootstrap
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 7).
print/title = 'NOTE: The Johnson-Neyman
method is available only for Models 1 and 3'.
end if.
do if (note(i,1) = 8).
print/title = 'NOTE: The Johnson-Neyman
method cannot be used with a dichotomous
moderator'.
end if.
do if (note(i,1)=9).
print bad/title = 'NOTE: Some bootstrap
samples had to be replaced. The number of
such replacements was:'.
end if.
do if (note(i,1) = 11).

```



```

print nmiss/title = 'NOTE: Some cases were
deleted due to missing data. The number of
such cases was:'.
end if.
do if (note(i,1) = 12).
print /title = 'NOTE: Monte Carlo method
available only for models 4 and 5.
Bootstrapping was used instead.'.
end if.
do if (note(i,1) = 13).
print/title = 'NOTE: The number of Monte Carlo
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 19).
print/title = 'NOTE: Effect sizes not available for
within-subject analyses.'.
end if.
do if (note(i,1) = 16).
print/title = 'NOTE: Normal theory tests not
available for within-subject analyses.'.
end if.
do if (note(i,1) = 17).
print/title = 'NOTE: Monte Carlo confidence
intervals not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 18 and warnrep=0).
print/title = 'WARNING: You have requested
OLS estimation with a dichotomous criterion.'.
print/title = 'Interpret model coefficients and
inferential statistics with caution.'/space=0.
compute warnrep=1.
end if.
do if (note(i,1) = 20).
print/title = 'NOTE: Saving of bootstrap
estimates not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 22).

```

```
print/title = 'NOTE: Effect size option with
covariates requires covariates in models of M
and Y.'.
end if.
do if (note(i,1) = 23).
print/title = 'NOTE: Johnson-Neyman method
not available with multicategorical X'.
end if.
do if (note(i,1) = 24).
print/title = 'NOTE: MMODVAL option not
available with a multicategorical moderator'.
end if.
do if (note(i,1) = 25).
print/title = 'NOTE: Omnibus test for relative
indirect effects is not available with the MC
option.'.
end if.
do if (note(i,1) = 26).
print/title = 'NOTE: CONTRAST option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 27).
print/title = 'NOTE: EFFSIZE option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 28).
print/title = 'NOTE: NORMAL option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 31).
print/title = 'NOTE: The TOTAL option is not
available when Y is dichotomous.'.
end if.
do if (note(i,1) = 32).
print/title = 'NOTE: Kappa-squared is disabled
from output as of version 2.16.'.
end if.
do if (note(i,1) = 33).
print/title = 'NOTE: COVMY option ignored in
models 1, 2, and 3.'.
```

| | | | |
|-----------|----------------|-------------|-------------|
| | | end if. | |
| | | end loop. | |
| | | end if. | |
| | | end matrix. | |
| Resources | Processor Time | | 00:00:00.99 |
| | Elapsed Time | | 00:00:00.99 |

Run MATRIX procedure:

***** PROCESS Procedure for SPSS Release 2.16.3 *****

Written by Andrew F. Hayes, Ph.D. www.afhayes.com
Documentation available in Hayes (2013). www.guilford.com/p/hayes3

Model = 1
Y = Operatio
X = Comm_Pra
M = Technolo

Sample size
856

Outcome: Operatio

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|----------|--------|----------|-------|
| .8758 | .7670 | .2799 | 934.9425 | 3.0000 | 852.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|---------|-------|----------|-------|---------|--------|
| constant | -1.2333 | .1415 | -8.7183 | .0000 | -1.5110 | -.9557 |
| Technolo | .8329 | .0604 | 13.7794 | .0000 | .7142 | .9515 |
| Comm_Pra | 1.2019 | .0604 | 19.8851 | .0000 | 1.0833 | 1.3205 |
| int_1 | -.1774 | .0134 | -13.2004 | .0000 | -.2038 | -.1511 |

Product terms key:

int_1 Comm_Pra X Technolo

R-square increase due to interaction(s):

| | R2-chng | F | df1 | df2 | p |
|-------|---------|----------|--------|----------|-------|
| int_1 | .0477 | 174.2497 | 1.0000 | 852.0000 | .0000 |

Conditional effect of X on Y at values of the moderator(s):

| Technolo | Effect | se | t | p | LLCI | ULCI |
|----------|--------|-------|---------|-------|-------|-------|
| 2.7372 | .7162 | .0423 | 16.9361 | .0000 | .6332 | .7992 |
| 3.8107 | .5257 | .0424 | 12.4046 | .0000 | .4425 | .6089 |
| 4.8843 | .3352 | .0471 | 7.1141 | .0000 | .2427 | .4277 |

Values for quantitative moderators are the mean and plus/minus one SD from mean.

Values for dichotomous moderators are the two values of the moderator.

***** JOHNSON-NEYMAN TECHNIQUE *****

There are no statistical significance transition points within the observed range of the moderator.

Data for visualizing conditional effect of X on Y

Paste text below into a SPSS syntax window and execute to produce plot.

DATA LIST FREE/Comm_Pract Technological_Advan Operational_Performance.
BEGIN DATA.

| | | |
|--------|--------|--------|
| 2.7372 | 2.7372 | 3.0068 |
| 3.8107 | 2.7372 | 3.7757 |
| 4.8843 | 2.7372 | 4.5445 |
| 2.7372 | 3.8107 | 3.3795 |
| 3.8107 | 3.8107 | 3.9439 |
| 4.8843 | 3.8107 | 4.5082 |
| 2.7372 | 4.8843 | 3.7522 |
| 3.8107 | 4.8843 | 4.1120 |
| 4.8843 | 4.8843 | 4.4719 |

END DATA.

GRAPH/SCATTERPLOT=Comm_Pract WITH Operational_Performance BY
Technological_Advan.

***** ANALYSIS NOTES AND WARNINGS *****

Level of confidence for all confidence intervals in output:
95.00

----- END MATRIX -----

restore.

Matrix

Notes

| | | | |
|----------------|--------------------------------|----------------------|--|
| Output Created | | 20-OCT-2020 16:05:49 | |
| Comments | | | |
| Input | Active Dataset | DataSet1 | |
| | Filter | <none> | |
| | Weight | <none> | |
| | Split File | <none> | |
| | N of Rows in Working Data File | 856 | |

Syntax

```
matrix.  
get dat/file = */variables =  
Financial_Performance Pro_Env_Strgy  
Technological_Advan /names =  
vnames/missing = 9999.  
compute ninit = nrow(dat).  
get dat/file = */variables =  
Financial_Performance Pro_Env_Strgy  
Technological_Advan /names =  
vnames/missing = omit.  
get tmp/file = */variables =  
Financial_Performance /names =  
ynames/missing = omit.  
get tmp2/file = */variables = Pro_Env_Strgy  
/names = xnames/missing = omit.  
get tmp/file = */variables =  
Technological_Advan /names =  
mnames/missing = omit.  
get tmp/file = */variables = w999999t z999999t  
v999999t q999999t.  
compute wname=tmp(1,1).  
do if (wname = ' ').  
compute wname = 'xxx'.  
end if.  
compute zname=tmp(1,2).  
do if (zname = ' ').  
compute zname = 'xxx'.  
end if.  
compute vname=tmp(1,3).  
do if (vname = ' ').  
compute vname = 'xxx'.  
end if.  
compute qname=tmp(1,4).  
do if (qname = ' ').  
compute qname = 'xxx'.  
end if.  
compute n = nrow(dat).  
compute p0=-.322232431088.  
compute p1 = -1.  
compute p2 = -.342242088547.
```

```

compute p3 = -.0204231210245.
compute p4 = -.0000453642210148.
compute q0 = .0993484626060.
compute q1 = .588581570495.
compute q2 = .531103462366.
compute q3 = .103537752850.
compute q4 = .0038560700634.
compute badend = 0.
compute priorlo = -9999999.
compute priorhi = 9999999.
compute criterr = 0.
compute cluster = 0.
compute clsdmy = 0.
compute jndich = 0.
compute wvdich=0.
compute zqdich=0.
compute mod74dic=0.
compute warnrep=0.
compute contrast = trunc( 0 ).
do if (contrast > 2 or contrast < 0).
compute contrast = 0.
end if.
compute booterr = 0.
compute effsize = ( 0 = 1).
compute note = make(10,1,0).
compute notes = 1.
compute iterr = 0.
compute clsmth = 0.
compute quantile = ( 0 = 1).
compute mmodval= 999.
compute jn = ( 1 = 1).
compute covcoeff=( 0 =1).
compute center = ( 0 = 1).
compute detail = ( 1 = 1).
compute coeffci = ( 1 = 1).
compute conf = 95.
compute longname = ( 1 =1).
compute bconoff=( 0 <> 1).
compute covmy = trunc( 0 ).
do if (covmy < 0 or covmy > 2).

```

```

compute covmy = 0.
end if.
do if (trunc(conf) >= 100 or (trunc(conf) <=
50)).
compute conf = 95.
compute note(notes,1) = 1.
compute notes = notes + 1.
end if.
do if (n < ninit).
compute nmiss = ninit-n.
compute note(notes,1) = 11.
compute notes = notes + 1.
end if.
compute errs = 0.
compute quantd = {0,0,0,0,0,0}.
compute quantc = {0,0,0,0,0,0}.
compute mcheck = 0.
compute ttt = 0.
compute plot = trunc( 1 ).
compute runerrs = make(50,1,0).
compute model = trunc( 1 ).
compute normal = 0.
compute ws= ( 0 =1).
compute olsdichm= ( 0 =1).
compute olsdichy= ( 0 =1).
compute mcx=trunc( 0 ).
compute mcm=trunc( 0 ).
do if (ws = 1).
do if (effsize = 1).
compute note(notes,1) = 19.
compute notes = notes + 1.
compute effsize=0.
end if.
do if (normal=1).
compute note(notes,1) = 16.
compute notes = notes + 1.
compute normal=0.
end if.
end if.
do if (model <> 4).

```



```

compute ws = 0.
end if.
do if (model > 0 and model < 4 and covmy <>
0).
compute covmy=0.
compute note(notes,1) = 33.
compute notes = notes + 1.
end if.
print/space=0.
do if (jn = 1 and model <> 1 and model <> 3).
compute note(notes,1) = 7.
compute notes = notes + 1.
end if.
do if ((model > 76) or (model < 1)).
compute model = 77.
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 19.
end if.
compute itprob=0.
do if (mcx < 0).
compute mcx=0.
end if.
do if (mcx > 4).
compute mcx=0.
end if.
do if (mcm < 0).
compute mcm=0.
end if.
do if (mcm > 4).
compute mcm=0.
end if.
do if (mcx > 0 and mcm > 0).
compute errs=errs+1.
compute runerrs(errs,1)=34.
compute criterr=1.
end if.
do if ((model <> 1 and model <> 4) and (mcx >
0 or mcm > 0)).
compute errs=errs+1.

```

```

compute runerrs(errs,1)=35.
compute criterr=1.
end if.
do if ((model = 4) and (mcm > 0)).
compute errs=errs+1.
compute runerrs(errs,1)=41.
compute criterr=1.
end if.
do if ((mcx > 0) and jn <> 0).
compute jn=0.
compute note(notes,1) = 23.
compute notes = notes + 1.
end if.
do if (mmodval <> 999 and mcm > 0).
compute note(notes,1) = 24.
compute notes = notes + 1.
end if.
compute toteff = 0.
compute toteff = (( 0 = 1)*( 1 = 4 or 1 = 6)).
compute varorder = 2.
compute hc3 = ( 0 <> 0).
compute cname = 'xxx'.
compute centvar = {'xxx'}.
compute nmeds = ncol(mnames).
do if (longname=0).

```

```
compute criterr = 1.  
compute errs = errs+1.  
compute runerrs(errs,1) = 33.
```

1

2

3

end if.

compute modelm =

```
{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,3;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,6;  
1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,7;  
1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8;  
1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,9;  
1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,10;  
1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,11;  
1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,0,12;  
1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,13;  
0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,14;
```

0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,2,15;
0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,4,16;
0,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,4,17;
0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,6,18;
0,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,6,19;
0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,6,20;
1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,7,21;
1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,7,22;
1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,23;
1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,24; 1,1,1,1

,0,0,0,0,0,0,0,0,0,0,0,0,0,0,25;
1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,0,26;
1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,0,27;
1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,7,28;
1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,7,29;
1,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,30;
1,1,0,1,0,0,1,1,0,1,0,0,0,0,0,0,31;
1,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,32;
1,1,1,1,0,0,1,1,1,1,0,0,0,0,0,0,33;
1,1,1,1,0,0,1,0,0,1,0,0,0,0,0,0,34;
1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,35;
1,0,0,1,1,0,0,0,0,0,1,1,0,0,0,0,0,36;
1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,37;
1,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,38;
1,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,39;
1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,40;
1,0,0,1,1,0,1,0,0,1,1,0,0,0,0,0,41;
1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,42;
1,0,0,1,1,1,1,0,0,1,1,1,0,0,0,0,0,43;
1,0,0,1,1,1,1,0,0,1,0,0,0,0,0,0,44;
1,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,45;
1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,46;
1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,47;
1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,48;
1,1,0,1,1,0,1,1,0,0,0,

0,0,0,0,0,0,0,0,49;
1,1,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,50;
1,1,1,1,1,0,1,1,1,0,0,0,0,0,0,0,0,51;

```

1,1,1,1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,52;
1,1,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,53;
1,1,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,54;
1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,55;
1,1,1,1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,56;
1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,57;
1,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,58;
1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,59;
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,60;
1,1,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,61;
1,1,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,62;
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,0,0,0,63;
1,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,64;
1,0,0,1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,65;
1,0,0,1,0,0,0,0,0,1,0,0,1,0,0,0,0,0,66;
1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,67;
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,68;
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,69;
1,0,0,1,0,0,0,0,0,0,0,0,1,1,0,0,0,0,70;
1,0,0,1,0,0,1,0,0,1,0,0,1,1,1,0,0,0,71;
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,1,1,0,72;
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,1,1,0,0
,73; 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,74;
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,75;
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,1,0,0,0,76;
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,77}.
compute wm = modelm(model, 1).
compute zm = modelm(model, 2).
compute wzm = modelm(model, 3).
compute vy = modelm(model, 4).
compute qy = modelm(model, 5).
compute vqy = modelm(model, 6).
compute wy = modelm(model, 7).
compute zy = modelm(model, 8).
compute wzy = modelm(model, 9).
compute vxy = modelm(model, 10).
compute qxy = modelm(model, 11).
compute vqxy = modelm(model, 12).
compute wmy = modelm(model, 13).

```

```

compute wvmy = modelm(model, 14).
compute wvxy = modelm(model, 15).
compute zmy = modelm(model,16).
compute wzmy = modelm(model,17).
compute xmy = modelm(model,18).
compute imm=modelm(model,19).
do if (model = 74).
compute vdich=0.
end if.
do if (ncol(xname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 20.
compute criterr = 1.
end if.
do if (ncol(mnames) > 10).
compute errs = errs+1.
compute runerrs(errs,1) = 39.
compute criterr = 1.
end if.
do if (ncol(yname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 21.
compute criterr = 1.
end if.
do if (( 999 =0) and ( 999 =0) and (model=3 or
model=2) and (contrast=1)).
compute errs = errs+1.
compute runerrs(errs,1) = 31.
compute criterr = 1.
end if.
compute xlist = (wm or zm or wzm or wy or zy
or wzy or vxy or qxy or vqxy or wvxy or xmy).
compute mlist = (vy or qy or vqy or zmy or wmy
or wzmy or xmy or (model < 4)).
compute bad=0.
do if (criterr = 0).
compute werr = 0.
compute verr = 0.
compute qerr = 0.
compute zerr = 0.

```



```

compute yerr = 1.
compute xerr = 1.
compute wlist = (wm or wzm or wy or wzy or
wm or wvmy or wvmy or wvxy or wzmy).
do if (wlist = 1 and wname = 'xxx').
compute werr = 1.
compute wlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (wlist = 1 and ((wname = qname) or
(wname = vname) or (wname = zname) or
(wname = xname) or (wname = yname))).
compute werr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 12.
end if.
compute zlist = (zm or wzm or zy or wzy or
zmy or wzmy).
do if (zlist = 1 and zname = 'xxx').
compute zerr = 1.
compute zlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (zlist = 1 and ((zname = qname) or
(zname = vname) or (zname = wname) or
(zname = xname) or (zname = yname))).
compute zerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 13.
end if.
compute qlist = (qy or vqy or qxy or vqxy).
do if (qlist = 1 and qname = 'xxx').
compute qerr = 1.
compute qlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.

```

```

do if (qlist = 1 and ((qname = zname) or
(qname = vname) or (qname = wname) or
(qname = xname) or (qname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 14.
end if.
compute vlist = (vy or vqy or vxy or vqxy or
wvmy or wvxy).
do if (vlist = 1 and vname = 'xxx').
compute verr = 1.
compute vlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (vlist = 1 and ((vname = zname) or
(vname = qname) or (vname = wname) or
(vname = xname) or (vname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 15.
end if.
do if (wlist = 0 and wname <> 'xxx').
compute werr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 8.
end if.
do if (zlist = 0 and zname <> 'xxx').
compute zerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 9.
end if.
do if (qlist = 0 and qname <> 'xxx').
compute qerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 10.
end if.
do if (vlist = 0 and vname <> 'xxx').
compute verr = 2.
compute errs = errs+1.

```

```

compute runerrs(errs,1) = 11.
end if.
do if (hc3 = 1).
compute note(notes,1) = 3.
compute notes = notes+1.
end if.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+(((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute cons = make(n,1,1).
compute temp = (n*sscp(dat))-
(t(csum(dat))*(csum(dat))).
compute temp = temp/(n*(n-1)).
compute temp = csum(((diag(temp))=0)).
compute temp2=1.
do if (ws = 1).
compute temp2 = (n*sscp(tmp2))-
(t(csum(tmp2))*(csum(tmp2))).
compute temp2 = temp2/(n*(n-1)).
end if.
do if ((temp > 0 and ws = 0) or (temp > 0 and
ws = 1 and temp2 > 0)).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 27.
end if.
compute nmeds = ncol(mnames).
compute mcmats=ident(nmeds*2).
compute mccoef=make(nmeds*2,1,0).
compute sobel = make(nmeds,4,-999).
do if (model = 6 and nmeds > 4).
compute errs = errs+1.
compute runerrs(errs,1)=2.
end if.
do if (model < 4 and nmeds > 1).
compute errs = errs+1.
compute runerrs(errs,1) = 3.
end if.

```

```

compute nmods = (model = 74).
compute bad = 0.
compute intcnt = 1.
compute modvals = 0.
compute modvalsd = 0.
compute yintemp = {'int_1', 'int_2', 'int_3',
'int_4', 'int_5', 'int_6', 'int_7', 'int_8', 'int_9',
'int_10', 'int_11', 'int_12', 'int_13', 'int_14',
'int_15'}.
compute yintemp = {yintemp, 'int_16', 'int_17',
'int_18', 'int_19', 'int_20', 'int_21', 'int_22',
'int_23', 'int_24', 'int_25', 'int_26', 'int_27',
'int_28'}.
compute yintemp = {yintemp, 'int_29', 'int_30',
'int_31', 'int_32', 'int_33', 'int_34', 'int_35',
'int_36', 'int_37', 'int_38', 'int_39', 'int_40'}.
compute yintemp = {yintemp, 'int_41', 'int_42',
'int_43', 'int_44', 'int_45', 'int_46', 'int_47',
'int_48', 'int_49', 'int_50', 'int_51', 'int_52'}.
compute yintemp = {yintemp, 'int_53', 'int_54',
'int_55', 'int_56', 'int_57', 'int_58', 'int_59',
'int_60', 'int_61', 'int_62', 'int_63', 'int_64'}.
compute
cntname={'(C1)';'(C2)';'(C3)';'(C4)';'(C5)';'(C6)';'(
C7)';'(C8)';'(C9)';'(C10)';'(C11)';'(C12)';'(C13)';(
C14)';'(C15)';'(C16)';'(C17)'}.
compute
cntname={cntname;'(C18)';'(C19)';'(C20)';'(C21
)';'(C22)';'(C23)';'(C24)';'(C25)';'(C26)';'(C27)';(
C28)';'(C29)';'(C30)';'(C31)'}.
compute
cntname={cntname;'(C32)';'(C33)';'(C34)';'(C35
)';'(C36)';'(C37)';'(C38)';'(C39)';'(C40)';'(C41)';(
C42)';'(C43)';'(C44)';'(C45)'}.
compute
cntname={cntname;'(C46)';'(C47)';'(C48)';'(C49
)';'(C50)';'(C51)';'(C52)';'(C53)';'(C54)';'(C55)';(
C56)';'(C57)';'(C58)';'(C59)'}.
compute
cntname={cntname;'(C60)';'(C61)';'(C62)';'(C63

```

```

);'(C64)';'(C65)';'(C66)';'(C67)';'(C68)';'(C69)';'(
C70)';'(C71)';'(C72)';'(C73)}.

compute
cntname={cntname;'(C74)';'(C75)';'(C76)';'(C77
)';'(C78)';'(C79)';'(C80)';'(C81)';'(C82)';'(C83)';'(
C84)';'(C85)';'(C86)';'(C87)}.

compute
cntname={cntname;'(C88)';'(C89)';'(C90)';'(C91
)';'(C92)';'(C93)';'(C94)';'(C95)';'(C96)';'(C97)';'(
C98)';'(C99)';'(C100)';'(C101)}.

compute
cntname={cntname;'(C102)';'(C103)';'(C104)';'(
C105)}.

compute apathnam={'a path'; 'a1 path'; 'a2
path'; 'a3 path'; 'a4 path'; 'a5 path'; 'a6 path';
'a7 path'; 'a8 path'; 'a9 path'; 'a10 path'}.
compute bpathnam={'b path'; 'b1 path'; 'b2
path'; 'b3 path'; 'b4path'; 'b5path'; 'b6 path'; 'b7
path'; 'b8 path'; 'b9 path'; 'b10 path'}.
compute modvnm = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute modvnm2 = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute mlab = { ' M1 ='; ' M2 ='; ' M3 ='; '
M4 ='; ' M5 ='; ' M6 ='; ' M7 ='; ' M8 ='; '
M9 ='; ' M10 ='}.
compute m = make(n,nmeds,1).
compute mmat = make(16,nmeds,0).
compute ymat = make(8,nmeds,0).
compute deco = make(10,1,0).
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
compute iterate = abs(trunc( 10000 )).
compute converge = abs( 0.00000001 ).
compute boot = abs(trunc( 5000 )).
do if (ws=1 and 0 > 0).
compute note(notes,1)=17.
compute notes=notes+1.

```

```

end if.
compute mc=(abs(trunc( 0 ))*(1-ws)).
do if (mc > 0 and model > 5).
do if (boot = 0).
compute boot=mc.
end if.
compute mc=0.
compute note(notes,1)=12.
compute notes = notes+1.
end if.
compute adjust = 0.
do if (mc > 0 and (model > 3 and model < 6)).
compute boot=0.
compute bconoff=0.
end if.
do if (boot > 0 and mc > 0).
compute mc = 0.
end if.
compute savboot = 0.
compute saveboot = ( 0 = 1).
do if (ws=1 and saveboot=1).
compute saveboot=0.
compute note(notes,1) = 20.
compute notes = notes + 1.
compute normal=0.
end if.
do if (saveboot =1 and boot > 0 and model >
3).
compute savboot = 1.
end if.
do if (boot <> 0 or mc <> 0).
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
loop.
compute cilow = rnd(bootsz*(1-(conf/100))/2).
compute cihigh =
trunc((bootsz*(conf/100)+(bootsz*(1-
(conf/100))/2)))+1.

```

```

do if (cilow < 1 or cihigh > bootsz).
compute
bootsz=trunc((bootsz+1000)/1000)*1000.
compute adjust = 1.
end if.
end loop if (cilow gt 0 and cihigh le bootsz).
compute boot=bootsz.
do if (mc > 0).
compute mc = bootsz.
end if.
do if (boot > 0 and mc > 0 and (model > 3 and
model < 6)).
compute boot=0.
end if.
do if (adjust = 1 and boot > 0).
compute note(notes,1)=6.
compute notes = notes+1.
end if.
do if (adjust = 1 and mc > 0).
compute note(notes,1)=13.
compute notes = notes+1.
end if.
end if.
do if (model = 6 and nmeds > 1).
compute mmpaths =
make((nmeds+2),(nmeds+2),0).
do if (nmeds = 2).
compute indboot = make(boot+1, 3, 999).
else if (nmeds = 3).
compute indboot = make(boot+1, 7, 999).
else if (nmeds = 4).
compute indboot = make(boot+1, 15, 999).
end if.
compute indlbl = {'Total:','Ind1 :','Ind2 :','Ind3
:','Ind4 :','Ind5 :','Ind6 :','Ind7 :','Ind8 :','Ind9
:','Ind10:','Ind11:','Ind12:','Ind13:','Ind14:','
Ind15:'}.
compute indlbl2 = {'Ind1','Ind2','Ind3','Ind4',
'Ind5','Ind6','Ind7','Ind8','Ind9','Ind10',
'Ind11','Ind12','Ind13','Ind14','Ind15'}.

```

```

compute indces = make(boot+1, 4, 999).
end if.
do if (model < 4).
compute boot = 0.
compute cmat = make(10,1,0).
compute zmat = make(10,1,0).
end if.
compute nvarch = make(1,ncol(dat),0).
compute wmatch = 0.
compute zmatch = 0.
compute vmatch = 0.
compute qmatch = 0.
compute mmatch=0.
compute minprobe=0.
compute maxprobe=0.
loop i = 1 to ncol(vnames).
do if (vnames(:,i)=yname).
compute y = dat(:,i).
compute nvarch(1,i)=1.
compute yerr = 0.
do if ((yname = xname) or (yname = wname) or
(yname = zname) or (yname = vname) or
(yname = qname)).
compute errs = errs+1.
compute runerrs(errs,1)=17.
end if.
end if.
do if (vnames(:,i)=xname).
compute x = dat(:,i).
compute nvarch(1,i)=1.
compute xdich = 1.
compute xerr=0.
loop jj = 1 to n.
do if ((x(jj,1) <> cmin(x)) and (x(jj,1) <>
cmax(x))).
compute xdich = 0.
break.
end if.
end loop.
compute xmean = csum(x)/n.

```



```

do if (center = 1 and (model < 4 or xlist > 0)
and mcx = 0).
compute meanvec = make(n,1, xmean).
compute x = x-meanvec.
compute centvar = {centvar, xname}.
end if.
compute xmean = csum(x)/n.
compute tmp = x-(cons*xmean).
compute xsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
do if (xdich = 0).
compute quantc(1,6) = 1.
compute matx = {xmean-xsd; xmean;
xmean+xsd}.
do if ( 999 = 999 and quantile = 0).
do if ((matx(1,1) < cmin(x)) and model = 74).
compute matx(1,1)=cmin(x).
compute minprobe=1.
end if.
do if ((matx(3,1) > cmax(x)) and model = 74).
compute matx(3,1)=cmax(x).
compute maxprobe=1.
end if.
end if.
do if (quantile = 1).
compute quantd(1,6) = 1.
compute quantc(1,6) = 0.
compute tmp = x.
compute tmp(GRADE(x)) = x.
compute matx =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (xdich = 1).
compute matx = {cmin(x); cmax(x)}.
do if (model=74).
compute matx=cmin(x).
compute mod74dic=1.
end if.

```

```

end if.
do if ( 999 <> 999).
compute matx = 999.
compute quantd(1,6) = 0.
compute quantc(1,6) = 0.
end if.
end if.
do if (werr = 0 and wlist = 1).
do if (vnames(:,i)=wname).
compute werr = 0.
compute wmatch = 1.
compute w = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(w)/n).
compute w = w-meanvec.
compute centvar = {centvar, wname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute wmean = csum(w)/n.
compute tmp = w-(cons*wmean).
compute wsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute wdich = 1.
loop jj = 1 to n.
do if ((w(jj,1) <> cmin(w)) and (w(jj,1) <>
cmax(w))).
compute wdich = 0.
break.
end if.
end loop.
do if (model = 3).
compute jndich=wdich.
compute jnmin=cmin(w).
compute jnmax=cmax(w).
end if.
do if (wdich = 0).
compute matw = {wmean-wsd; wmean;
wmean+wsd}.
do if ( 999 = 999 and quantile = 0).
do if (matw(1,1) < cmin(w)).

```

```

compute matw(1,1)=cmin(w).
compute minprobe=1.
end if.
do if (matw(3,1) > cmax(w)).
compute matw(3,1)=cmax(w).
compute maxprobe=1.
end if.
end if.
compute quantc(1,1) = 1.
do if (quantile = 1).
compute quantd(1,1) = 1.
compute quantc(1,1) = 0.
compute tmp = w.
compute tmp(GRADE(w)) = w.
compute matw =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (wdich = 1).
compute matw = {cmin(w); cmax(w)}.
compute wvdich=1.
compute cmaxw=cmax(w).
compute cminw=cmin(w).
end if.
do if ( 999 <> 999).
compute matw = 999.
compute quantd(1,1) = 0.
compute quantc(1,1) = 0.
end if.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm(1,1)=wname.
compute modmatp(1,1) = 1.
end if.
end if.
do if (zerr = 0 and zlist = 1).
do if (vnames(:,i)=zname).
compute zerr = 0.

```

```

compute zmatch = 1.
compute z = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(z)/n).
compute z = z-meanvec.
compute centvar = {centvar, zname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute zmean = csum(z)/n.
compute tmp = z-(cons*zmean).
compute zsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute zdich = 1.
loop jj = 1 to n.
do if ((z(jj,1) <> cmin(z)) and (z(jj,1) <>
cmax(z))).
compute zdich = 0.
break.
end if.
end loop.
do if (zdich = 0).
compute matz = {zmean-zsd; zmean;
zmean+zsd}.
do if ( 999 = 999 and quantile = 0).
do if (matz(1,1) < cmin(z)).
compute matz(1,1)=cmin(z).
compute minprobe=1.
end if.
do if (matz(3,1) > cmax(z)).
compute matz(3,1)=cmax(z).
compute maxprobe=1.
end if.
end if.
compute quantc(1,2) = 1.
do if (quantile = 1).
compute quantd(1,2) = 1.
compute quantc(1,2) = 0.
compute tmp = z.
compute tmp(GRADE(z)) = z.

```

```

compute matz = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (zdich = 1).
compute matz = {cmin(z); cmax(z)}.
compute cmaxz=cmax(z).
compute cminz=cmin(z).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matz = 999.
compute quantd(1,2) = 0.
compute quantc(1,2) = 0.
end if.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm(1,2)=zname.
compute modmatp(1,2) = 1.
end if.
end if.
do if (verr = 0 and vlist = 1).
do if (vnames(:,i)=vname).
compute verr = 0.
compute vmatch = 1.
compute v = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(v)/n).
compute v = v-meanvec.
compute centvar = {centvar, vname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute vmean = csum(v)/n.
compute tmp = v-(cons*vmean).
compute vsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute vdich = 1.
loop jj = 1 to n.

```

```

do if ((v(jj,1) <> cmin(v)) and (v(jj,1) <>
cmax(v))).
compute vdich = 0.
break.
end if.
end loop.
do if (vdich = 0).
compute matv = {vmean-vsd; vmean;
vmean+vsd}.
do if ( 999 = 999 and quantile = 0).
do if (matv(1,1) < cmin(v)).
compute matv(1,1)=cmin(v).
compute minprobe=1.
end if.
do if (matv(3,1) > cmax(v)).
compute matv(3,1)=cmax(v).
compute maxprobe=1.
end if.
end if.
compute quantc(1,3) = 1.
do if (quantile = 1).
compute quantd(1,3) = 1.
compute quantc(1,3) = 0.
compute tmp = v.
compute tmp(GRADE(v)) = v.
compute matv = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (vdich = 1).
compute matv = {cmin(v); cmax(v)}.
compute wvdich=1.
compute cmaxv=cmax(v).
compute cminv=cmin(v).
end if.
do if ( 999 <> 999).
compute matv = 999.
compute quantd(1,3) = 0.
compute quantc(1,3) = 0.

```

```

end if.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm(1,3)=vname.
compute modmatp(1,3) = 1.
end if.
end if.
do if (qerr = 0 and qlist = 1).
do if (vnames(:,i)=qname).
compute qerr = 0.
compute qmatch = 1.
compute q = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(q)/n).
compute q = q-meanvec.
compute centvar = {centvar, qname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute qmean = csum(q)/n.
compute tmp = q-(cons*qmean).
compute qsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute qdich = 1.
loop jj = 1 to n.
do if ((q(jj,1) <> cmin(q)) and (q(jj,1) <>
cmax(q))).
compute qdich = 0.
break.
end if.
end loop.
do if (qdich = 0).
compute matq = {qmean-qsd; qmean;
qmean+qsd}.
do if ( 999 = 999 and quantile = 0).
do if (matq(1,1) < cmin(q)).
compute matq(1,1)=cmin(q).
compute minprobe=1.
end if.
do if (matq(3,1) > cmax(q)).
compute matq(3,1)=cmax(q).

```

```

compute maxprobe=1.
end if.
end if.
compute quantc(1,4) = 1.
do if (quantile = 1).
compute quantd(1,4) = 1.
compute quantc(1,4) = 0.
compute tmp = q.
compute tmp(GRADE(q)) = q.
compute matq = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (qdich = 1).
compute matq = {cmin(q); cmax(q)}.
compute cmaxq=cmax(q).
compute cminq=cmin(q).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matq = 999.
compute quantd(1,4) = 0.
compute quantc(1,4) = 0.
end if.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm(1,4)=qname.
compute modmatp(1,4) = 1.
end if.
end if.
do if (vnames(:,i)=cname).
compute cld = dat(:,i).
compute cvname = vnames(:,i).
compute nvarch(1,i)=1.
compute clsmtch = 1.
end if.
loop j = 1 to ncol(mnames).
do if (vnames(:,i)=mnames(1,j)).
compute mmatch = mmatch + 1.

```



```

compute m(:,j)=dat(:,i).
do if (center = 1 and nvarch(1,i) = 0 and mlist >
0 and mcm = 0).
compute meanvec = make(n,1,csum(m(:,j))/n).
compute m(:,j) = m(:,j)-meanvec.
compute centvar = {centvar, mnames(1,j)}.
end if.
compute nvarch(1,i)=1.
compute dichm = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute dichm = 0.
break.
end if.
end loop.
do if ((dichm = 1) and (olsdichm = 1)).
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichm = 1 and model > 3 and mcheck = 0
and oldsdichm = 0).
compute errs= errs+1.
compute runerrs(errs,1) = 1.
compute mcheck = 1.
end if.
do if ((model <= 3) and (ncol(mnames)=1)).
compute nmods = nmods + 1.
compute mmean = csum(m(:,j))/n.
compute tmp = m(:,j)-(cons*mmean).
compute msd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute mdich = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute mdich = 0.
break.
end if.
end loop.
do if (model = 1).

```

```

compute jndich=mdich.
compute jnmin=cmin(m(:,j)).
compute jnmax=cmax(m(:,j)).
end if.
do if (mdich = 0).
compute matm = {mmean-msd; mmean;
mmean+msd}.
do if ( 999 = 999 and quantile = 0).
do if (matm(1,1) < cmin(m(:,j))).
compute matm(1,1)=cmin(m(:,j)).
compute minprobe=1.
end if.
do if (matm(3,1) > cmax(m(:,j))).
compute matm(3,1)=cmax(m(:,j)).
compute maxprobe=1.
end if.
end if.
compute quantc(1,5) = 1.
do if (quantile = 1).
compute quantd(1,5) = 1.
compute quantc(1,5) = 0.
compute tmp = m(:,j).
compute tmp(GRADE(m(:,j))) = m(:,j).
compute matm = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (mdich = 1).
compute matm = {cmin(m); cmax(m)}.
end if.
do if ( 999 <> 999).
compute matm = 999.
compute quantd(1,5) = 0.
compute quantc(1,5) = 0.
end if.
compute modmatv(1,5)=nrow(matm).
compute modmat((1:nrow(matm)), 5) = matm.
compute modvnm(1,5)=mnames(1,j).
compute modmatp(1,5) = 1.

```

```

end if.
end if.
end loop.
end loop.
do if (minprobe=1).
compute note(notes,1) = 14.
compute notes = notes + 1.
end if.
do if (maxprobe=1).
compute note(notes,1) = 15.
compute notes = notes + 1.
end if.
do if (cname <> 'xxx' and clsmatch = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 23.
end if.
do if (cname <> 'xxx').
do if ((cname = zname) or (cname = vname)
or (cname = wname) or (cname = xname) or
(cname = yname) or (cname = qname)).
compute errs = errs+1.
compute runerrs(errs,1) = 24.
end if.
end if.
do if (wlist = 1 and werr = 0 and wmatch = 0).
compute werr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (zlist = 1 and zerr = 0 and zmatch = 0).
compute zerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (qlist = 1 and qerr = 0 and qmatch = 0).
compute qerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (vlist = 1 and verr = 0 and vmatch = 0).

```

```

compute verr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (yerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 16.
end if.
do if (xerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 32.
end if.
do if (model = 6 and nmeds < 2).
compute errs = errs+1.
compute runerrs(errs,1) = 18.
end if.
do if (mmatch < ncol(mnames)).
compute errs=errs+1.
compute runerrs(errs,1) = 25.
end if.
do if (cname <> 'xxx').
compute cld = design(cld).
compute cluster = ncol(cld).
compute cld = cld(:,2:ncol(cld)).
compute clsdmy = ncol(cld).
do if (clsdmy > 19).
compute errs = errs+1.
compute runerrs(errs,1) = 26.
end if.
do if (mcx > 0 or mcm > 0).
compute errs = errs+1.
compute runerrs(errs,1) = 38.
end if.
end if.
compute dichy = 1.
loop jj = 1 to n.
do if ((y(jj,1) <> cmin(y)) and (y(jj,1) <>
cmax(y))).
compute dichy = 0.
break.

```

```

end if.
end loop.
do if ((dichy = 1) and (olsdichy = 1)).
compute dichy=0.
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichy = 1).
do if (toteff=1).
compute toteff=0.
compute note(notes,1) = 31.
compute notes = notes + 1.
end if.
compute jncrit=xp2*xp2.
do if (ws = 1).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 28.
end if.
end if.
compute ncovs = ncol(dat)-rsum(nvarch).
do if (effsize = 1 and covmy <> 0 and ncovs >
0 and model > 3 and model < 7).
compute note(notes,1) = 22.
compute notes = notes + 1.
compute effsize = 0.
end if.
do if (ws = 1 and ncovs < nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 29.
end if.
do if (ws = 1 and ncovs > nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 30.
end if.
end if.
do if (errs = 0).
do if (rsum(quantd) > 0).

```

```

compute note(notes,1) = 4.
compute notes = notes+1.
end if.
do if (rsum(quantc) > 0).
compute note(notes,1) = 5.
compute notes = notes+1.
end if.
do if (ncovs > 0).
compute c = make(n,ncovs,0).
compute cnames = {'x'}.
compute j = 1.
loop i = 1 to ncol(vnames).
do if (nvarch(1,i)) = 0.
compute c(:,j) = dat(:,i).
compute nvarch(1,i)=1.
compute j=j+1.
compute cnames = {cnames, vnames(:,i)}.
end if.
end loop.
compute cnames = cnames(1,2:ncol(cnames)).
do if (ws = 1).
compute covmean=csum(c)/n.
loop i = 1 to ncovs.
compute meanvec = make(n,1, covmean(1,i)).
compute c(:,i) = c(:,i)-meanvec.
end loop.
compute centvar = {centvar, cnames}.
compute x=c(:,1:ncovs).
do if (ncovs=nmads).
compute ncovs=0.
end if.
end if.
end if.
compute names = {yname, xname, mnames,
wname, zname, vname, qname}.
do if (ncovs > 0).
compute names = {names, cnames}.
end if.
do if (dichy = 1 and effsize = 1).
compute note(notes,1) = 2.

```

```

compute notes = notes+1.
end if.
do if (model > 3 and model < 6).
compute indeff=make(nmeds,1,0).
compute indboot=make(boot+1,nmeds,999).
do if (mc > 0).
compute indboot=make(mc+1,nmeds,999).
end if.
do if (effsize = 1 and dichy = 0).
compute rmeff=make(boot+1,nmeds+1,999).
compute abpseff=make(boot+1,nmeds+1,999).
compute abcseff=make(boot+1,nmeds+1,999).
compute pmeff=make(boot+1,nmeds+1,999).
compute r245 = make(boot+1,1,999).
compute kappa2 = make(boot+1,1,999).
end if.
end if.
do if (model = 6 and effsize = 1 and dichy = 0).
compute
rmeff=make(boot+1,ncol(indboot),999).
compute
abpseff=make(boot+1,ncol(indboot),999).
compute
abcseff=make(boot+1,ncol(indboot),999).
compute
pmeff=make(boot+1,ncol(indboot),999).
end if.
do if (nmods > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm(1,tmp) = modvnm(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:nmods).
compute modvnm=modvnm(:,1:nmods).
compute modmatv=modmatv(:,1:nmods).

```

```

loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvals =
make((modmatv(1,1)*modprod(1,1)),
nmods,0).
loop i = 1 to nmods.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvals)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvals(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
do if (model = 74).
compute modvals=matx.
compute modvnm=xname.
end if.
compute vmat = make(8,nrow(modvals),0).
compute vmat(1,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute vmat(5,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute indeff=make(nrow(modvals),1,0).
do if (model <> 5).
compute indboot = make(((boot+1)*nmeds),
nrow(modvals), -99999999).
compute indbootp =
make(boot+1,(nmeds*nmods), -99999999).
do if (imm > 4 and imm < 8).
do if (imm = 7).

```



```

compute immmv1=matw.
compute immmv2=matv.
end if.
do if (imm = 5).
compute immmv1=matw.
compute immmv2=matz.
end if.
do if (imm = 6).
compute immmv1=matv.
compute immmv2=matq.
end if.
compute
indboop1=make((boot+1),(nmeds*(nrow(immmv2))),-9999999).
compute
indboop2=make((boot+1),(nmeds*(nrow(immmv1))),-9999999).
compute onesmv1=make(1,nrow(immmv1),1).
compute onesmv2=make(1,nrow(immmv2),1).
compute
nimvs=nrow(immmv1)+nrow(immmv2).
compute cmmval=0.
compute cmmlbs1={'xx'}.
compute cmmlbs2={'xx'}.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv2}.
compute
cmmlbt=make(nrow(immmv2),1,mnames(1,i)).
compute cmmlbs1={cmmlbs1;cmmlbt}.
end loop.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv1}.
compute
cmmlbt=make(nrow(immmv1),1,mnames(1,i)).
compute cmmlbs2={cmmlbs2;cmmlbt}.
end loop.
compute cmmval=cmmval(2:nrow(cmmval),:).
end if.
end if.
end if.

```

```

do if (nmods > 0).
loop i = 1 to ncol(modvals).
do if (modvnm(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm(1,i)=qname).
compute qcol = i.
end if.
end loop.
end if.
do if (dichy = 1).
compute omx = cmax(y).
compute omn = cmin(y).
compute y = (y = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute data = {cons,y,m,x}.
compute datamed = data.
compute datayed = data.
compute datanm = {'constant'; yname;
t(mnames); xname}.
compute datanmm = {'constant'; yname;
t(mnames); xname}.
compute datanmy = {'constant'; yname;
t(mnames); xname}.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
do if (model < 4 and errs = 0).
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' ', ' '}.
compute datayed = {datayed, x&*m}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).

```

```

compute vmat(1,i) = 1.
compute vmat(2,i) = modvals(i,1).
end loop.
compute mmat = make(16,nmeds,1).
end if.
do if (model = 2 or model = 3).
compute int1 = x&*w.
compute datayed = {datayed, w, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,2).
compute vmat(3,i) = modvals(i,1).
compute vmat(4,i) = modvals(i,1)*modvals(i,2).
end loop.
end if.
do if (model = 3).
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*m.
compute int2 = x&*w&*m.
compute datayed = {datayed, int1, int2}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' X', wname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (model = 4 or model = 5).
compute vmat = make(8,1,1).
end if.
compute yintkey2 = yintkey.
do if (wm = 1).
compute int1 = x&*w.

```

```

compute datamed = {datamed, w,int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmm = {datanmm; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,wcol).
end loop.
do if (zm = 1).
compute int1 = x&*z.
compute datamed = {datamed, z, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(3,i) = modvals(i,zcol).
end loop.
end if.
do if (wzm = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*z.
compute int2 = x&*w&*z.
compute datamed = {datamed, int1, int2}.
loop i = 1 to nrow(modvals).
compute vmat(4,i) =
modvals(i,wcol)*modvals(i,zcol).
end loop.
end if.

```

```

end if.
compute mdatacol=ncol(datamed).
compute mintkey = yintkey.
compute yintkey = {' ',' ',' ',' ',' ',' '}.
compute medints = intcnt-1.
do if (vy = 1 or xmy = 1).
compute mp = 1.
do if (xmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,1).
end loop.
end if.
do if (vy = 1).
compute datayed = {datayed, v}.
compute datanmy = {datanmy; vname}.
compute mmods = 1.
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,vcol).
end loop.
do if (qy = 1).
compute mp = 2.
compute datayed = {datayed,q}.
compute datanmy = {datanmy; qname}.
compute mmods = 2.
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,qcol).
end loop.
end if.
do if (vqy = 1).
compute mp = 3.
compute datayed = {datayed, v&*q}.
compute mmods = 3.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,vcol)*modvals(i,qcol).
end loop.
end if.
end if.
compute mints = make(n,(nmeds*mp),0).
loop i = 0 to (nmeds-1).

```

```

do if (i = 0 and vqy = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
vname, ' X', qname, '', ''}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, '', ''}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (xmy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))&*x.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', xname, '', ''}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (qy = 1).
compute mints(:,((i*mp)+2))=m(:,(i+1))&*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', qname, '', ''}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqy = 1).
compute mints(:,((i*mp)+3))=m(:,(i+1))&*v&*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end loop.

```

```

compute datayed = {datayed, mints}.
end if.
compute mp = 1.
do if (wvmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,wcol)*modvals(i,vcol).
end loop.
end if.
compute mints2 = make(n,(nmeds*mp),0).
do if (wmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,wcol).
end loop.
do if (wy = 0 and model > 3).
compute datayed = {datayed, w}.
compute datanmy = {datanmy; wname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wvmy = 1).
compute datayed = {datayed,w&*v}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints2(:,((i*mp)+1))= m(:,(i+1))&*w.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wvmy = 1).
compute mints2(:,((i*mp)+2))=
m(:,(i+1))&*w&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', vname}.

```

```

compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints2}.
end if.
compute mp = 1.
do if (zmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,zcol).
end loop.
do if (wzmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,zcol)&*modvals(i,wcol).
end loop.
end if.
end if.
do if (zmy = 1).
compute mints3 = make(n,(nmeds*mp),0).
do if (zy = 0).
compute datayed = {datayed, z}.
compute datanmy = {datanmy; zname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wzmy = 1 and wzy = 0).
compute datayed = {datayed,w&*z}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints3(:,((i*mp)+1))= m(:,(i+1))&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.

```



```

compute intcnt = intcnt+1.
do if (wzmy = 1).
compute mints3(:,(i*mp)+2))=
m(:,(i+1))*w*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', zname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints3}.
end if.
compute decoc = 1.
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
do if (wy = 1 and model > 3).
compute datayed = {datayed, w, x*w}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm2(1,1)=wname.
compute modmatp(1,1) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (zy = 1).
compute datayed = {datayed,z,x*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm2(1,2)=zname.
compute modmatp(1,2) = 1.

```

```

compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ',', ' '}.
compute datanmy = {datanmy; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (wzy = 1).
compute datayed = {datayed,w&*z,x&*w&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ',', ' '}.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vxy = 1).
compute datayed = {datayed, x&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm2(1,3)=vname.
compute modmatp(1,3) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, ',', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (qxy = 1).
compute datayed = {datayed, x&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.

```

```

compute modvnm2(1,4)=qname.
compute modmatp(1,4) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqxy = 1).
compute datayed = {datayed, x&*v&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, 'X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end if.
do if (wvxy = 1).
compute datayed = {datayed, x&*w&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, 'X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute modvalsd = 0.
compute ttt = rsum(modmatp).
do if (rsum(modmatp) > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm2(1,tmp) = modvnm2(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.

```

```

end loop.
compute modmat=modmat(:,1:ttt).
compute modvnm2=modvnm2(:,1:ttt).
compute modmatv=modmatv(:,1:ttt).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvalsd =
make((modmatv(1,1)*modprod(1,1)), ttt,0).
loop i = 1 to ttt.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvalsd)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvalsd(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
end if.
do if (ttt > 0).
loop i = 1 to ncol(modvalsd).
do if (modvnm2(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm2(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm2(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm2(1,i)=qname).
compute qcol = i.

```

```

end if.
end loop.
compute directv = make(nrow(modvalsd),1,1).
do if (wy = 1).
compute directv ={directv, modvalsd(:,wcol)}.
end if.
do if (zy = 1).
compute directv ={directv, modvalsd(:,zcol)}.
end if.
do if (wzy = 1).
compute directv ={directv,
(modvalsd(:,wcol)&*modvalsd(:,zcol))}.
end if.
do if (vxy = 1).
compute directv ={directv, modvalsd(:,vcol)}.
end if.
do if (qxy = 1).
compute directv ={directv, modvalsd(:,qcol)}.
end if.
do if (vqxy = 1).
compute directv ={directv,
(modvalsd(:,vcol)&*modvalsd(:,qcol))}.
end if.
do if (wvxy = 1).
compute directv ={directv,
(modvalsd(:,vcol)&*modvalsd(:,wcol))}.
end if.
end if.
compute ydatacol=ncol(datayed).
do if (ncovs > 0).
do if (covmy <> 2).
compute datamed = {datamed,c}.
end if.
do if (covmy <> 1).
compute datayed = {datayed,c}.
end if.
do if (covmy = 1 and model = 4 and mcx > 0).
compute datayed = {datayed,c}.
end if.
compute covmeans = csum(c)/n.

```

```

end if.
do if (cluster > 0).
compute datamed = {datamed,cld}.
compute datayed = {datayed, cld}.
compute cldmeans = csum(cld)/n.
end if.
compute mst = 3.
compute mnd = mst+nmeds-1.
compute ydatacol=ncol(datayed).
compute mdatacol=ncol(datamed).
do if (ncovs > 0).
compute datanmy = {datanmy; t(cnames)}.
do if (model > 3).
compute datanmm = {datanmm; t(cnames)}.
end if.
end if.
compute datanmy = {'constant';
datanmy(3:nrow(datanmy),1)}.
do if (model > 3).
compute datanmm = {'constant';
datanmm(3:nrow(datanmm),1)}.
end if.
compute amm = make(2,1,0).
compute abmm = make(2,1,0).
compute mnv = csum(datayed(:,2)/n).
compute mnv = make(n,1,mnv).
compute ssty = csum((datayed(:,2)-mnv)**2).
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
do if (ws=1).
compute stddevm=diag(sigma).
compute
stddevm=sqrt(stddevm(3:(nmeds+2),1)).
end if.
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute r2xy =
(sigma(2,(3+nmeds)))/(stddevy*stddevx)**2.

```

```

compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))**2.
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).
do if (model = 4 and nmeds = 1 and cluster = 0
and ncovs = 0).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
compute datatm = datamed.
compute dataty = datayed.
compute mdlrms2 = { '1' ; yname; xname}.
compute mdlrms = {'Model ='; ' Y ='; ' X ='
}.
do if (ws = 1).
compute mdlrms2 = { '1' ; yname}.
compute mdlrms = {'Model ='; ' Y ='}.
end if.

```

```

loop i = 1 to ncol(mnames).
compute mdlInms2 = {mdlInms2; mnames(1,i)}.
do if (i = 1 and ncol(mnames) = 1).
compute mdlInms = {mdlInms; ' M = '}.
else.
compute mdlInms = {mdlInms; mlab(i,1)}.
end if.
end loop.
do if (wname <> 'xxx').
compute mdlInms2 = {mdlInms2; wname}.
compute mdlInms = {mdlInms; ' W = '}.
end if.
do if (zname <> 'xxx').
compute mdlInms2 = {mdlInms2; zname}.
compute mdlInms = {mdlInms; ' Z = '}.
end if.
do if (vname <> 'xxx').
compute mdlInms2 = {mdlInms2; vname}.
compute mdlInms = {mdlInms; ' V = '}.
end if.
do if (qname <> 'xxx').
compute mdlInms2 = {mdlInms2; qname}.
compute mdlInms = {mdlInms; ' Q = '}.
end if.
do if (jn = 1 and model = 1 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
do if (jn = 1 and model = 3 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
compute mdpbe=0.
compute medte2=0.
do if ( 1=4 and mcx > 0).
compute medte2=1.
end if.
do if (((mcx > 0) or (mcm > 0)) AND model =
1).
compute mdpbe=1.

```



```

compute dd={datayed(:,4),datayed(:,3)}.
do if (ncovs > 0).
compute dd={datayed(:,6:(5+ncovs)),dd}.
end if.
compute dd={datayed(:,2),dd}.
end if.
compute yes = ((nrow(modvals)=1) and
(contrast = 1) and (model = 3 or model = 2)).
compute yes2 = ((nrow(modvals)=1) and
(contrast = 1) and (model = 2)).
print/title = '***** PROCESS Procedure
for SPSS Release 2.16.3 *****'.
print/title = '      Written by Andrew F. Hayes,
Ph.D.    www.afhayes.com'.
print/title = '  Documentation available in
Hayes (2013).
www.guilford.com/p/hayes3/space=0.
print mdlnms2/title =
*****
*****'/rnames = mdlnms/format =
a8.
do if (ncovs > 0).
print cnames/title = 'Statistical Controls:'/rlabels
= 'CONTROL='/'format a8.
end if.
print n/title = 'Sample size'/'format F10.0.
do if (cluster > 0).
print cluster/rnames = cvname/title =
'Clustering Variable and Number of Clusters'.
end if.
do if (model > 3 and ( 'random' <> 'random')).
compute seedt= 'random'.
print seedt/title='Custom'/'format=A12/rlabel =
'Seed:'.
end if.
do if (mdpbe=0 and medte2 = 0).
loop bt = 1 to (boot+1).
do if (bt = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)),-999).

```

```

end if.
compute bootcoef = 0.
do if (bt > 1).
loop.
compute v=trunc(uniform(n,1)*n)+1.
compute datayed = dataty(v,:).
do if (ws=1).
compute
bmeans=csum(datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed)))/n.
compute cntmat=make(n,nmeds,1).
compute bmeans2=bmeans.
do if (nmeds > 1).
compute bmeans2=mdia(bmeans).
end if.
compute cntmat=cntmat*bmeans2.
compute datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed))=datayed(:,(ncol(data
yed)-nmeds+1):ncol(datayed))-cntmat.
end if.
compute detcheck=det(t(datayed)*datayed).
compute rk=(detcheck <> 0).
do if (model > 3).
compute datamed = datatm(v,:).
compute detcheck=det(t(datamed)*datamed).
do if (rk=1).
compute rk=(detcheck <> 0).
end if.
end if.
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
compute temp=diag(sigma).
compute bad = bad+(1-rk).
compute false = 1.
end loop if (rk = 1).
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).

```

```

compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).
do if (model = 4 and nmeds = 1 and ncovs = 0
and cluster = 0).
compute r2xy =
(sigma(2,4)/(stddevy*stddevx))&**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))&**2.
compute sstot = sigma(2,2)*(n-1).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
end if.
do if (model > 3).
loop im = 1 to nmeds.
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.

```

```

compute xnm = {'constant';
datanmm((2+nmeds):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
do if (model = 6).
do if (im = 1).
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
end if.
do if (im > 1).
compute xm={cons, datamed(:,3:(im+1)),
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant'; datanmm(2:im,1);
datanmm((mnd):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
compute
mmpaths((im+1),(2:im))=t(coeff(2:im,1)).
end if.
end if.
do if (ws = 1).
compute
coeff(1,1)=csum(datamed(:,(2+im)))/n.
end if.
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
do if (bt = 1).
compute resid=datamed(:,(2+im))-xm*coeff.
compute sse =cssq(resid).
compute mse = sse/(n-ncol(xm)).
compute mnv = csum(data(:,(2+im)))/n.
compute mnv = make(n,1,mnv).
compute sstm = csum((data(:,(2+im))-
mnv)**2).

```

```

compute k3 = nrow(coeff).
do if (hc3 = 1).
compute h = xm(:,1).
loop i3=1 to n.
compute h(i3,1)= xm(i3,:)*invXtX*t(xm(i3,:)).
end loop.
loop i3=1 to k3.
compute xm(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xm(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xm(:,i3) = sqrt(mse)&*xm(:,i3).
end loop.
end if.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute hccov=invXtX*t(xm)*xm*invXtX.
compute mcmats(im,im)=hccov(2,2).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*hccov*lmat)*((t(lmat)
)*coeff))/dfnum.
compute coeff = coeff(1:(nrow(coeff)-
clsdmy),1).
compute mccoef(im,1)=coeff(2,1).
compute standerr =
sqrt(diag(invXtX*t(xm)*xm*invXtX)).
compute standerr = standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1).
compute standerr(1,1)=stddevm(im,1)/sqrt(n).
end if.
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-
ncol(xm)))).
compute temp=(n-ncol(xm)).
do if (ws=1).

```

```

compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute temp=(n-1).
end if.
compute xd = abs(xp2).
compute temp = (temp* (exp((temp-
(5/6))*((xd/(temp-(2/3)+(.11/temp)))*(xd/(temp-
(2/3)+(.11/temp)))))-1)).
compute temp1 = coeff-
sqrt(abs(temp))*standerr.
compute temp2 =
coeff+sqrt(abs(temp))*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (ws=1).
compute op=op(1,:).
compute xnmn =
apathnam(((nmeds>1)+1+(im-1)),1).
end if.
compute sobel(im,1) = coeff(2,1).
compute sobel(im,2)=standerr(2,1).
compute temp = mnames(1,im).
compute r2full = 1-(sse/sstm).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute summ = {sqrt(r2full), r2full, mse,
Fratio, dfnum, dfden, pfr}.
do if (detail = 1).
print temp/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.
do if (ws <> 1).
print summ/title = 'Model Summary'/'clabels 'R',
'R-sq', 'MSE','F', 'df1', 'df2', 'p'/'format = F10.4.
end if.
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
print op/title = 'Model'/'rnames = xnmn/'clabels
'coeff', 'se', 't', 'p', 'LLCI', 'ULCI'/'format = F10.4.
do if (covcoeff=1 and ws <> 1).

```

```

compute
hccovtmp=hccov(1:nrow(op),1:nrow(op)).
compute cnamestp=t(xmnm).
print hccovtmp/title='Covariance matrix of
regression parameter
estimates'/rnames=xmnm/cnames=cnamestp/f
ormat= F10.4.
end if.
do if (nmods > 0 and nrow(mintkey) > 1).
print mintkey/title = 'Product terms key:.'/format
= A8.
end if.
do if (ws = 1).
compute wsdf=n-1.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
end if.
end if.
compute ymat(1,im) = coeff(2,1).
do if (ws = 1).
compute ymat(1,im)=coeff(1,1).
end if.
do if (wm = 1).
compute ymat(2,im) = coeff(4,1).
do if (zm = 1).
compute ymat(3,im) = coeff(6,1).
do if (wzm = 1).
compute ymat(4, im) = coeff(8,1).
end if.
end if.
end if.
do if (model = 6).
compute mmpaths((im+1),1)=coeff((im+1),1).
end if.
end loop.
end if.
loop totlp = 1 to (1+(toteff*(bt = 1))).
do if (toteff = 1 and totlp =2).
compute xy =
{cons,datayed(:,(3+nmeds):ydatacol)}.

```

```

else.
compute xy={cons, datayed(:,3:ydatacol)}.
end if.
do if (dichy = 1).
compute pt2 =
make(nrow(datayed(:,2)),1,(csum(datayed(:,2))
/n)).
compute LL3 = datayed(:,2)*ln(pt2)+(1-
datayed(:,2))*ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xy),1,0).
compute LL1 = 0.
compute xy22=xy.
loop jjj = 1 to iterate.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.
compute coeff =
bt1+inv(t(xy22)*xy)*t(xy)*(datayed(:,2)-pt1).
compute pt1 = 1/(1+exp(-(xy*coeff))).
compute itprob = csum((pt1 <
.00000000000001) or (pt1 >
.99999999999999)).
do if (itprob = 0).
compute LL = datayed(:,2)*ln(pt1)+(1-
datayed(:,2))*ln(1-pt1).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
break.
end if.
compute bt1 = coeff.
compute LL1 = LL2.
end loop.
do if (jjj >= iterate and iterr = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
compute iterr = 1.
end if.

```



```

loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)&*pt1&*(1-pt1).
end loop.
compute covmat = inv(t(xy22)*xy).
release xy22.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
end if.
do if (dichy = 0).
compute invXtX = inv(t(xy)*xy).
compute coeff =invXtX*t(xy)*datayed(:,2).
do if (effsize=1 and model > 3 and model < 7
and (ncovs > 0 or clsdmy > 0)).
compute xysdy =
{cons,datayed(:,(4+nmeds):ydatacol)}.
compute coeffsd
=inv(t(xysdy)*xysdy)*t(xysdy)*datayed(:,2).
compute resid=datayed(:,2)-(xysdy*coeffsd).
compute sdycov =sqrt(csum(resid&*resid)/(n-
ncol(xysdy))).
compute xvaron=datayed(:,(3+nmeds)).
compute
coeffx2=inv(t(xysdy)*xysdy)*t(xysdy)*xvaron.
compute residx2=xvaron-(xysdy*coeffx2).
compute sdxcov
=sqrt(csum(residx2&*residx2)/(n-ncol(xysdy))).
end if.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4 and bt > 1).
compute resid=datayed(:,2)-xy*coeff.
compute sse = cssq(resid).
compute r2full = 1-(sse/sstot).

```

```

end if.
do if (bt = 1).
compute resid=data(:,2)-xy*coeff.
compute k3 = nrow(coeff).
compute sse = cssq(resid).
compute mse = sse/(n-ncol(xy)).
compute
hcnott=coeff&/(sqrt(diag(mse*invXtX))).
do if (hc3 = 1).
compute h = xy(:,1).
loop i3=1 to n.
compute h(i3,1)= xy(i3,:)*invXtX*t(xy(i3,:)).
end loop.
loop i3=1 to k3.
compute xy(:,i3) = (resid(:,ncol(resid))&/(1-
h))*xy(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xy(:,i3) = sqrt(mse)*xy(:,i3).
end loop.
end if.
compute covmat = (invXtX*t(xy)*xy*invXtX).
end if.
end if.
do if (bt = 1).
do if (model = 2).
compute xy2={cons, datayed(:,3:ydatacol)}.
compute temp = ncol(xy2).
do if (temp > 6).
compute xy3=xy2(:,7:temp)}.
end if.
compute xy2={xy2(:,1:3), xy2(:,5)}.
do if (temp > 6).
compute xy2={xy2, xy3}.
release xy3.
end if.
compute invXtX = inv(t(xy2)*xy2).
compute coeff2 =invXtX*t(xy2)*datayed(:,2).

```

```

compute ssem2=cssq(datayed(:,2)-xy2*coeff2).
release xy2.
end if.
compute standerr = sqrt(diag(covmat)).
do if (totlp =1).
compute
mcmats((nmeds+1):ncol(mcmats),(nmeds+1):n
col(mcmats))=covmat(2:(1+nmeds),2:(1+nmed
s)).
end if.
compute standerr=standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1 and totp=2).
compute standerr(1,1)=stddevy/sqrt(n).
end if.
compute coeffplt = coeff.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*coeff)))/dfnum).
compute coeff=coeff(1:(nrow(coeff)-clsdmy),1).
do if (totlp=1).
compute
mccoeff((nmeds+1):nrow(mccoeff))=coeff(2:(1+
nmeds),1).
end if.
compute bbbb=coeff(2,1).
do if totp = 1.
compute deco(1,1)=2+nmeds.
compute deco = deco(1:decoc,1).
compute covdirt = make((nrow(covmat)-
clsdmy),(ncol(covmat)-clsdmy),0).
compute covdirt = covmat(deco,:).
compute covdir =
make(nrow(covdirt),ncol(covdirt),0).
compute covdir = covdirt(:,t(deco)).
compute deco=coeff(deco,1).

```

```

do if (ttt > 0).
compute sedir =
sqrt(diag(directv*covdir*t(directv))).
compute directv = directv*deco.
end if.
compute sobel(:,3)=coeff(2:(1+nmeds),1).
compute sobel(:,4) = standerr(2:(1+nmeds),1).
compute sobel2 = sobel&*sobel.
do if (varorder <> 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)&*sobel2(:,4)+sobel2(:,3)&*sob
el2(:,2)).
end if.
do if (varorder = 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)&*sobel2(:,4)+sobel2(:,3)&*sob
el2(:,2)+sobel2(:,2)&*sobel2(:,4)).
end if.
compute sobel(:,1) = sobel(:,1)&*sobel(:,3).
compute sobel(:,3) = sobel(:,1)&/sobel(:,2).
compute sobel(:,4) = 2*(1-
cdfnorm(abs(sobel(:,3)))).
end if.
do if (dichy = 0).
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
do if (ws= 1 and totlp=2).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute dfden=n-1.
end if.
compute cnms = {'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute op = {coeff, standerr, tratio, p}.
end if.
do if (dichy = 1).
compute tratio = (coeff&/standerr).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute wald = tratio&*tratio.
compute cnms = {'coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.

```

```

compute temp=coeff-abs(xp2)*standerr.
compute op = {coeff, standerr, tratio, p, temp}.
compute temp=coeff+abs(xp2)*standerr.
compute op = {op, temp}.
end if.
do if (detail = 1).
do if (totlp = 2).
print yname/title = '*****'
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:.'/format = A8.
end if.
do if (totlp <> 2).
print yname/title =
*****
*****'/rlabels = 'Outcome:.'/format
= A8.
end if.
end if.
do if (dichy = 1 and bt = 1 and totp = 1).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:.'/cnames = nmsd/format = F9.2.
end if.
do if (dichy = 0).
compute r2full = 1-(sse/ssty).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute jndf=dfden.
compute wsdf=dfden.
compute xd = abs(xp2).
compute jncrit = (dfden* (exp((dfden-
(5/6))*((xd/(dfden-
(2/3)+(.11/dfden)))*(xd/(dfden-
(2/3)+(.11/dfden)))))-1)).
compute summ = {sqrt(r2full), r2full, mse,
fratio, dfnum, dfden, pfr}.
compute temp1=coeff-sqrt(jncrit)*standerr.
compute temp2=coeff+sqrt(jncrit)*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.

```

```

do if (detail = 1).
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels =
'R', 'R-sq', 'MSE' 'F', 'df1', 'df2', 'p'/format =
F10.4.
end if.
end if.
end if.
do if (dichy = 1).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(coeff)-
1)).
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
do if (detail = 1).
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format =
F10.4.
end if.
end if.
do if (totlp = 2).
compute datanmy={'constant';
datanmy((nmeds+2):nrow(datanmy),1)}.
end if.
do if (detail = 1).
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
do if (ws=1).
compute op2=op(1:(1+nmeds),:).
compute datanmy2 = {'c" path'; bpathnam(
(1+(nmeds>1)): ((1+(nmeds>1))+(nmeds-
1)),1)}.
do if (totlp=2).
compute datanmy2={'c path'}.
compute op2=op(1,:).

```

```

compute wsdf=n-1.
end if.
print op2/title = 'Model'/rnames =
datanmy2/cnames = cnms/format = F10.4.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
do if (ws <> 1).
print op/title = 'Model'/rnames =
datanmy/cnames = cnms/format = F10.4.
do if (covcoeff=1).
compute
covmattp=covmat(1:nrow(op),1:nrow(op)).
compute cnamestp=t(datanmy).
print covmattp/title='Covariance matrix of
regression parameter
estimates'/rnames=datanmy/cnames=cnamest
p/format= F10.4.
end if.
end if.
end if.
do if (ttt = 0 and totlp = 1).
compute deco = op((nmeds+2),:).
do if (ws = 1).
compute deco=op(1,:).
end if.
end if.
do if (ttt = 0 and totlp = 2).
compute decotot = op(2,:).
do if (ws = 1).
compute decotot=op(1,:).
end if.
end if.
do if (nmods > 0 and model > 4 and detail = 1
and nrow(yintkey) > 1)).
print yintkey/title = 'Product terms key: '/format
= A8.
end if.
do if (nmods > 0 and model < 4 and detail = 1).
print yintkey2/title = 'Product terms key: '/format
= A8.

```

```

do if ((model = 1 or model = 2) and dichy = 0).
compute temp={((op(4,3)**2)*(1-r2full))/dfden,
op(4,3)**2, 1, dfden, op(4,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(4,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(2,1).
do if (model = 2).
compute temp={temp;((op(6,3)**2)*(1-
r2full))/dfden, op(6,3)**2, 1, dfden, op(6,4)}.
do if (hc3=1).
compute temp(2,1)=((hcnot(6,1)**2)*(1-
r2full))/dfden.
end if.
compute frat2=(dfden*(r2full-(1-
(ssem2/ssty))))/(2*(1-r2full)).
do if (hc3=1).
compute lmat=make(nrow(coeffplt),2,0).
compute lmat(4,1)=1.
compute lmat(6,2)=1.
compute frat2 =
(t(t(lmat)*coeffplt)*inv(t(lmat)*covmat*lmat)*((t(l
mat)*coeffplt))/2).
end if.
compute temp={temp;(r2full-(1-
(ssem2/ssty))),frat2,2,dfden,1-
fcdf(frat2,2,dfden)}.
compute rnms={rnms; yintkey2(3,1),'Both'}.
end if.
print temp/title = 'R-square increase due to
interaction(s):'/rnames=rnms/clabels = 'R2-
chng', 'F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (model = 3 and dichy = 0).
compute temp={((op(8,3)**2)*(1-r2full))/dfden,
op(8,3)**2, dfden, op(8,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(8,1)**2)*(1-
r2full))/dfden.

```



```

end if.
compute rnms=yintkey2(5,1).
print temp/title = 'R-square increase due to
three-way interaction:.'/rnames=rnms/clabels =
'R2-chng', 'F(1,df2)', 'df2', 'p'/format = F10.4.
end if.
end if.
end if.
do if (model = 6 and totlp = 1).
compute
mmpaths(nrow(mmpaths),1)=coeff(nrow(mmpa
ths),1).
compute
mmpaths(nrow(mmpaths),(2:(nmeds+1)))=t(co
eff(2:(nmeds+1),1)).
end if.
do if (totlp = 1).
loop im = 1 to nmeds.
do if (model < 4).
compute ymat(1,im) = coeff(3,1)*(1-yes).
compute ymat(2,im) = coeff(4,1).
compute cmat(1,im) = covmat(3,3).
compute cmat(2,im) = covmat(4,4).
compute cmat(5,im) = covmat(3,4).
compute jnb1=coeff(3,1).
compute jnb3=coeff(4,1).
compute jnsb1=covmat(3,3).
compute jnsb3=covmat(4,4).
compute jnsb1b3=covmat(3,4).
do if (model = 2 or model = 3).
compute ymat(3,im) = coeff(6,1).
compute cmat(3,im) = covmat(6,6).
compute cmat(6,im) = covmat(3,6).
compute cmat(8,im) = covmat(4,6).
end if.
do if (model = 3).
compute ymat(4,im) = coeff(8,1).
compute cmat(4,im) = covmat(8,8).
compute cmat(7,im) = covmat(3,8).
compute cmat(9,im) = covmat(4,8).

```

```

compute cmat(10,im) = covmat(6,8).
compute jnb1=coeff(4,1).
compute jnb3=coeff(8,1).
compute jnsb1=covmat(4,4).
compute jnsb3=covmat(8,8).
compute jnsb1b3=covmat(4,8).
end if.
end if.
do if (model > 3).
compute ymat(5,im) = coeff((1+im),1).
end if.
do if (xmy = 1).
compute ymat(6,im) = coeff((2+nmeds+im),1).
end if.
do if (vy = 1).
compute ymat(6,im) = coeff((3+nmeds+im),1).
end if.
do if (qy = 1 and vy = 1).
compute ymat(6,im) = coeff((5+nmeds+((im-1)*2)),1).
compute ymat(7,im) = coeff((6+nmeds+((im-1)*2)),1).
end if.
do if (vqy = 1).
compute ymat(6,im) = coeff((6+nmeds+((im-1)*3)),1).
compute ymat(7,im) = coeff((7+nmeds+((im-1)*3)),1).
compute ymat(8,im) = coeff((8+nmeds+((im-1)*3)),1).
end if.
do if (wmy = 1).
compute ymat(7,im) = coeff((3+nmeds+im-wy),1).
end if.
do if (wmy = 1 and vy = 1).
compute ymat(7,im) =coeff((4+(nmeds*2)+im-wy),1).
end if.
do if (wmy = 1 and vy = 1 and wvmy = 1).

```

```

compute ymat(7,im) =
coeff((6+(nmeds*2)+((im-1)*2)-wy),1).
compute ymat(8,im) =
coeff((7+(nmeds*2)+((im-1)*2)-wy),1).
end if.
do if (wmy = 1 and zmy = 1).
compute ymat(6,im) = coeff((6-(wzy*3)+(wzm-
1)+(nmeds*2)+((im-1)*2)*wzm)+((im-1)*(1-
wzm))-((zy-wzy)*2),1).
compute ymat(7,im) = coeff((3-
zy+im+nmeds),1).
do if (wzmy = 1).
compute ymat(8,im) = coeff((7-
(wzy*3)+(nmeds*2)+((im-1)*2)),1).
end if.
end if.
do if (nmods > 0 and model <> 5).
loop indlp = 1 to nrow(modvals).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp)).
do if (model > 6).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp))*csum(ym
at(5:8,im)*vmat(5:8,indlp)).
end if.
end loop.
compute indboot((bt+(im-
1)*(boot+1)),:)=t(indeff).
do if (imm = 1 or imm = 3).
compute
indbootp(bt,im)=ymat(2,im)*ymat(5,im).
do if (imm=3).
compute
indbootp(bt,(im+nmeds))=ymat(3,im)*ymat(5,i
m).
do if (zdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxz-cminz).
end if.

```

```

end if.
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
end if.
end if.
do if ((imm= 2 or imm = 4) or model = 74).
compute
indbootp(bt,im)=ymat(1,im)*ymat(6,im).
do if (imm=4).
compute
indbootp(bt,(im+nmeds))=ymat(1,im)*ymat(7,i
m).
do if (qdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxq-cminq).
end if.
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
end if.
end if.
do if (imm=7).
compute
indbootp(bt,im)=ymat(2,im)*ymat(6,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(2,im)*y
mat(6,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(6,im)*onesmv1)+(ymat(2,im)*y
mat(6,im)*t(immmv1)).
do if (wdich=1).

```

```

compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxv-cminv).
end if.
end if.
do if (imm=5).
compute
indbootp(bt,im)=ymat(4,im)*ymat(5,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(4,im)*y
mat(5,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(3,im)*ymat(5,im)*onesmv1)+(ymat(4,im)*y
mat(5,im)*t(immmv1)).
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).

```

```

end if.
do if (zdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxz-
cminz).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxz-cminz).
end if.
end if.
do if (imm=6).
compute
indbootp(bt,im)=ymat(1,im)*ymat(8,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(1,im)*ymat(6,im)*onesmv2)+(ymat(1,im)*y
mat(8,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(7,im)*onesmv1)+(ymat(1,im)*y
mat(8,im)*t(immmv1)).
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxv-cminv).
end if.
do if (qdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxq-
cminq).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-

```

```

1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxq-cminq).
end if.
end if.
do if ((model = 58 or model = 59) and
wvdich=1).
compute
indbootp(bt,im)=(ymat(1,im)*ymat(7,im)*(cmax
w-cminw))+(ymat(2,im)*ymat(5,im)*(cmaxw-
cminw))+(ymat(2,im)*ymat(7,im)*((cmaxw*cma
xw)-(cminw*cminw))).
end if.
end if.
do if (model = 4 or model = 5).
compute
indboot(bt,im)=csum(ymat(1:4,im)*vmat(1:4,1
))*csum(ymat(5:8,im)*vmat(5:8,1)).
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.000000000000001.
end if.
compute sumind=rsum(indboot(bt,:)).
do if (im=nmeds).
compute
pmeff(bt,2:(im+1))=indboot(bt,:)/(sumind+coeff(
(2+nmeds),1)).
end if.
compute
rmeff(bt,(im+1))=indboot(bt,im)/coeff((2+nmeds
),1).
compute
abpseff(bt,(im+1))=indboot(bt,im)/stddevy.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*stddevx.
do if (ncovs > 0 or clsdmy > 0).
compute
abpseff(bt,(im+1))=indboot(bt,im)/sdy cov.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*sdx cov.
end if.

```

```

do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,1) = r2my-(r2full-r2xy).
compute abmmr = 1.
do if (indboot(bt,im) < 0).
compute abmmr = cmin(abmm).
end if.
do if (indboot(bt,im) > 0).
compute abmmr = cmax(abmm).
end if.
compute kappa2(bt,1)=indboot(bt,im)/abmmr.
compute tmp = indboot(bt,im)/abmmr.
end if.
end if.
end if.
do if (model = 6).
do if (nmeds = 2).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(4,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3).
compute
indboot(bt,3)=mmpaths(3,1)*mmpaths(4,3).
end if.
do if (nmeds = 3).
compute indboot(bt,1) =
mmpaths(2,1)*mmpaths(5,2).
compute indboot(bt,2) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3).
compute indboot(bt,3) =
mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4).
compute indboot(bt,4) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4).
compute indboot(bt,5) =
mmpaths(3,1)*mmpaths(5,3).
compute indboot(bt,6) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4).

```



```

compute indboot(bt,7) =
mmpaths(4,1)*mmpaths(5,4).
end if.
do if (nmeds = 4).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(6,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(6,3).
compute
indboot(bt,3)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(6,4).
compute
indboot(bt,4)=mmpaths(2,1)*mmpaths(5,2)*mmpaths(6,5).
compute
indboot(bt,5)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(6,4).
compute
indboot(bt,6)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3)*mmpaths(6,5).
compute
indboot(bt,7)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,8)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,9)=mmpaths(3,1)*mmpaths(6,3).
compute
indboot(bt,10)=mmpaths(3,1)*mmpaths(4,3)*mmpaths(6,4).
compute indboot(bt,11)
=mmpaths(3,1)*mmpaths(5,3)*mmpaths(6,5).
compute indboot(bt,12) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,13) =
mmpaths(4,1)*mmpaths(6,4).

```

```

compute indboot(bt,14) =
mmpaths(4,1)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,15) =
mmpaths(5,1)*mmpaths(6,5).
end if.
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.00000000000001.
end if.
compute
pmeff(bt,:)=indboot(bt,:)/(rsum(indboot(bt,:))+m
mpaths(nrow(mmpaths),1)).
compute rmeff(bt,:) =
indboot(bt,:)/mmpaths(nrow(mmpaths),1).
compute abpseff(bt,:)=indboot(bt,:)/stddevy.
compute
abcseff(bt,:)=(stddevx*indboot(bt,:))/stddevy.
do if (ncovs > 0 or clsdmy > 0).
compute abpseff(bt,:)=indboot(bt,:)/sdy cov.
compute
abcseff(bt,:)=(sdxcov*indboot(bt,:))/sdy cov.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,:) = r2my-(r2full-r2xy).
end if.
end if.
end if.
end loop.
end if.
end loop.
do if (savboot = 1 and bt > 1).
compute bootstrp((bt-1),:)=bootcoef.
end if.
end loop.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.

```

```

release datayed, dat, datamed, data, y, x,
datatm, dataty, mints2, cons, mnv, tmp.
do if (mc > 0).
compute x1 = sqrt(-
2*ln(uniform(mc,nrow(mcmats))))&*cos((2*3.14
159265358979)*uniform(mc,nrow(mcmats))).
compute x1=x1*chol(mcmats).
loop i=1 to nrow(x1).
compute x1(i,:)=x1(i,:)+t(mccoeff).
end loop.
loop i = 1 to nmeds.
compute
indboot(2:nrow(indboot),i)=x1(:,i)&*x1(:,(i+nme
ds)).
end loop.
end if.
do if (ttt = 0 and model > 3).
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
end if.
do if (model <> 74).
do if (dichy = 0).
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 't', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 't', 'p', 'LLCI', 'ULCI'/format =
F10.4.
else.
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 'Z', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.

```

```

print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 'Z', 'p', 'LLCI', 'ULCI' /format =
F10.4.
end if.
end if.
end if.
do if (ttt > 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
compute clbs = {modvnm2, 'Effect', 'SE', 't', 'p',
'LLCI', 'ULCI'}.
compute tratio = (directv&/sedir).
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy))))).
compute outp = {modvalsd, directv, sedir,
tratio, p}.
do if (dichy = 0).
compute temp1=directv-sqrt(jncrit)*sedir.
compute temp2=directv+sqrt(jncrit)*sedir.
compute outp = {outp, temp1, temp2}.
end if.
do if (dichy = 1).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute temp = directv-abs(xp2)*sedir.
compute outp = {outp, temp}.
compute temp = directv+abs(xp2)*sedir.
compute outp = {outp, temp}.
compute clbs = {modvnm2, 'Effect', 'SE', 'Z', 'p',
'LLCI', 'ULCI'}.
end if.
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
print outp/title = 'Conditional direct effect(s) of
X on Y at values of the moderator(s):'/format =
F10.4 /cnames = clbs.
end if.
do if (nmods > 0 and model <> 5).
do if (model < 4).

```

```

print/title =
*****
*****!

compute zmat(1,1) = 1*(1-yes).
compute cfse = make(nrow(modvals),1,0).
loop #m = 1 to nrow(modvals).
do if (model = 1).
compute zmat(2,1)=modvals(#m,1)**2.
compute zmat(5,1)=2*modvals(#m,1).
end if.
do if (model = 2 or model = 3).
compute zmat(2,1)=modvals(#m,2)**2.
compute zmat(3,1)=modvals(#m,1)**2.
compute
zmat(4,1)=(modvals(#m,1)**2)*(modvals(#m,2)
**2)*(1-yes2).
compute zmat(5,1)=2*modvals(#m,2)*(1-yes).
compute zmat(6,1)=2*modvals(#m,1)*(1-yes).
compute
zmat(7,1)=2*modvals(#m,1)*modvals(#m,2)*(1-
yes).
compute
zmat(8,1)=2*modvals(#m,1)*modvals(#m,2).
compute
zmat(9,1)=2*modvals(#m,1)*(modvals(#m,2)**
2)*(1-yes2).
compute
zmat(10,1)=2*(modvals(#m,1)**2)*modvals(#m
,2)*(1-yes2).
end if.
compute cfse(#m,1)=sqrt(csum(zmat&*cmat)).
end loop.
end if.
do if (nmods > 0).
compute clbs = {modvnm, 'Effect'}.
loop im = 1 to nmeds.
compute obs = t(indboot(1+(im-1)*(boot+1),:)).
compute outp = {modvals, obs}.
do if (model < 4).
compute tstat = obs&/cfse.

```

```

do if (dichy = 0).
compute pval = 2*(1-tcdf(abs(tstat), (n-
ncol(xy)))).
compute temp=obs-sqrt(jncrit)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp=obs+sqrt(jncrit)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 't', 'p', 'LLCI', 'ULCI'}.
compute jnclbs=clbs.
end if.
do if (dichy = 1).
compute pval = 2*(1-cdfnorm(abs(tstat))).
compute temp = obs-abs(xp2)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp = obs+abs(xp2)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute jnclbs=clbs.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indboot((1+(im-
1)*(boot+1)),:).
compute indboot2 = indboot((2+(im-
1)*(boot+1)):(1+(im-1)*(boot+1)+boot),:).
compute mnind = t(csum(indboot2)/boot).
compute stdind=t(sqrt((cssq(indboot2)-
((csum(indboot2)**2)/boot))/(boot-1))).
compute llci=make(1,ncol(indboot2),-999).
compute ulci=make(1,ncol(indboot2),-999).
loop #e = 1 to ncol(indboot2).
.
compute temp = indboot2(:,#e).
compute temp(GRADE( indboot2(:,#e) )) =
indboot2(:,#e).
compute badlo = 0.
compute badhi = 0.

```

```

do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.

```

```

do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {modvals, obs, stdind, t(llci),
t(ulci)}.
compute clbs = {modvnm, 'Effect', 'Boot SE',
'BootLLCI', 'BootULCI'}.
end if.
compute mtemp = mnames(1, im).
compute rlbs =
make(nrow(modvals),1,mnames(1,im)).
do if (model < 4).
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
do if (yes=0).
print outp/title = 'Conditional effect of X on Y at
values of the moderator(s):'/cnames =
clbs/format = F10.4.
else.
compute outp=outp(:,3:ncol(outp)).
compute clbs=clbs(:,3:ncol(clbs)).
compute clbs(1,1)='Contrast'.
print outp/title = 'Contrast of conditional effects
of X on Y'/cnames = clbs/format = F10.4.
end if.
end if.
do if (model > 5 and mod74dic <> 1).
do if (im = 1).
print/title = 'Conditional indirect effect(s) of X on
Y at values of the moderator(s):'.
end if.
print outp/title = 'Mediator'/rnames =
rlbs/cnames = clbs/format = F10.4.

```



```

else if (mod74dic = 1 and model = 74).
do if (im = 1).
print/title = 'Indirect effect(s) of X on Y:'.
compute clbs3=clbs(1,2:ncol(clbs)).
end if.
compute outp3=outp(1,2:ncol(outp)).
print outp3/title = 'Mediator'/rnames =
rlbs/cnames = clbs3/format = F10.4.
end if.
end loop.
loop i = notes to 1 by -1.
do if (note(i,1) = 4 and yes=0).
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 5 and yes=0).
print/title = 'Values for quantitative moderators
are the mean and plus/minus one SD from
mean.'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 14 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    below the mean was replaced
with the minimum because one SD below the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.
do if (note(i,1) = 15 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    above the mean was replaced
with the maximum because one SD above the
mean'/space=0.

```

```

print/title = '      is outside of the range of the
data.'/space=0.
end if.
end loop.
do if (model = 3 and yes=0).
compute jnvals=make(nrow(matw),7,0).
compute jnvals(:,1)=matw.
compute jnvals(:,2)=jnb1+jnb3*jnvals(:,1).
compute
jnvals(:,3)=sqrt(jnsb1+2*jnvals(:,1)*jnsb1b3+(jn
vals(:,1)*jnvals(:,1))*jnsb3).
compute jnvals(:,4)=jnvals(:,2)/jnvals(:,3).
do if (dichy = 0).
compute jnvals(:,5)=2*(1-tcdf(abs(jnvals(:,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(:,5)=2*(1-
cdfnorm(abs(jnvals(:,4)))).
end if.
compute jnvals(:,6)=jnvals(:,2)-
sqrt(jncrit)*jnvals(:,3).
compute
jnvals(:,7)=jnvals(:,2)+sqrt(jncrit)*jnvals(:,3).
compute clbs={clbs(:,1),clbs(:,3:ncol(clbs))}.
do if (coeffci = 0).
compute jnvals =jnvals(:,1:(ncol(jnvals)-2)).
end if.
print jnvals/title = 'Conditional effect of X*M
interaction at values of W: '/cnames
=clbs/format = F10.4.
end if.
do if (jn = 1 and (model = 1 or model = 3) and
jndich = 0 and yes=0).
print/title = '***** JOHNSON-
NEYMAN TECHNIQUE
*****'.
compute ajn =(jncrit*jnsb3)-(jnb3*jnb3).
compute bjn = 2*((jncrit*jnsb1b3)-(jnb1*jnb3)).
compute cjn = (jncrit*jnsb1)-(jnb1*jnb1).

```

```

compute radarg = (bjn*bjn)-(4*ajn*cjn).
compute den = 2*ajn.
compute nrts = 0.
do if (radarg >= 0 and den <> 0).
compute x21 = (-bjn+sqrt(radarg))/den.
compute x22 = (-bjn-sqrt(radarg))/den.
compute roots = 0.
do if (x21 >= jnmin and x21 <= jnmax).
compute nrts = 1.
compute roots = {roots; x21}.
end if.
do if (x22 >= jnmin and x22 <= jnmax).
compute nrts = nrts + 1.
compute roots = {roots; x22}.
end if.
compute roots={roots,make(nrow(roots),2,0)}.
compute modtemp=m.
do if (model=3).
compute modtemp=w.
end if.
do if (nrts > 0).
compute roots = roots(2:nrow(roots),1:3).
compute roots(1,2)=(csum(modtemp <
roots(1,1))/n)*100.
compute roots(1,3)=(csum(modtemp >
roots(1,1))/n)*100.
do if (nrow(roots)=2).
compute roots(2,2)=(csum(modtemp <
roots(2,1))/n)*100.
compute roots(2,3)=(csum(modtemp >
roots(2,1))/n)*100.
end if.
print roots/title = 'Moderator value(s) defining
Johnson-Neyman significance
region(s)'/clabels = 'Value', '% below', '%
above'/format = F10.4.
compute jnvals=make((21+nrts),7,0).
loop i= 0 to 20.
compute jnvals((i+1),1)=jnmin+(i*((jnmax-
jnmin)/20)).

```

```

end loop.
loop i = 1 to nrts.
loop j = 2 to nrow(jnvals).
do if ((roots(i,1) > jnvals((j-1),1)) and (roots(i,1)
< jnvals(j,1))).
compute
jnvals((j+1):(21+i),1)=jnvals(j:(20+i),1).
compute jnvals(j,1)=roots(i,1).
end if.
end loop.
end loop.
loop i = 1 to nrow(jnvals).
compute jnvals(i,2)=jnb1+jnb3*jnvals(i,1).
compute
jnvals(i,3)=sqrt(jnsb1+2*jnvals(i,1)*jnsb1b3+(jn
vals(i,1)*jnvals(i,1))*jnsb3).
compute jnvals(i,4)=jnvals(i,2)/jnvals(i,3).
do if (dichy = 0).
compute jnvals(i,5)=2*(1-tcdf(abs(jnvals(i,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(i,5)=2*(1-
cdfnorm(abs(jnvals(i,4)))).
end if.
compute jnvals(i,6)=jnvals(i,2)-
sqrt(jncrit)*jnvals(i,3).
compute
jnvals(i,7)=jnvals(i,2)+sqrt(jncrit)*jnvals(i,3).
end loop.
do if (model = 1).
print jnvals/title = 'Conditional effect of X on Y
at values of the moderator (M)'/cnames
=jnclbs/format = F10.4.
end if.
do if (model = 3).
compute
jnclbs={jnclbs(:,1),jnclbs(:,3:ncol(jnclbs))}.

```

```

print jnvals/title = 'Conditional effect of X*M on
Y at values of the moderator (W)'/cnames
=jnclbs/format = F10.4.
end if.
end if.
do if (nrts = 0).
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
else.
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
end if.
end if.
do if (model < 4 and (plot = 1)).
compute dataplot =
make((nrow(modvals)*nrow(matx)),(ncol(modv
als)+1),0).
compute tmp = 1.
loop i = 1 to nrow(modvals).
loop j = 1 to nrow(matx).
compute dataplot(tmp,:)= {matx(j,1),
modvals(i,:)}.
compute tmp=tmp+1.
end loop.
end loop.
compute dataplot = {dataplot,
make(nrow(dataplot),(1+dichy),0)}.
compute dataplo2 = make(nrow(dataplot),1,1).
do if (model = 1).
compute dataplo2 = {dataplo2, dataplot(:,2),
dataplot(:,1), (dataplot(:,1)*dataplot(:,2))}.
end if.
do if (model = 2 or model = 3).
compute dataplo2 = {dataplo2, dataplot(:,3),
dataplot(:,1), (dataplot(:,1)*dataplot(:,3)),
dataplot(:,2), (dataplot(:,1)*dataplot(:,2))}.

```

```

do if (model = 3).
compute dataplo2 = {dataplo2,
(dataplot(:,2)*dataplot(:,3)),
(dataplot(:,1)*dataplot(:,2)*dataplot(:,3))}.
end if.
end if.
loop i = 1 to nrow(dataplot).
compute tmp=dataplo2(i,:).
do if (ncovs > 0).
compute tmp = {tmp, covmeans}.
end if.
do if (cluster > 0).
compute tmp = {tmp, cldmeans}.
end if.
compute dataplot(i,(ncol(dataplot)-
(dichy)))=tmp*coeffplt.
do if (dichy = 1).
compute
dataplot(i,(ncol(dataplot)))=exp(tmp*coeffplt)/(1
+exp(tmp*coeffplt)).
end if.
end loop.
compute clbs = {xname, modvnm, 'yhat'}.
do if (dichy = 1).
compute clbs = {xname, modvnm, 'ln(odds)',
'prob'}.
end if.
print/title =
*****
*****!

```

```

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (model = 1).
do if (dichy = 0).
print/title = 'DATA LIST FREE/Pro_Env_Strgy
Technological_Advan Financial_Performance.'
/space=1.
else if (dichy = 1).
print/title = 'DATA LIST FREE/Pro_Env_Strgy
Technological_Advan logodds prob.'
/space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH Financial_Performance BY
Technological_Advan.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH Financial_Performance BY
Pro_Env_Strgy.' /space=0.
end if.
end if.

```

```

do if (dichy = 1).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH logodds BY Technological_Advan.'
/space=0.
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH prob BY Technological_Advan.'
/space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Pro_Env_Strgy.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Pro_Env_Strgy.' /space=0.
end if.
end if.
end if.
do if (model = 2 or model = 3).
do if (dichy = 0).
print/title = 'DATA LIST FREE/Pro_Env_Strgy
Technological_Advan Financial_Performance.'
/space=1.
end if.
do if (dichy = 1).
print/title = 'DATA LIST FREE/Pro_Env_Strgy
Technological_Advan logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH Financial_Performance BY

```



```

Technological_Advan/PANEL ROWVAR='
/space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH Financial_Performance BY
Pro_Env_Strgy/PANEL ROWVAR=' /space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH logodds BY
Technological_Advan/PANEL ROWVAR='
/space=0.
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH prob BY Technological_Advan/PANEL
ROWVAR=' /space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Pro_Env_Strgy/PANEL
ROWVAR=' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Pro_Env_Strgy/PANEL
ROWVAR=' /space=0.
end if.
end if.
end if.
do if (ncovs > 0).
print/title = '* Estimates are based on setting
covariates to their sample means.'.
end if.
end if.
end if.

```

```

do if ((imm > 0) or (model = 74 and
mod74dic=0) or ((model = 58 or model = 59)
and wvdich=1))).
do if (imm > 4 and imm < 8).
compute
indbootp={indbootp(:,1:nmeds),indboop1,indbo
op2}.
end if.
compute obsprod = t(indbootp(1,:)).
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indbootp(1,:).
compute indbootp = indbootp(2:(boot+1),:).
compute mnindp = t(csum(indbootp)/boot).
compute stdindp=t(sqrt((cssq(indbootp)-
((csum(indbootp)**2)/boot))/(boot-1))).
compute llcip=make(1,ncol(indbootp),-999).
compute ulcip=make(1,ncol(indbootp),-999).
loop #e = 1 to ncol(indbootp).

.
compute temp = indbootp(:,#e).
compute temp(GRADE( indbootp(:,#e) )) =
indbootp(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).

```

```

compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcip(1,#e)=llcit.
compute ulcip(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {obsprod, stdindp, t(llcip),
t(ulcip)}.
compute clbs = {'Effect', 'SE(Boot)', 'BootLLCI',
'BootULCI'}.
do if (model = 8 or model = 12).

```

```

print/title='-----'.
compute outpimm=outp(1:nmeds,:).
print/title = 'Indirect effect of highest order
product.'/space=0.
print outpimm/title = 'Mediator'/cnames =
clbs/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
compute clbs3 = {'Index', 'SE(Boot)',
'BootLLCI', 'BootULCI'}.
do if (imm < 5).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator.'/format=A8.
end if.
compute outpimm=outp((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
do if (imm > 4 and imm < 8).

```

```

print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'.

compute outpimm=outp(1:nmeds,:).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
compute
outpimm=outp((nmeds+1):nrow(outp),:).
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.

loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),clbs3}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3(1,1)=modvnm(1,1).
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.

```

```

print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
end if.
do if (boot = 0).
do if (model = 8 or model = 12).
print/title='-----'.
compute obsimm=obsprod(1:nmeds,:).
print/title = 'Indirect effect of highest order
product.'/space=0.
print obsimm/title = 'Mediator'/clabels =
'Effect'/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
do if ((imm > 0 and imm < 5) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=obsprod((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
end loop.
end if.

```

```

do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****!
.
compute outpimm=obsprod(1:nmeds,1).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
compute
outpimm=obsprod((nmeds+1):nrow(obsprod),:)
.
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),'Index'}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3={modvnm(1,1),'Index'}.
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator: '/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
end if.
end if.

```

```

end if.
compute conmake=0.
compute concols = 0.
do if ((model > 3 and model < 7) and (contrast
> 0) and nmods = 0 and nmeds > 1 ).
compute concols =
((ncol(indboot)*(ncol(indboot)-1))/2).
compute indcon=make(nrow(indboot),concols,-
999).
compute conkey = { ' ', ' ', ' ', ' ' }.
compute temp=1.
compute conmake=1.
loop i = 1 to (ncol(indboot)-1).
loop j = (i+1) to (ncol(indboot)).
compute indcon(:,temp)=indboot(:,i)-
indboot(:,j).
do if (contrast = 2).
compute indcon(:,temp)=abs(indboot(:,i))-
abs(indboot(:,j)).
end if.
do if (model <= 6).
compute conkey={conkey; mnames(1,i),
'minus', mnames(1,j)}.
end if.
do if (model = 6).
compute conkey={conkey; indlbl2(i,1), 'minus',
indlbl2(j,1)}.
end if.
compute temp=temp+1.
end loop.
end loop.
end if.
do if (model = 4 or model = 5).
compute clbs = {'Effect'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obs = {csum(obs); obs}.
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.

```



```

compute rlbs = {rlbs;
cntname(1:ncol(indcon),1)}.
end if.
compute outp = obs.
compute outp2=outp.
do if (effsize = 1 and dichy = 0).
compute
pmeff(:,1)=rsum(pmeff(:,2:ncol(pmeff))).
compute
rmeff(:,1)=rsum(rmeff(:,2:ncol(rmeff))).
compute
abpseff(:,1)=rsum(abpseff(:,2:ncol(abpseff))).
compute
abcseff(:,1)=rsum(abcseff(:,2:ncol(abcseff))).
compute eff = {pmeff, rmeff, abpseff, abcseff}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute eff = {eff, r245}.
compute r245obs = {r245(1,1);r245(1,1)}.
compute kappa2ob = kappa2(1,1).
end if.
compute pmobs = t(pmeff(1,1:(nmeds+1))).
compute rmobs = t(rmeff(1,1:(nmeds+1))).
compute psobs = t(abpseff(1,1:(nmeds+1))).
compute csobs = t(abcseff(1,1:(nmeds+1))).
do if (contrast = 0).
compute outp2 = {obs, psobs, csobs, pmobs,
rmobs}.
end if.
do if (contrast > 0).
compute obs2=obs(1:nrow(psobs),:).
compute outp2 = {obs2, psobs, csobs, pmobs,
rmobs}.
end if.
compute clbs = {'ab', 'ab_ps', 'ab_cs', 'ab/c',
'ab/c'}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute outp2 = {outp2, r245obs}.
compute clbs = {clbs, 'R-sq_med'}.

```

```

do if (boot = 0 and mc = 0).
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
do if (boot = 0 and mc = 0).
do if (nmeds = 1).
compute outp2 = outp2(2,:).
compute rlbs = rlbs(2,1).
end if.
do if (model=5).
compute outp2=outp2(:,1:3).
end if.
print outp2/title = 'Indirect effect(s) of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1 and nmeds >
1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0 or mc > 0).
compute indboot = {rsum(indboot), indboot}.
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
compute ones = make(bootsz,1,1).
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(bootSZ+1),:).
compute mnind = t(csum(indboot)/bootSZ).

```

```

compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/bootSZ))/(bootSZ-1))).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).

.

compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.

```

```

end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llici(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
do if (effsize = 1 and dichy = 0 and boot > 0).
compute estmte=eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).
.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.

```

```

compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).

```

```

compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
end if.
end if.
do if (boot > 0 or mc > 0).
compute outp = {obs, stdind, t(llci), t(ulci)}.
do if (nmeds = 1).
compute outp = outp(2,:).
compute rlbs = rlbs(2,1).
end if.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
do if (mc > 0).
compute clbs = {'Effect', 'MC SE', 'MC LLCI',
'MC ULCI'}.
end if.
print outp/title = 'Indirect effect of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (dichy = 0 and effsize = 1 and boot > 0).
compute outp = {psobs,
stdindf((2*(nmeds+1)+1):(3*(nmeds+1)),1),
t(llcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))),
t(ulcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Partially standardized indirect
effect of X on Y'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute outp = {csobs,
stdindf((3*(nmeds+1)+1):(4*(nmeds+1)),1),
t(llcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))),
t(ulcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.

```

```

print outp/title = 'Completely standardized
indirect effect of X on Y'/rnames = rlbs/cnames
= clbs/format = F10.4.
compute outp = {pmobs,
stdindf(1:(nmeds+1),1),t(llcif(1,1:(nmeds+1))),
t(ulcif(1,1:(nmeds+1))) }.
do if (model = 4).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
compute outp = {rmobs, stdindf(((nmeds+1)+1)
:(2*(nmeds+1)) ,1),
t(llcif(1,((nmeds+1)+1):(2*(nmeds+1)))),
t(ulcif(1,((nmeds+1)+1):(2*(nmeds+1))))}.
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
do if (nmeds = 1 and cluster = 0 and ncovs = 0
and model = 4).
compute r245obs = r245obs(1,1).
compute outp = {r245obs,
stdindf((4*(nmeds+1)+1):(4*(nmeds+1)+1),1),
t(llcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1))),
t(ulcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1)))}.
print outp/title = 'R-squared mediation effect
size (R-sq_med)'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
end if.
do if (normal = 1).

```

```

do if (nmeds = 1).
print sobel/title = 'Normal theory tests for
indirect effect'/clabels = 'Effect', 'se', 'Z',
'p'/format = F10.4.
end if.
do if (nmeds > 1).
compute rlbs2 = rlbs(2:nrow(rlbs),1).
print sobel/title = 'Normal theory tests for
specific indirect effects'/rnames = rlbs2/clabels
= 'Effect', 'se', 'Z', 'p'/format = F10.4.
end if.
end if.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
do if (model = 6).
compute clbs = {'eff'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obsfsum=csum(obs).
compute obs = {obsfsum; obs}.
compute indlbl = indlbl(1:nrow(obs),1).
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute indlbl = {indlbl;
cntname(1:ncol(indcon),1)}.
end if.
compute obs2=obs.
do if (boot = 0).
do if (dichy = 0 and effsize = 1).
compute clbs = {'eff', 'eff_ps', 'eff_cs', 'eff/c',
'eff/c'}.
do if (ncovs = 0).

```



```

compute obs = {obs, obs/stddevy,
(obs*stddevx/stddevy), obs/ctot,
obs/mmpaths(nrow(mmpaths),1)}.
end if.
do if (ncovs > 0 or clsdmy > 0).
compute obs = {obs, obs/sdycov,
(obs*sdxcov/sdycov),
obs/(obsfsum+mmpaths(nrow(mmpaths),1)),
obs/mmpaths(nrow(mmpaths),1)}.
end if.
compute obs2=obs.
do if (contrast > 0).
compute obs2 = obs(1:(nrow(obs)-concols),:).
end if.
end if.
print obs2/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute indboot = {rsum(indboot), indboot}.
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(boot+1),:).
compute mnind = t(csum(indboot)/boot).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/boot))/(boot-1))).
compute temp = nrow(indboot).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).

```

```

loop #e = 1 to ncol(indboot).

.

compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.

```

```

do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute obs = {obs, stdind, t(llci), t(ulci)}.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
print obs/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (effsize = 1 and dichy = 0).
compute indboot=indboot(:,1:(ncol(indboot)-
concols)).
compute eff = {rsum(abpseff), abpseff,
rsum(abcseff), abcseff, rsum(pmeff), pmeff,
rsum(rmeff), rmeff}.
compute effobs = eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).
.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).

```

```

compute badlo = 0.
compute badhi = 0.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(effobs(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.

```

```

compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute temp2 = stdindf(1:ncol(indboot),1).
compute temp3 = effobs(:,1:ncol(indboot)).
compute templo = llcif(1,1:ncol(indboot)).
compute temphi = ulcif(1,1:ncol(indboot)).
compute outp = {t(temp3), temp2,t(templo),
t(temphi)}.
print outp/title = 'Partially standardized indirect
effect of X on Y'/cnames = clbs/rnames =
indlbl/format = F10.4.
compute temp2 =
stdindf((ncol(indboot)+1):(2*ncol(indboot)),1).
compute temp3 =
effobs(:,(ncol(indboot)+1):(2*ncol(indboot))).
compute templo =
llcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute temphi =
ulcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templo),
t(temphi)}.
print outp/title = 'Completely standardized
indirect effect of X on Y'/cnames = clbs/rnames
= indlbl/format = F10.4.
compute temp2 =
stdindf((2*(ncol(indboot))+1):(3*ncol(indboot)),1
).
compute temp3 =
effobs(:,(2*(ncol(indboot))+1):(3*ncol(indboot)))
.

```

```

compute templow =
  llcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute temphi =
  ulcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
  t(temphi)}.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
compute temp =
  eff(:,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temp2 =
  stdindf((3*(ncol(indboot))+1):(4*ncol(indboot)),1
  ).
compute temp3 =
  effobs(:,(3*(ncol(indboot))+1):(4*ncol(indboot)))
  .
compute templow =
  llcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temphi =
  ulcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
  t(temphi)}.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
end if.
end if.
do if (nmeds = 2).
compute effkey = {xname, '->', mnames(1,1), '-
->', yname, '', ''}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, '', ''}.
compute effkey = {indlbl(2:4,1), effkey}.
end if.
do if (nmeds = 3).
compute effkey = {xname, '->', mnames(1,1), '-
->', yname, '', '', ''}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, '',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, '',' ',' ',' '}.
compute effkey = {indbl(2:8,1), effkey}.
end if.
do if (nmeds = 4).
compute effkey = {xname, '->' , mnames(1,1), '-
>', yname, '',' ',' ',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,4), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname, '',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,4), '->', yname, '',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, '',' '}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', mnames(1,4), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,4), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', mnames(1,4), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,4), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {indlbl(2:16,1), effkey}.
end if.
print effkey/title = 'Indirect effect key'/format =
A8.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
end if.
DO IF (medte2=1).
compute
name=t(mdlnms2(2:nrow(mdlnms2),1)).
compute tmpdat=datayed(:,2:ncol(datayed)).
compute dd=tmpdat(:,1).

```



```

compute
dd={dd,tmpdat(:,(2+nmeds)),tmpdat(:,2:(1+nmeds))}.
compute stratify=0.
do if (ncovs > 0).
compute cname = cnames.
compute
dd={dd,tmpdat(:,(3+nmeds):(2+nmeds+ncovs))}.
end if.
compute method = 1.
compute samples = boot.
do if (mc > 0).
compute samples = mc.
compute method=2.
end if.
compute omnibus = 1.
compute booterr = 0.
do if (mcx < 0 or mcx > 4).
compute mcx = 1.
end if.
compute ovals = ncol(design(dd(:,1))).
compute nte = notes.
compute criterr = 0.
compute xskip = 0.
compute dumok = 0.
compute badend = 0.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute priorlo = -9999999.
compute priorhi = 9999999.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute reps = samples.
do if (contrast = 1).
compute note(nte,1) = 26.
compute nte=nte+1.

```

```

end if.
do if (effsize = 1).
compute note(nte,1) = 27.
compute nte=nte+1.
end if.
do if (normal = 1).
compute note(nte,1) = 28.
compute nte=nte+1.
end if.
do if (method = 2).
do if (omnibus = 1).
compute note(nte,1) = 25.
compute nte=nte+1.
end if.
compute samples = 0.
do if (reps < 1000).
compute reps = 1000.
end if.
end if.
do if (method = 1).
compute samples = reps.
end if.
compute temp = ncol(dd).
compute n = nrow(dd).
compute ones = make(n,1,1).
compute temp=dd(:,2).
compute nxd = ncol(temp).
compute nx = nxd.
do if (nx = 1 and mcx > 0).
compute temp = dd.
compute temp(GRADE(dd(:,2)),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,2)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute mnvls = cmin(t(nnvls)).
do if (mnvls < 2).
compute errs = errs + 1.
compute runerrs(errs,1) = 37.
compute criterr = 1.

```

```

end if.
do if (nvls > 9).
compute errs = errs+1.
compute runerrs(errs,1) = 36.
compute criterr = 1.
end if.
do if (criterr = 0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,2).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,2) <> nnvls((temp-1),1)).
compute nnvls(temp,1)=dd(i,2).
compute temp = temp+1.
end if.
end loop.
do if (mcx > 0).
compute x = dummy(:,2:ncol(dummy)).
compute nx = ncol(x).
compute minus1 = make(1,ncol(x),-1).
compute xdes=make((nx+1),3,0).
compute xdes(1,1)=dd(1,2).
compute xdes(1,2)=1.
compute temp = 2.
loop k = 2 to n.
do if (dd(k,2) <> dd((k-1),2)).
compute xdes(temp,2) = k.
compute xdes(temp,1) = dd(k,2).
compute xdes((temp-1),3) = k-1.
compute temp=temp+1.
end if.
end loop.
compute xdes((temp-1),3)=n.
compute xdes = {xdes, (xdes(:,3)-xdes(:,2)+1)}.
do if (stratify <> 0).
compute note(nte,1) = 7.
compute nte=nte+1.
end if.
do if (mcx = 4).

```

```

loop k = 1 to n.
do if (rsum(x(k,:)) = 0).
compute x(k,:) = minus1.
end if.
end loop.
end if.
do if (mcx = 2 or mcx = 3).
compute note(nte,1)=4.
compute nte = nte+1.
loop k = 1 to n.
do if (rsum(x(k,:)) > 0).
loop i = 1 to ncol(x).
do if (x(k,i) = 0).
compute x(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcx = 3).
compute conmat1={-8,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1; 0,0,0,0,0,-
3,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.
loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,:)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute x(k,:)=conmat1((rsum(x(k,:))+1),:).
end loop.
end if.
end if.
compute xdata = x.
compute xskip = 1.
compute xname = ddd(1,1:nx).
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).

```

```

compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcx = 4).
compute dummat(1,:) = minus1.
end if.
do if (mcx = 2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcx = 3).
compute dummat=conmat1.
end if.
compute dummat={nnvls, dummat}.
compute xname2={name(1,2), xname}.
end if.
end if.
compute nm=nmeds.
compute r2b = make(nm,1,-999).
compute indeff = make((nx*nm),1,-999).
compute cov =
make((nx*nm+nm),(nx*nm+nm),0).
compute mns=make(nrow(cov),1,0).
compute ny = ncol(dd)-nm-nxd-ncovs.
compute y = dd(:,1).
compute ydata = y.
compute yname = name(1,1).
do if (xskip = 0).
compute x = dd(:,2:(1+nxd)).
compute xdata = x.
compute xname = name(1,2:(1+nxd)).
compute indlbs = t(xname).
end if.
compute m = dd(:,(2+nxd):(1+nxd+nm)).
compute mdata = m.

```

```

compute mlab = { ' M1 ='; ' M2 ='; ' M3 ='; '
M4 ='; ' M5 ='; ' M6 ='; ' M7 ='; ' M8 ='; '
M9 ='; ' M10 ='.
compute mlab = {mlab; ' M11 ='; ' M12 ='; '
M13 ='; ' M14 ='; ' M15 ='.
compute mname =
name(1,(2+nxd):(1+nxd+nm)).
do if (ncovs > 0).
compute c =
dd(:,(2+nxd+nm):(1+nxd+nm+ncovs)).
compute x = {x, c}.
compute xdata = x.
compute cdata = c.
compute xname = {xname, cname}.
end if.
compute nms={yname; t(mname);
t(xname(1,1:(ncol(xname)-ncovs)))}.
do if (dumok = 1).
compute nms={yname; t(mname); name(1,2)}.
end if.
do if (nm < 16).
do if (dumok = 1).
print dummat/title = 'Coding of categorical X
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
compute
boots=make((samples+1),((nx+1)*nm),-999).
do if (errs = 0 and criterr = 0).
loop k = 1 to (samples+1).
do if (k = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef=0.
do if (k > 1).
compute u=trunc(uniform(n,1)*n)+1.
compute xdata = x(u,:).
compute mdata = m(u,:).

```

```

compute ydata = y(u,:).
do if (ncovs > 0).
compute cdata = c(u,:).
end if.
end if.
loop i = 1 to nm.
compute start = 1+(i-1)*nx.
compute xmat = {ones, xdata}.
do if (covmy=2).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = mdata(:,i).
compute amat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*amat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-(((1-r2)*(n-1))/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute r2cha = r2.
compute adjr2cha = adjr2.
do if (ncovs > 0 and covmy <> 2).
compute xmat = {ones, cdata}.
compute atmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*atmat)**2).
compute r2step1 = (sstotal-ssresid2)/sstotal.
compute r2cha = r2-r2step1.
compute adjr2cha = adjr2- (1-(((1-r2step1)*(n-
1))/(n-ncol(xmat)))).
compute xmat = {ones, xdata}.
end if.
compute bootcoef={bootcoef,t(amat)}.
compute r2b(i,1)=adjr2cha.
do if (k = 1).

```

```

compute hcinvtX=inv(t( xmat )' xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,1)*hcinvtX*t( xmat
(i3,1)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtX*t( xmat )' xmat
*hcinvXtX
.
compute tratio = amat&/sqrt(diag(covmat)).
compute rnms = mname(1,i).
print rnms/title =
*****
*****'/rlabels = 'Outcome:'format
= A8.
compute lmat = ident(nrow(amat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*amat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*amat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'format = F10.4
/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'.

```



```

compute op = {amat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*(xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute cov(start:(start+(nx-
1)),start:(start+(nx-
1)))=covmat(2:(ncol(covmat)-(ncovs*(covmy <>
2))),2:(ncol(covmat)-(ncovs*(covmy <> 2)))).
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns(start:(start+(nx-
1)),1)=amat(2:(1+nx)).
end loop.
compute xmat = {ones,mdata,xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (k = 1).
print yname/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.

```

```

do if (ovals = 2).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:'/cnames = nmsd/format = F9.2.
end if.
end if.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(yamat)/n)).
compute LL3 = yamat*&ln(pt2)+(1-yamat)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= yamat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).

```

```

compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute bootcoef={bootcoef,t(bmat)}.
compute LL1 = LL2.
compute covmat=varb.
end if.
do if (ovals <> 2).
compute bmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*bmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-(((1-r2)*(n-1))/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute bootcoef={bootcoef,t(bmat)}.
end if.
do if (k = 1).
do if (ovals = 2).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.

```

```

compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).
compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms2/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute
opdirsum=outp((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
end if.
do if (ovals <> 2).
.
compute hcinvtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).

```

```

end loop.
loop i3=1 to k3.
compute xmat(:,i3) = ( resid(:,ncol( resid
))&/(1-h))&* xmat(:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat(:,i3) = sqrt( mse )&* xmat(:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(bmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/(ncol(xmat)-1).
compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat((2+nmeds):(2+nmeds+(nx-1)),:)=
lmat2.
compute fratiodr =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/nx.
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute df2=(n-ncol(xmat)).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), df2, pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = bmat&/sqrt(diag(covmat)).
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute op = {bmat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat)))) }.

```

```

compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp))))*(xp2/(temp-
(2/3)+(.11/temp)))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms2/format =
F10.4.
compute
opdirsum=op((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
compute xmat = {ones,mdata}.
do if (ncovs > 0).
compute xmat = {xmat, cdata}.
end if.
compute btmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*btmat)**2).
compute r2direct = r2-((sstotal-
ssresid2)/sstotal).
compute pfr = 1-fcdf(fratiodr,(nx),(n-1-nx-nm-
ncovs)).
compute r2direct = {r2direct, fratiodr, nx, (n-1-
nx-nm-ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms2/cnames=rnms2/form
at= F10.4.
end if.
end if.

```

```

do if (k = 1 and toteff = 1).
print yname/title = '*****'
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:.'/format = A8.
compute xmat = {ones, xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(ymat)/n)).
compute LL3 = ymat*&ln(pt2)+(1-ymat*)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= ymat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.

```

```

compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute LL1 = LL2.
compute covmat=varb.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(bmat)-
1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms = {'Constant'; t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).

```



```

compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute optotsum=outp(2:(1+nx),:).
end if.
do if (ovals <> 2).
compute cmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*cmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).

.

compute hcinvXtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).

```

```

loop i3=1 to k3.
compute xmat(:,i3) = sqrt( mse )&* xmat(:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(cmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = cmatrix/sqrt(diag(covmat)).
compute op = {cmatrix, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute optotsum=op(2:(1+nx),:).
compute ssresid2 = ssresid.
compute r2total = r2.

```

```

compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat(2:(1+nx),:) = lmat2.
compute fratio3 =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/nx.
do if (ncovs > 0).
compute xmat = {ones, cdata}.
compute ctmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*ctmat)**2).
compute r2total = r2-((sstotal-ssresid2)/sstotal).
end if.
compute pfr = 1-fcdf(fratio3,(nx),(n-1-nx-
ncovs)).
compute r2tot2 = {r2total, fratio3, nx, (n-1-nx-
ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns((nrow(mns)-
nm+1):nrow(mns),1)=bmat(2:(1+nm),1).
compute tmp = 1.
loop i = 1 to (nm*nx).
compute
boots(k,i)=mns(i,1)*mns(((nm*nx)+tmp),1).
do if ((i/nx)=trunc(i/nx)).
compute
boots(k,((nm*nx)+tmp))=r2b(tmp,1)*mns(((nm*
nx)+tmp),1).
compute tmp = tmp+1.
end if.
end loop.
do if (savboot = 1 and k > 1).

```

```

compute bootstrp((k-1),:)=bootcoef.
end if.
end loop.
do if (method = 1).
compute llci=make(1,ncol(boots),-999).
compute ulci=make(1,ncol(boots),-999).
compute indeff = t(boots(1,1:(nm*nx))).
compute indeff22=t(boots(1,:)).
compute omni = t(boots(1,((nm*nx)+1) :
ncol(boots))).
do if (nrow(boots) > 1).
compute boots = boots(2:nrow(boots),:).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
loop i = 1 to ncol(boots).
.
compute temp = boots(:,i).
compute temp(GRADE( boots(:,i) )) = boots(:,i).
compute badlo = 0.
compute badhi = 0.
do if ( indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(indeff22(i,1)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2)))+1.

```

```

do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,i)=llcit.
compute ulci(1,i)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute llci = t(llci).
compute ulci= t(ulci).
compute llci={llci,ulci}.
end if.
end if.
do if (method = 2).
compute tmp=1.
loop i = 1 to (nm*nx).
compute
indeff(i,1)=mns(i,1)&*mns(((nm*nx)+tmp),1).

```

```

do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = sqrt(-
2*ln(uniform(reps,nrow(cov))))&*cos((2*3.1415
9265358979)*uniform(reps,nrow(cov))).
compute boots=boots*chol(cov).
loop i = 1 to ncol(boots).
compute boots(:,i)=boots(:,i)+mns(i,1).
end loop.
compute tmp=1.
loop i = 1 to (nm*nx).
compute
boots(:,i)=boots(:,i)&*boots(:,((nm*nx)+tmp)).
compute temp = boots(:,i).
compute temp(GRADE(boots(:,i))) = boots(:,i).
compute boots(:,i) = temp.
do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = boots(:,1:(nm*nx)).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
compute llci =
t(((boots(cilow,:));(boots(cihigh,:)))).
end if.
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
compute rnmssum=rnms(2:(1+nx),1).
do if (ovals <> 2).
print optotsum/title = 'Relative total effects of X
of Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.

```

```

print r2tot2/title = 'Omnibus test of total effect of
X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print optotsum/title = 'Relative total effects of X
on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
end if.
compute
rnmssum=rnms2((2+nmeds):(1+nmeds+nx),1).
do if (ovals <> 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.
print r2direct/title = 'Omnibus test of direct
effect of X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
do if (omnibus = 1 and (method = 1 or method
= 3)).
compute indlbs = {indlbs; 'Omnibus'}.
end if.
loop i = 1 to nm.
compute op = mname(1,i).
print op/title = 'Relative indirect effect(s) of X on
Y through:.'/format = A8/space=0.
do if (method = 2).
compute clbs = {'Effect', 'SE MC', 'MC LLCI',
'MC ULCI'}.
compute op = {indeff, bootse(1:(nm*nx)), lici}.
end if.

```

```

do if (method = 1).
compute clbs = {'Effect', 'SE(boot)', 'LLCI',
'ULCI'}.
compute op = indeff.
do if (samples > 0).
compute op = {indeff, bootse(1:(nm*nx),:),
llci(1:(nm*nx),:)}).
end if.
end if.
compute temp = op((((i-1)*nx)+1):(nx*i),:).
do if ((method = 1) and omnibus = 1).
do if (samples = 0).
compute temp={temp;omni(i,1)}.
end if.
do if (samples > 0).
compute temp = {temp; omni(i,1),
bootse(((nm*nx)+i),:), llci(((nm*nx)+i),:)}).
end if.
end if.
print temp/title = '/rnames = indlbs/cnames =
clbs/format = F10.4 /space=0.
print/title = '-----'.
end loop.
end if.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
END IF.
DO IF (mdpbe=1).
compute nm={yname,xname,mnames}.
do if (ncovs > 0).

```



```

compute
nm={yname,cnames,xname,mnames}.
end if.

compute tpx=xname.
compute x = dd(:,2:ncol(dd)).
compute tpy = yname.
compute tempy= dd(:,1).
compute n = nrow(dd).
compute nx=1.
compute criterr=0.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute ddd1 = {'int_1', 'int_2', 'int_3', 'int_4',
'int_5', 'int_6', 'int_7', 'int_8', 'int_9'}.
compute dumok = 0.
compute mcfoc=mcx.
compute mcmmod=mcm.
compute mcloc=(mcfoc>0).
compute con = make(n,1,1).
compute ncovs=ncol(x)-2.
do if (mcfoc > 0 or mcmmod > 0).
compute temp = dd.
compute temp(GRADE(dd(:,(ncol(dd)-
mcloc))),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,ncol(dd)-
mcloc)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute toosmall=rsum(nnvls < 2).
compute mnvls = cmin(t(nnvls)).
do if ((rsum(nnvls < 2)) > 0).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=37.
end if.
do if (nvls > 10).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=36.

```

```

end if.
do if (criterr=0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,ncol(dd)-mcloc).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,ncol(dd)-mcloc) <> nnvls((temp-
1),1)).
compute nnvls(temp,1)=dd(i,ncol(dd)-mcloc).
compute temp = temp+1.
end if.
end loop.
compute dummy = dummy(:,2:ncol(dummy)).
do if (mcfoc=4 or mcmmod=4).
compute minus1=make(1,ncol(dummy),-1).
loop k = 1 to n.
do if (rsum(dummy(k,:)) = 0).
compute dummy(k,:)=minus1.
end if.
end loop.
end if.
do if (mcfoc = 2 or mcfoc = 3) or (mcmmod = 2 or
mcmmod = 3)).
loop k = 1 to n.
do if (rsum(dummy(k,:)) > 0).
loop i = 1 to ncol(dummy).
do if (dummy(k,i) = 0).
compute dummy(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcfoc = 3 or mcmmod=3).
compute conmat1={-8,1,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1,1; 0,0,0,0,0,-
3,1,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.

```

```

loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,:)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute
dummy(k,:)=conmat1((rsum(dummy(k,:))+1),:).
end loop.
end if.
end if.
compute nx = ncol(dummy).
compute xname = ddd(1,1:nx).
compute xdata = dummy.
compute xname = ddd(1,1:nx).
compute xname2={nm(1,(ncol(nm)-mcloc)),
xname}.
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcfoc = 2 or mcmmod=2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcfoc = 3 or mcmmod=3).
compute dummat=conmat1.
end if.
do if (mcfoc = 4 or mcmmod = 4).
compute dummat(1,:)=minus1.
end if.
compute dummat={nnvls, dummat}.
end if.
end if.
do if (criterr=0).
compute y = dd(:,1).
compute ovals = ncol(design(dd(:,1))).

```

```

compute itprob = 0.
do if (ovals = 2).
compute omx = cmax(y(:,1)).
compute omn = cmin(y(:,1)).
compute y(:,1) = (y(:,1) = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute sstotal = csum((y-(csum(y)/n))&**2).
compute outv = t(nm(1,1)).
compute mdtr = nm(1,ncol(dd)).
compute fciv = nm(1,(ncol(dd)-1)).
do if (mcfoc <> 0 or mcmod <> 0).
compute inter=make(n,nx,0).
loop i = 1 to nx.
compute inter(:,i)=dummy(:,i)&*dd(:,ncol(dd)-
(1-mcloc)).
end loop.
do if (ncovs = 0).
compute x={con,dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
do if (ncovs > 0).
compute
x={con,dd(:,2:(1+ncovs)),dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={t(nm(1,2:(1+ncovs)));
nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
end if.
compute xmns = csum(x)/n.
compute focvals = {1,0,0}.
compute highwarn = 0.
compute lowwarn = 0.

```

```

do if (ncol(x) > (2*nx+2)).
compute covmns = {1, xmns(1,2:(ncol(x)-1-
(2*nx)))}.
compute focvals = {covmns,0,0}.
end if.
compute dfres = n-ncol(x).
do if (ovals = 2).
compute pt2 = make(n,1,(csum(y)/n)).
compute LL3 = y*<math>\ln</math>(pt2)+(1-y)*<math>\ln</math>(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(x),1,0).
compute LL1 = 0.

.

compute pt1lp= pt1.
compute xlp= x.
compute ylp= y.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*<math>\ln</math>(pt1lp)+(1-ylp)*<math>\ln</math>(1-
pt1lp).

```

```

compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
else if (ovals <> 2).
compute invXtX=inv(t(x)*x).
compute b = invXtX*t(x)*y.
compute resid = y-(x*b).
compute ssresid=csum(resid&*resid).
compute msresid=ssresid/dfres.
compute varb = msresid*invXtX.
compute
seint=sqrt(varb(nrow(varb),nrow(varb))).
compute k3 = nrow(b).
do if (hc3 = 1).
compute xhc=x.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)= xhc(i3,:)*invXtX*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).
end loop.
compute varb=(invXtX*t(xhc)*xhc*invXtX).
end if.
compute seb = sqrt(diag(varb)).
compute r2 = 1-(ssresid/sstotal).

```

```

do if (mcfoc <> 0 or mcmod <> 0).
compute ytp=y.
compute xtp=x(:,1:(ncol(x)-nx)).
compute btp = inv(t(xtp)*xtp)*t(xtp)*ytp.
compute residt = ytp-(xtp*btp).
compute ssresidt=csum(residt*residt).
compute r2noint = 1-(ssresidt/sstotal).
end if.
compute pr = ncol(x)-1.
compute lmat = ident(nrow(b)).
compute lmat = lmat(:,2:ncol(lmat)).
compute f =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
pr).
compute pf = 1-fcdf(f,pr,dfres).
compute pf = {sqrt(r2),r2,msresid,f,pr,dfres,pf}.
compute xd=abs(xp2).
end if.
do if (itprob = 0).
compute tstat = b&/seb.
do if (ovals <> 2).
compute p = 2*(1-tcdf(abs(tstat), dfres)).
else if (ovals = 2).
compute p = 2*(1-cdfnorm(abs(tstat))).
end if.
compute outp = {b,seb,tstat,p}.
compute nms = {'constant'; nms; 'interact'}.
do if (ovals <> 2).
compute tval = (dfres* (exp((dfres-
(5/6))*((xd/(dfres-(2/3)+(.11/dfres)))*(xd/(dfres-
(2/3)+(.11/dfres)))))-1)).
compute outp={outp,(b-
sqrt(tval)&*seb),(b+sqrt(tval)&*seb)}.
end if.
compute bb = tval.
do if (dumok = 1).
do if (mcfoc > 0).
print dummat/title = 'Coding of categorical X
variable for analysis:'/cnames = xname2/format
= F5.2.

```

```

else if (mcmod > 0).
print dummat/title = 'Coding of categorical M
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
print yname/title =
*****
*****'/rlabels = 'Outcome: '/format
= A8.
do if (ovals = 2).
compute nmsd = {outv, 'Analysis'}.
print rcd/title = 'Coding of binary Y for
analysis: '/cnames = nmsd/format = F10.4.
compute outp = {outp,(outp(:,1)-
sqrt(tval)&*outp(:,2)),(outp(:,1)+sqrt(tval)&*outp
(:,2))}.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
print outp/title = 'Logistic Regression
Model'/rnames = nms/clabels 'Coeff' 'se' 'Z' 'p'
'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
compute varbtmp=varb.
compute btmp=b.

```



```

compute LL2f=LL2.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=x(:,1:(ncol(x)-nx)).
compute bt1 = make(ncol(xlp),1,0).

.

compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(y-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)*&ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.

```

```

compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL2f.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(LLdiff,nx).
compute rcha={LLdiff,nx,pchi}.
end if.
do if (ovals <> 2).
print pf/title = 'Model Summary'/clabels = 'R' 'R-
sq' 'MSE' 'F' 'df1' 'df2' 'p'/format F10.4.
compute lmat=make(nrow(b),1,0).
compute lmat(nrow(lmat),1)=1.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
1).
compute
rcha=((b(nrow(b),1)/seint)*(b(nrow(b),1)/seint))*
(1-r2)/dfres.
compute rcha = {rcha, fcha, 1, dfres,
outp((pr+1),4)}.
print outp/title = 'Model'/rnames = nms/clabels
'coeff' 'se' 't' 'p' 'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
do if (mcfoc > 0 or mcmod > 0).
compute rcha=r2-r2noint.
compute lmat=make((nrow(b)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
nx).
compute pvalr2c=1-fcdf(fcha,nx,dfres).

```

```

compute rcha = {rcha, fcha, nx, dfres,pvalr2c}.
end if.
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute intkey = {'a', 'b', 'c', 'd', 'e'}.
loop i = 1 to nx.
compute intkey={intkey; ddd1(1,i), ' : ', ddd(1,i),
' X ', nm(1,ncol(dd)-(1-mcloc))}.
end loop.
compute intkey=intkey(2:nrow(intkey),:).
print intkey/title='Product terms key:.'/format =
a8.
end if.
do if (ovals <> 2).
print rcha/title = 'R-square increase due to
interaction:.'/clabels 'R2-chng' 'F' 'df1' 'df2'
'p'/format F10.4.
else if (ovals = 2).
print rcha/title = 'Likelihood ratio test for
interaction:.'/clabels 'Chi-sq' 'df' 'p'/format F10.4.
end if.
print/title =
*****
*****'.
compute mdvar = x(:,(ncol(x)-1)).
do if (mcfoc = 0 and mcmmod = 0).
compute g1 = b((ncol(x)-2),1).
compute g3 = b(ncol(x),1).
compute vg1 = varb((ncol(x)-2),(ncol(x)-2)).
compute vg3 = varb(ncol(x),ncol(x)).
compute covg1g3 = varb((ncol(x)-2), ncol(x)).
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute mdvar = x(:,(ncol(x)-(2*nx))).
end if.
compute mdmin = cmin(mdvar).
compute mdmax = cmax(mdvar).
compute fvar = x(:,(ncol(x)-2)).
compute nval = ncol(design(mdvar)).
compute fvmin = cmin(fvar).

```

```

compute fvmax = cmax(fvar).
do if (mmodval = 999).
compute mnmd = csum(mdvar)/n.
compute tmp = make(n,1,mnmd).
compute sdmd = sqrt(csum(((mdvar-
tmp)**2))/(n-1)).
compute probeval = {mnmd-sdmd; mnmd;
mnmd+sdmd}.
do if (probeval(1,1) < mdmin).
compute lowwarn = 1.
compute probeval(1,1)=mdmin.
end if.
do if (probeval(nrow(probeval),1) > mdmax).
compute highwarn = 1.
compute probeval(nrow(probeval),1)=mdmax.
end if.
do if (quantile = 1).
compute tmp = mdvar.
compute tmp(GRADE(mdvar(:,1)),:)= mdvar.
compute mdvar = tmp.
compute
probeval={mdvar(trunc(0.1*n),1);mdvar(trunc(0
.25*n),1);mdvar(trunc(0.5*n),1);mdvar(trunc(0.7
5*n),1);mdvar(trunc(0.9*n),1)}.
end if.
end if.
do if (nval = 2).
compute probeval = make(2,1,0).
compute probeval(1,1)=cmin(mdvar).
loop i = 1 to n.
do if (mdvar(i,1) <> probeval(1,1)).
compute probeval(2,1) = mdvar(i,1).
BREAK.
end if.
end loop.
end if.
do if (mmodval <> 999).
compute probeval = mmodval.
end if.
compute outp = make(nrow(probeval),7,0).

```

```

do if (mcfoc > 0 or mcmmod > 0).
compute focvals=make(1,ncol(x)+1,1).
do if (mcfoc > 0).
print/title = 'Conditional effect of X on Y at
values of the moderator:'.
print/title = ' '/space=0.
compute rnn2=mdtr.
compute matt=make(nx,6,0).
end if.
loop jj=1 to nrow(probeval).
do if (mcfoc > 0).
loop ii=1 to nx.
compute g1=b((ncol(x)-(2*nx)+ii),1).
compute g3=b((ncol(x)-nx+ii),1).
compute vg1=varb((ncol(x)-(2*nx)+ii),(ncol(x)-
(2*nx)+ii)).
compute vg3=varb((ncol(x)-nx+ii),(ncol(x)-
nx+ii)).
compute covg1g3=varb((ncol(x)-
(2*nx)+ii),(ncol(x)-nx+ii)).
compute x2 = probeval(jj,1).
compute w1 = g1+g3*x2.
compute varw1 =
vg1+(2*x2*covg1g3)+((x2*x2)*vg3).
compute sew1 = sqrt(varw1).
compute t1 = w1/sew1.
do if (ovals <> 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-tcdf(abs(t1), dfres)).
compute cnms = {'Coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)= {w1,sew1,t1,p,llci,ulci}.
else if (ovals = 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-cdfnorm(abs(t1))).
compute cnms = {'Coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)= {w1,sew1,t1,p,llci,ulci}.

```

```

end if.
end loop.
compute rnms=t(xname).
compute mdvalpr=probeval(jj,1).
print mdvalpr/title = 'Moderator
value: '/rnames=rnn2/format= F10.4 /space=0.
print matt/title='
'/cnames=cnms/rnames=rnms/format= F10.4
/space=0.
compute xprob=x(:,(ncol(x)-(2*nx)))-mdvalpr.
loop kk = 1 to nx.
compute xprob={xprob,
(xprob(:,1)&*x(:,(ncol(x)-(2*nx)+kk)))}.
end loop.
compute xprob={x(:,1:((ncol(x)-(2*nx))-
1)),xprob}.
do if (ovals <> 2).
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute residc = y-(xprob*bmultc).
compute ssresidc=csum(residc&*residc).
compute r2c = r2-(1-(ssresidc/sstotal)).
compute fcha2=(dfres*r2c)/(nx*(1-r2)).
do if (hc3 = 1).
loop kk = 1 to nx.
compute xprob={xprob, x(:,(ncol(x)-
(2*nx)+kk))}.
end loop.
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute k3 = nrow(bmultc).
compute xhc=xprob.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)=
xhc(i3,:)*inv(t(xprob)*xprob)*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).

```

```

end loop.
compute
varbc=(inv(t(xprob)*xprob)*t(xhc)*xhc*inv(t(xprob)*xprob)).
compute lmat=make((nrow(bmultc)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha2 =
(t(t(lmat)*bmultc)*inv(t(lmat)*varbc*lmat)*((t(lmat)*bmultc)))/nx).
end if.
compute pvalr2cc=1-fcdf(fcha2,nx,dfres).
compute rcha2 = {r2c, fcha2, nx,
dfres,pvalr2cc}.
print rcha2/title = 'Test of equality of conditional
means at this value of the moderator'/clabels
'R2-chng' 'F' 'df1' 'df2' 'p'/format F10.4.
else if (ovals = 2).
compute btmp=b.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=xprob.
compute bt1 = make(ncol(xlp),1,0).

.
compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.

```

```

end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&*ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*ulp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL4.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(lldiff,nx).
compute rcha2 = {LLdiff, nx,pchi}.
print rcha2/title = 'Test of equality of log odds
conditioned on this moderator value'/clabels
'Chi-sq' 'df' 'p'/format F10.4.
end if.
end if.
compute
tttt=make(nrow(dummat),1,probeval(jj,1)).
compute tttt={tttt,dummat(:,2:ncol(dummat))}.
loop kkk=1 to nx.
compute tttt={tttt,tttt(:,1)&*tttt(:,1+kkk)}.
end loop.
compute ones=make(nrow(dummat),1,1).
do if (ncol(x) > (2*nx+2)).

```



```

compute
covmnmat=make(nrow(tttt),ncol(covmns),0).
loop kkk=1 to nrow(tttt).
compute covmnmat(kkk,:)=covmns.
end loop.
compute tttt={covmnmat,tttt}.
else.
compute tttt={ones,tttt}.
end if.
compute focvals={focvals;dummat(:,1),tttt}.
do if (mcfoc > 0).
compute yhat={dummat(:,1),(tttt*b)}.
compute cmnms={fciv, 'yhat'}.
do if (ovals <> 2).
print yhat/title = 'Estimated conditional means
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (ovals = 2).
print yhat/title = 'Estimated conditional log odds
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (jj <> nrow(probeval)).
print/title='-----'/space=0.
end if.
end if.
end loop.
compute focvals=focvals(2:nrow(focvals),:).
compute yhat=focvals(:,2:ncol(focvals))*b.
compute
focvals={focvals(:,1),focvals(:,(ncol(focvals)-
(2*nx))),yhat}.
compute cnms={fciv,mdtr,'yhat'}.
do if (mcmod > 0).
compute cnms={mdtr,fciv,'yhat'}.
end if.
end if.
do if (mcmod > 0).
compute outp=make((nx+1),7,0).

```

```

compute bcatm={b((ncol(x)-
(2*nx)),1);b((ncol(x)-nx+1):ncol(x),1)}.
compute outp(:,1)=dummat(:,1).
compute bcatcov=varb((ncol(x)-
nx):ncol(x),(ncol(x)-nx):ncol(x)).
compute bcatcov(1,1)=varb((ncol(x)-
(2*nx)),(ncol(x)-(2*nx))).
compute
bcatcov(2:nrow(bcatcov),1)=varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx)).
compute
bcatcov(1,2:nrow(bcatcov))=t(varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx))).
loop i = 1 to nrow(dummat).
compute
catmval={1,dummat(i,2:ncol(dummat))}.
compute condeff=catmval*bcatm.
compute
condse=sqrt(catmval*bcatcov*t(catmval)).
compute outp(i,2:3)={condeff,condse}.
end loop.
compute outp(:,4)=outp(:,2)&/outp(:,3).
compute outp(:,5) = 2*(1-tcdf(abs(outp(:,4))),
dfres)).
compute outp(:,6) = (outp(:,2)-
sqrt(tval)&*outp(:,3)).
compute outp(:,7) =
(outp(:,2)+sqrt(tval)&*outp(:,3)).
do if (ovals <> 2).
compute cnmms = {xname2(1,1), 'coeff', 'se',
't', 'p', 'LLCI', 'ULCI'}.
end if.
do if (ovals = 2).
compute outp(:,5) = 2*(1-
cdfnorm(abs(outp(:,4)))).
compute cnmms = {xname2(1,1), 'coeff', 'se',
'Z', 'p', 'LLCI', 'ULCI'}.
end if.

```

```

print outp/title = 'Conditional Effect of Focal
Predictor in Groups Defined by the Moderator
Variable:.'/cnames=cnmms/format = F10.4.
end if.
do if (nval > 2 and (mmodval = 999) and
mcmmod = 0).
do if (quantile <> 1).
print/title = 'Moderator values are the sample
mean and plus/minus one SD from mean'.
else.
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
end if.
do if (highwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD above the mean was'.
print/title = 'replaced with the maximum
because one SD above the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
do if (lowwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD below the mean was'.
print/title = 'replaced with the minimum
because one SD below the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
end if.
do if (nval = 2 and (mmodval = 999)).
print/title = 'The moderator variable is
dichotomous'.
end if.
do if (mmodval <> 999 and mcmmod = 0 and
((mmodval < mdmin) or (mmodval > mdmax))).
print/title = 'Warning: Moderator value specified
is outside of the range of the data'.
end if.

```

```

do if (plot = 1).
do if (mcfoc = 0 and mcmmod=0).
compute cnms = {t(nms((ncol(x)-2):(ncol(x)-
1),1)), 'yhat'}.
end if.
do if (ovals = 2).
compute prob =
exp(focvals(:,3))/(1+exp(focvals(:,3))).
compute focvals = {focvals, prob}.
compute cnms = {cnms, 'prob'}.
end if.
print/title =
*****
*****!

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (ovals = 2).
do if (mcfoc=0).
print/title = 'DATA LIST
FREE/Technological_Advan Pro_Env_Strgy
logodds prob.' /space=1.
end if.
do if (mcmmod=0).

```

```

print/title = 'DATA LIST FREE/Pro_Env_Strgy
Technological_Advan logodds prob.'
/space=1.
end if.
else if (ovals <> 2).
do if (mcfoc=0).
print/title = 'DATA LIST
FREE/Technological_Advan Pro_Env_Strgy
Financial_Performance.' /space=1.
end if.
do if (mcmmod=0).
print/title = 'DATA LIST FREE/Pro_Env_Strgy
Technological_Advan Financial_Performance.'
/space=1.
end if.
end if.
print/title = 'BEGIN DATA.' /space=0.
print focvals/title = ' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (ovals <> 2 and mcmmod=0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH Financial_Performance BY
Pro_Env_Strgy.' /space=0.
end if.
do if (ovals <> 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH Financial_Performance BY
Technological_Advan.' /space=0.
end if.
do if (ovals = 2 and mcmmod=0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Pro_Env_Strgy.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Pro_Env_Strgy.' /space=0.
end if.
do if (ovals = 2 and mcfoc=0).

```

```

print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH logodds BY Technological_Advan.'
/space=0.
print/title =
'GRAPH/SCATTERPLOT=Pro_Env_Strgy
WITH prob BY Technological_Advan.'
/space=0.
end if.
do if (ncovs > 0).
print/title = 'NOTE: For data above, covariates
are set to their sample means.'.
end if.
end if.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
end if.
END IF.
end if.
do if (bad > 0).
compute note(notes,1) = 9.
compute notes = notes + 1.
end if.
print/title = '***** ANALYSIS
NOTES AND WARNINGS
*****'.
loop i = 1 to errs.
do if (runerrs(i,1) = 1).
print/title = 'ERROR: One of your declared
mediators is dichotomous. This procedure
can"t be used.'.
end if.
do if (runerrs(i,1) = 2).

```

```
print/title = 'ERROR: For model 6, this
procedure limits the number of mediators to
four.'.
end if.
do if (runerrs(i,1) = 3).
print/title = 'ERROR: For models 1, 2, and 3,
only a single variable can be listed in the M
list.'.
end if.
do if (runerrs(i,1) = 4).
print/title = 'ERROR: You requested a model
involving W but did not provide a valid W
variable name.'.
end if.
do if (runerrs(i,1) = 5).
print/title = 'ERROR: You requested a model
involving Z but did not provide a valid Z
variable name.'.
end if.
do if (runerrs(i,1) = 6).
print/title = 'ERROR: You requested a model
involving Q but did not provide a valid Q
variable name.'.
end if.
do if (runerrs(i,1) = 7).
print/title = 'ERROR: You requested a model
involving V but did not provide a valid V
variable name.'.
end if.
do if (runerrs(i,1) = 8).
print/title = 'ERROR: You specified a W
variable for a model that does not need it'.
end if.
do if (runerrs(i,1) = 9).
print/title = 'ERROR: You specified a Z variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 10).
print/title = 'ERROR: You specified a Q variable
for a model that does not need it'.
```

```

end if.
do if (runerrs(i,1) = 11).
print/title = 'ERROR: You specified a V variable
for a model that does not need it.'.
end if.
do if (runerrs(i,1) = 12).
print/title = 'ERROR: The variable specified for
W has already been assigned.'.
end if.
do if (runerrs(i,1) = 13).
print/title = 'ERROR: The variable specified for
Z has already been assigned.'.
end if.
do if (runerrs(i,1) = 14).
print/title = 'ERROR: The variable specified for
Q has already been assigned.'.
end if.
do if (runerrs(i,1) = 15).
print/title = 'ERROR: The variable specified for
V has already been assigned.'.
end if.
do if (runerrs(i,1) = 16).
print/title = 'ERROR: You did not provide a
valid Y variable name.'.
end if.
do if (runerrs(i,1) = 17).
print/title = 'ERROR: The variable specified for
Y has already been assigned.'.
end if.
do if (runerrs(i,1) = 18).
print/title = 'ERROR: Model 6 requires more
than one mediator.'.
end if.
do if (runerrs(i,1) = 19).
print/title = 'ERROR: You have not specified a
valid model number.'.
end if.
do if (runerrs(i,1) = 20).
print/title = 'ERROR: At least one and only one
variable must be listed for X.'.

```



```
end if.  
do if (runerrs(i,1) = 21).  
print/title = 'ERROR: At least one and only one  
variable must be listed for Y.'.  
end if.  
do if (runerrs(i,1) = 22).  
print/title = 'ERROR: Iteration didn"t converge  
to a solution. Interpret results with caution.'.  
end if.  
do if (runerrs(i,1) = 23).  
print/title = 'ERROR: You specified a clustering  
variable that does not exist in your variable  
list.'.  
end if.  
do if (runerrs(i,1) = 24).  
print/title = 'ERROR: You specified a clustering  
variable that has already been assigned.'.  
end if.  
do if (runerrs(i,1) = 25).  
print/title = 'ERROR: One or more of your M  
variables is not listed in the variables list.'.  
end if.  
do if (runerrs(i,1) = 26).  
print/title = 'ERROR: A maximum of 20 cluster  
units is allowed. Use multilevel modeling  
instead.'.  
end if.  
do if (runerrs(i,1) = 27).  
print/title = 'ERROR: One of the variables in  
your model is a constant.'.  
end if.  
do if (runerrs(i,1) = 28).  
print/title = 'ERROR: Dichotomous Y is not  
permitted with WS option.'.  
end if.  
do if (runerrs(i,1) = 29).  
print/title = 'ERROR: Insufficient number of  
variables in vars= list when using WS option.'.  
end if.  
do if (runerrs(i,1) = 30).
```

```

print/title = 'ERROR: Too many variables in
vars= list when using WS option. Covariates
not allowed.'.
end if.
do if (runerrs(i,1) = 31).
print/title = 'ERROR: mmodval and wmodval
can"t both be set to zero with contrast option.'.
end if.
do if (runerrs(i,1) = 32).
print/title = 'ERROR: You did not provide a
valid X variable name.'.
end if.
do if (runerrs(i,1) = 33).
print/title = 'ERROR: PROCESS requires all
variable names to be eight characters or
fewer.'.
print/title = '      Please shorten variable names
and reexecute.'/space=0.
end if.
do if (runerrs(i,1)=34).
print/title = 'ERROR: X and M cannot both be
specified as multicategorical.'.
end if.
do if (runerrs(i,1)=35).
print/title = 'ERROR: Multicategorical variable
specification available only for models 1 and
4.'.
end if.
do if (runerrs(i,1)=36).
print/title = 'ERROR: Categorical variables
cannot have more than 10 categories.'.
end if.
do if (runerrs(i,1)=37).
print/title = 'ERROR: Each group must have at
least two cases.'.
end if.
do if (runerrs(i,1)=38).
print/title = 'ERROR: Cluster option not
available with multicategorical variables.'.
end if.

```

```

do if (runerrs(i,1)=39).
print/title = 'ERROR: A maximum of 10
variables is allowed in the M= list.'.
end if.
do if (runerrs(i,1) = 41).
print/title = 'ERROR: MCM option not permitted
in model 4.'.
end if.
end loop.
do if (errs = 0).
do if (boot > 1 or mc > 0).
do if (bconoff = 1 and boot > 1).
print boot/title = 'Number of bootstrap samples
for bias corrected bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (bconoff = 0 and boot > 1).
print boot/title = 'Number of bootstrap samples
for percentile bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (mc > 1).
print mc/title = 'Number of samples for Monte
Carlo confidence intervals:'/format = F8.0.
end if.
do if (booterr = 1).
compute badend = badend(1,2:ncol(badend)).
print badend/title = 'WARNING: Bootstrap CI
endpoints below not trustworthy. Decrease
confidence or increase bootstraps'/format =
F10.4.
end if.
end if.
print conf/title = 'Level of confidence for all
confidence intervals in output:'/format = F8.2.
do if ((center = 1 or ws=1) and (ncol(centvar) >
1)).
compute centvar = centvar(1,2:ncol(centvar)).

```

```

print centvar/title = 'NOTE: The following
variables were mean centered prior to
analysis:.'/format = a8.
end if.
loop i = 1 to notes.
do if (note(i,1) = 1).
print/title = 'NOTE: Confidence level restricted
to between 50 and 99.9999%. 95% confidence
is provided in output'.
end if.
do if (note(i,1) = 2).
print/title = 'NOTE: Effect size measures for
indirect effects are not available for models
with dichotomous outcomes'.
end if.
do if (note(i,1) = 3).
print/title = 'NOTE: All standard errors for
continuous outcome models are based on the
HC3 estimator'.
end if.
do if (note(i,1) = 6).
print/title = 'NOTE: The number of bootstrap
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 7).
print/title = 'NOTE: The Johnson-Neyman
method is available only for Models 1 and 3'.
end if.
do if (note(i,1) = 8).
print/title = 'NOTE: The Johnson-Neyman
method cannot be used with a dichotomous
moderator'.
end if.
do if (note(i,1)=9).
print bad/title = 'NOTE: Some bootstrap
samples had to be replaced. The number of
such replacements was:'.
end if.
do if (note(i,1) = 11).

```

```

print nmiss/title = 'NOTE: Some cases were
deleted due to missing data. The number of
such cases was:'.
end if.
do if (note(i,1) = 12).
print /title = 'NOTE: Monte Carlo method
available only for models 4 and 5.
Bootstrapping was used instead.'.
end if.
do if (note(i,1) = 13).
print/title = 'NOTE: The number of Monte Carlo
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 19).
print/title = 'NOTE: Effect sizes not available for
within-subject analyses.'.
end if.
do if (note(i,1) = 16).
print/title = 'NOTE: Normal theory tests not
available for within-subject analyses.'.
end if.
do if (note(i,1) = 17).
print/title = 'NOTE: Monte Carlo confidence
intervals not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 18 and warnrep=0).
print/title = 'WARNING: You have requested
OLS estimation with a dichotomous criterion.'.
print/title = 'Interpret model coefficients and
inferential statistics with caution.'/space=0.
compute warnrep=1.
end if.
do if (note(i,1) = 20).
print/title = 'NOTE: Saving of bootstrap
estimates not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 22).

```

```
print/title = 'NOTE: Effect size option with
covariates requires covariates in models of M
and Y.'.
end if.
do if (note(i,1) = 23).
print/title = 'NOTE: Johnson-Neyman method
not available with multicategorical X'.
end if.
do if (note(i,1) = 24).
print/title = 'NOTE: MMODVAL option not
available with a multicategorical moderator'.
end if.
do if (note(i,1) = 25).
print/title = 'NOTE: Omnibus test for relative
indirect effects is not available with the MC
option.'.
end if.
do if (note(i,1) = 26).
print/title = 'NOTE: CONTRAST option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 27).
print/title = 'NOTE: EFFSIZE option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 28).
print/title = 'NOTE: NORMAL option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 31).
print/title = 'NOTE: The TOTAL option is not
available when Y is dichotomous.'.
end if.
do if (note(i,1) = 32).
print/title = 'NOTE: Kappa-squared is disabled
from output as of version 2.16.'.
end if.
do if (note(i,1) = 33).
print/title = 'NOTE: COVMY option ignored in
models 1, 2, and 3.'.
```

| | | | |
|-----------|----------------|-------------|-------------|
| | | end if. | |
| | | end loop. | |
| | | end if. | |
| | | end matrix. | |
| Resources | Processor Time | | 00:00:01.00 |
| | Elapsed Time | | 00:00:01.01 |

Run MATRIX procedure:

***** PROCESS Procedure for SPSS Release 2.16.3 *****

Written by Andrew F. Hayes, Ph.D. www.afhayes.com
Documentation available in Hayes (2013). www.guilford.com/p/hayes3

Model = 1
Y = Financia
X = Pro_Env_
M = Technolo

Sample size
856

Outcome: Financia

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|-----------|--------|----------|-------|
| .9098 | .8277 | .2092 | 1364.4527 | 3.0000 | 852.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|--------|-------|---------|-------|---------|--------|
| constant | -.7622 | .1293 | -5.8925 | .0000 | -1.0160 | -.5083 |
| Technolo | 1.2306 | .0487 | 25.2871 | .0000 | 1.1351 | 1.3262 |
| Pro_Env_ | .4145 | .0485 | 8.5438 | .0000 | .3193 | .5097 |
| int_1 | -.1124 | .0133 | -8.4544 | .0000 | -.1386 | -.0863 |

Product terms key:

int_1 Pro_Env_ X Technolo

R-square increase due to interaction(s):

| | R2-chng | F | df1 | df2 | p |
|-------|---------|---------|--------|----------|-------|
| int_1 | .0145 | 71.4764 | 1.0000 | 852.0000 | .0000 |

Conditional effect of X on Y at values of the moderator(s):

| Technolo | Effect | se | t | p | LLCI | ULCI |
|----------|--------|-------|---------|-------|--------|--------|
| 2.7372 | .1067 | .0244 | 4.3751 | .0000 | .0588 | .1545 |
| 3.8107 | -.0140 | .0251 | -.5599 | .5757 | -.0632 | .0352 |
| 4.8843 | -.1348 | .0327 | -4.1192 | .0000 | -.1990 | -.0705 |

Values for quantitative moderators are the mean and plus/minus one SD from mean.

Values for dichotomous moderators are the two values of the moderator.

***** JOHNSON-NEYMAN TECHNIQUE *****

Moderator value(s) defining Johnson-Neyman significance region(s)

| Value | % below | % above |
|--------|---------|---------|
| 3.2727 | 32.1262 | 67.8738 |
| 4.1563 | 71.0280 | 28.9720 |

Conditional effect of X on Y at values of the moderator (M)

| Technolo | Effect | se | t | p | LLCI | ULCI |
|----------|--------|-------|---------|-------|--------|--------|
| 1.0000 | .3020 | .0375 | 8.0619 | .0000 | .2285 | .3756 |
| 1.2000 | .2795 | .0354 | 7.8876 | .0000 | .2100 | .3491 |
| 1.4000 | .2571 | .0335 | 7.6718 | .0000 | .1913 | .3228 |
| 1.6000 | .2346 | .0317 | 7.4047 | .0000 | .1724 | .2967 |
| 1.8000 | .2121 | .0300 | 7.0754 | .0000 | .1532 | .2709 |
| 2.0000 | .1896 | .0284 | 6.6717 | .0000 | .1338 | .2454 |
| 2.2000 | .1671 | .0270 | 6.1814 | .0000 | .1140 | .2201 |
| 2.4000 | .1446 | .0258 | 5.5944 | .0000 | .0939 | .1953 |
| 2.6000 | .1221 | .0249 | 4.9056 | .0000 | .0733 | .1710 |
| 2.8000 | .0996 | .0242 | 4.1177 | .0000 | .0521 | .1471 |
| 3.0000 | .0771 | .0238 | 3.2445 | .0012 | .0305 | .1238 |
| 3.2000 | .0546 | .0236 | 2.3108 | .0211 | .0082 | .1011 |
| 3.2727 | .0465 | .0237 | 1.9628 | .0500 | .0000 | .0929 |
| 3.4000 | .0322 | .0238 | 1.3499 | .1774 | -.0146 | .0789 |
| 3.6000 | .0097 | .0243 | .3979 | .6908 | -.0380 | .0573 |
| 3.8000 | -.0128 | .0250 | -.5126 | .6084 | -.0619 | .0363 |
| 4.0000 | -.0353 | .0260 | -1.3576 | .1750 | -.0864 | .0157 |
| 4.1563 | -.0529 | .0269 | -1.9628 | .0500 | -.1058 | .0000 |
| 4.2000 | -.0578 | .0272 | -2.1229 | .0341 | -.1113 | -.0044 |
| 4.4000 | -.0803 | .0286 | -2.8034 | .0052 | -.1365 | -.0241 |
| 4.6000 | -.1028 | .0302 | -3.4008 | .0007 | -.1621 | -.0435 |
| 4.8000 | -.1253 | .0319 | -3.9212 | .0001 | -.1880 | -.0626 |
| 5.0000 | -.1478 | .0338 | -4.3724 | .0000 | -.2141 | -.0814 |

Data for visualizing conditional effect of X on Y

Paste text below into a SPSS syntax window and execute to produce plot.

DATA LIST FREE/Pro_Env_Strgy Technological_Advan Financial_Performance.
BEGIN DATA.

| | | |
|--------|--------|--------|
| 2.7685 | 2.7372 | 2.9017 |
| 3.8879 | 2.7372 | 3.0211 |
| 5.0072 | 2.7372 | 3.1405 |
| 2.7685 | 3.8107 | 3.8886 |
| 3.8879 | 3.8107 | 3.8729 |
| 5.0072 | 3.8107 | 3.8572 |
| 2.7685 | 4.8843 | 4.8755 |
| 3.8879 | 4.8843 | 4.7247 |
| 5.0072 | 4.8843 | 4.5738 |

END DATA.

GRAPH/SCATTERPLOT=Pro_Env_Strgy WITH Financial_Performance BY
Technological_Advan.

***** ANALYSIS NOTES AND WARNINGS *****

Level of confidence for all confidence intervals in output:
95.00

----- END MATRIX -----

restore.

Matrix

| Notes | | |
|----------------|--------------------------------|----------------------|
| Output Created | | 20-OCT-2020 16:06:23 |
| Comments | | |
| Input | Active Dataset | DataSet1 |
| | Filter | <none> |
| | Weight | <none> |
| | Split File | <none> |
| | N of Rows in Working Data File | 856 |

Syntax

```
matrix.  
get dat/file = */variables =  
Financial_Performance Plan_Orga_Pract  
Technological_Advan /names =  
vnames/missing = 9999.  
compute ninit = nrow(dat).  
get dat/file = */variables =  
Financial_Performance Plan_Orga_Pract  
Technological_Advan /names =  
vnames/missing = omit.  
get tmp/file = */variables =  
Financial_Performance /names =  
ynames/missing = omit.  
get tmp2/file = */variables = Plan_Orga_Pract  
/names = xnames/missing = omit.  
get tmp/file = */variables =  
Technological_Advan /names =  
mnames/missing = omit.  
get tmp/file = */variables = w999999t z999999t  
v999999t q999999t.  
compute wname=tmp(1,1).  
do if (wname = ' ').  
compute wname = 'xxx'.  
end if.  
compute zname=tmp(1,2).  
do if (zname = ' ').  
compute zname = 'xxx'.  
end if.  
compute vname=tmp(1,3).  
do if (vname = ' ').  
compute vname = 'xxx'.  
end if.  
compute qname=tmp(1,4).  
do if (qname = ' ').  
compute qname = 'xxx'.  
end if.  
compute n = nrow(dat).  
compute p0=-.322232431088.  
compute p1 = -1.  
compute p2 = -.342242088547.
```

```

compute p3 = -.0204231210245.
compute p4 = -.0000453642210148.
compute q0 = .0993484626060.
compute q1 = .588581570495.
compute q2 = .531103462366.
compute q3 = .103537752850.
compute q4 = .0038560700634.
compute badend = 0.
compute priorlo = -9999999.
compute priorhi = 9999999.
compute criterr = 0.
compute cluster = 0.
compute clsdmy = 0.
compute jndich = 0.
compute wvdich=0.
compute zqdich=0.
compute mod74dic=0.
compute warnrep=0.
compute contrast = trunc( 0 ).
do if (contrast > 2 or contrast < 0).
compute contrast = 0.
end if.
compute booterr = 0.
compute effsize = ( 0 = 1).
compute note = make(10,1,0).
compute notes = 1.
compute iterr = 0.
compute clsmth = 0.
compute quantile = ( 0 = 1).
compute mmodval= 999.
compute jn = ( 1 = 1).
compute covcoeff=( 0 =1).
compute center = ( 0 = 1).
compute detail = ( 1 = 1).
compute coeffci = ( 1 = 1).
compute conf = 95.
compute longname = ( 1 =1).
compute bconoff=( 0 <> 1).
compute covmy = trunc( 0 ).
do if (covmy < 0 or covmy > 2).

```

```

compute covmy = 0.
end if.
do if (trunc(conf) >= 100 or (trunc(conf) <=
50)).
compute conf = 95.
compute note(notes,1) = 1.
compute notes = notes + 1.
end if.
do if (n < ninit).
compute nmiss = ninit-n.
compute note(notes,1) = 11.
compute notes = notes + 1.
end if.
compute errs = 0.
compute quantd = {0,0,0,0,0,0}.
compute quantc = {0,0,0,0,0,0}.
compute mcheck = 0.
compute ttt = 0.
compute plot = trunc( 1 ).
compute runerrs = make(50,1,0).
compute model = trunc( 1 ).
compute normal = 0.
compute ws= ( 0 =1).
compute olsdichm= ( 0 =1).
compute olsdichy= ( 0 =1).
compute mcx=trunc( 0 ).
compute mcm=trunc( 0 ).
do if (ws = 1).
do if (effsize = 1).
compute note(notes,1) = 19.
compute notes = notes + 1.
compute effsize=0.
end if.
do if (normal=1).
compute note(notes,1) = 16.
compute notes = notes + 1.
compute normal=0.
end if.
end if.
do if (model <> 4).

```

```

compute ws = 0.
end if.
do if (model > 0 and model < 4 and covmy <>
0).
compute covmy=0.
compute note(notes,1) = 33.
compute notes = notes + 1.
end if.
print/space=0.
do if (jn = 1 and model <> 1 and model <> 3).
compute note(notes,1) = 7.
compute notes = notes + 1.
end if.
do if ((model > 76) or (model < 1)).
compute model = 77.
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 19.
end if.
compute itprob=0.
do if (mcx < 0).
compute mcx=0.
end if.
do if (mcx > 4).
compute mcx=0.
end if.
do if (mcm < 0).
compute mcm=0.
end if.
do if (mcm > 4).
compute mcm=0.
end if.
do if (mcx > 0 and mcm > 0).
compute errs=errs+1.
compute runerrs(errs,1)=34.
compute criterr=1.
end if.
do if ((model <> 1 and model <> 4) and (mcx >
0 or mcm > 0)).
compute errs=errs+1.

```

```

compute runerrs(errs,1)=35.
compute criterr=1.
end if.
do if ((model = 4) and (mcm > 0)).
compute errs=errs+1.
compute runerrs(errs,1)=41.
compute criterr=1.
end if.
do if ((mcx > 0) and jn <> 0).
compute jn=0.
compute note(notes,1) = 23.
compute notes = notes + 1.
end if.
do if (mmodval <> 999 and mcm > 0).
compute note(notes,1) = 24.
compute notes = notes + 1.
end if.
compute toteff = 0.
compute toteff = (( 0 = 1)*( 1 = 4 or 1 = 6)).
compute varorder = 2.
compute hc3 = ( 0 <> 0).
compute cname = 'xxx'.
compute centvar = {'xxx'}.
compute nmeds = ncol(mnames).
do if (longname=0).

```

```
compute criterr = 1.  
compute errs = errs+1.  
compute runerrs(errs,1) = 33.
```

|

|

|

end if.

compute modelm =

```
{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,2;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,3;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,5;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,6;  
1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,7;  
1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,8;  
1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,9;  
1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,3,10;
```

1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,5,11;
1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,0,5,12;
1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,5,13;
0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,2,14;
0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,2,15;
0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,4,16;
0,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,4,17;
0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,6,18;
0,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,6,19;
0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,6,20;
1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,7,21;
1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,0,7,22;
1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,23;
1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,0,0,24; 1,1,1,1

,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,25;
1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,0,0,26;
1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,0,0,27;
1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,7,28;
1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,0,7,29;
1,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,30;
1,1,0,1,0,0,1,1,0,1,0,0,0,0,0,0,0,31;
1,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,0,32;
1,1,1,1,0,0,1,1,1,1,0,0,0,0,0,0,0,33;
1,1,1,1,0,0,1,0,0,1,0,0,0,0,0,0,0,34;
1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,35;
1,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,36;
1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,37;
1,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,38;
1,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,0,39;
1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,0,40;
1,0,0,1,1,0,1,0,0,1,1,0,0,0,0,0,0,41;
1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,42;
1,0,0,1,1,1,1,0,0,1,1,1,0,0,0,0,0,43;
1,0,0,1,1,1,1,0,0,1,0,0,0,0,0,0,0,44;
1,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,45;
1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,46;
1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,47;
1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,48;
1,1,0,1,1,0,1,1,0,0,0,

```

0,0,0,0,0,0,0,0,49;
1,1,0,1,1,0,0,0,1,1,0,0,0,0,0,0,0,50;
1,1,1,1,1,0,1,1,1,0,0,0,0,0,0,0,0,51;
1,1,1,1,1,0,0,0,1,1,0,0,0,0,0,0,0,52;
1,1,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,53;
1,1,0,1,1,1,0,0,1,1,1,0,0,0,0,0,0,54;
1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,55;
1,1,1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,56;
1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,57;
1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,58;
1,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,59;
1,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,60;
1,1,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,61;
1,1,0,0,0,0,1,0,0,0,1,0,0,0,0,0,0,62;
1,1,0,0,0,1,1,0,0,0,1,0,0,0,0,0,0,63;
1,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,64;
1,0,0,1,0,0,1,0,0,0,0,1,0,0,0,0,0,65;
1,0,0,1,0,0,0,0,1,0,0,1,0,0,0,0,0,66;
1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,0,0,67;
1,1,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,68;
1,1,1,0,0,1,1,1,0,0,0,1,0,0,0,0,0,69;
1,0,0,1,0,0,0,0,0,0,0,1,1,0,0,0,0,70;
1,0,0,1,0,0,1,0,0,1,0,0,1,1,0,0,0,71;
1,1,1,0,0,0,0,0,0,0,0,1,0,0,1,1,0,72;
1,1,1,0,0,1,1,1,0,0,0,1,0,0,1,1,0,0

,73; 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,74;
1,1,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,75;
1,1,0,0,0,1,1,0,0,0,1,0,0,1,0,0,76;
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,77}.
compute wm = modelm(model, 1).
compute zm = modelm(model, 2).
compute wzm = modelm(model, 3).
compute vy = modelm(model, 4).
compute qy = modelm(model, 5).
compute vqy = modelm(model, 6).
compute wy = modelm(model, 7).
compute zy = modelm(model, 8).
compute wzy = modelm(model, 9).

```

```

compute vxy = modelm(model, 10).
compute qxy = modelm(model, 11).
compute vqxy = modelm(model, 12).
compute wmy = modelm(model, 13).
compute wvmy = modelm(model, 14).
compute wvxy = modelm(model, 15).
compute zmy = modelm(model, 16).
compute wzmy = modelm(model, 17).
compute xmy = modelm(model, 18).
compute imm=modelm(model, 19).
do if (model = 74).
compute vdich=0.
end if.
do if (ncol(xname) <> 1).
compute errs = errs+1.
compute runerrs(errs, 1) = 20.
compute criterr = 1.
end if.
do if (ncol(mnames) > 10).
compute errs = errs+1.
compute runerrs(errs, 1) = 39.
compute criterr = 1.
end if.
do if (ncol(yname) <> 1).
compute errs = errs+1.
compute runerrs(errs, 1) = 21.
compute criterr = 1.
end if.
do if (( 999 =0) and ( 999 =0) and (model=3 or
model=2) and (contrast=1)).
compute errs = errs+1.
compute runerrs(errs, 1) = 31.
compute criterr = 1.
end if.
compute xlist = (wm or zm or wzm or wy or zy
or wzy or vxy or qxy or vqxy or wvxy or xmy).
compute mlist = (vy or qy or vqy or zmy or wmy
or wzmy or xmy or (model < 4)).
compute bad=0.
do if (criterr = 0).

```

```

compute werr = 0.
compute verr = 0.
compute qerr = 0.
compute zerr = 0.
compute yerr = 1.
compute xerr = 1.
compute wlist = (wm or wzm or wy or wzy or
wm or wvmy or wvmy or wvxy or wzmy).
do if (wlist = 1 and wname = 'xxx').
compute werr = 1.
compute wlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (wlist = 1 and ((wname = qname) or
(wname = vname) or (wname = zname) or
(wname = xname) or (wname = yname))).
compute werr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 12).
end if.
compute zlist = (zm or wzm or zy or wzy or
zmy or wzmy).
do if (zlist = 1 and zname = 'xxx').
compute zerr = 1.
compute zlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (zlist = 1 and ((zname = qname) or
(zname = vname) or (zname = wname) or
(zname = xname) or (zname = yname))).
compute zerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 13.
end if.
compute qlist = (qy or vqy or qxy or vqxy).
do if (qlist = 1 and qname = 'xxx').
compute qerr = 1.
compute qlist = 0.

```

```

compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (qlist = 1 and ((qname = zname) or
(qname = vname) or (qname = wname) or
(qname = xname) or (qname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 14.
end if.
compute vlist = (vy or vqy or vxy or vqxy or
wvmy or wvxy).
do if (vlist = 1 and vname = 'xxx').
compute verr = 1.
compute vlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (vlist = 1 and ((vname = zname) or
(vname = qname) or (vname = wname) or
(vname = xname) or (vname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 15.
end if.
do if (wlist = 0 and wname <> 'xxx').
compute werr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 8.
end if.
do if (zlist = 0 and zname <> 'xxx').
compute zerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 9.
end if.
do if (qlist = 0 and qname <> 'xxx').
compute qerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 10.
end if.

```

```

do if (vlist = 0 and vname <> 'xxx').
compute verr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 11.
end if.
do if (hc3 = 1).
compute note(notes,1) = 3.
compute notes = notes+1.
end if.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute cons = make(n,1,1).
compute temp = (n*sscp(dat))-
(t(csum(dat))*(csum(dat))).
compute temp = temp/(n*(n-1)).
compute temp = csum(((diag(temp))=0)).
compute temp2=1.
do if (ws = 1).
compute temp2 = (n*sscp(tmp2))-
(t(csum(tmp2))*(csum(tmp2))).
compute temp2 = temp2/(n*(n-1)).
end if.
do if ((temp > 0 and ws = 0) or (temp > 0 and
ws = 1 and temp2 > 0)).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 27.
end if.
compute nmeds = ncol(mnames).
compute mcmats=ident(nmeds*2).
compute mccoef=make(nmeds*2,1,0).
compute sobel = make(nmeds,4,-999).
do if (model = 6 and nmeds > 4).
compute errs = errs+1.
compute runerrs(errs,1)=2.
end if.
do if (model < 4 and nmeds > 1).

```

```

compute errs = errs+1.
compute runerrs(errs,1) = 3.
end if.
compute nmods = (model = 74).
compute bad = 0.
compute intcnt = 1.
compute modvals = 0.
compute modvalsd = 0.
compute yintemp = {'int_1', 'int_2', 'int_3',
'int_4', 'int_5', 'int_6', 'int_7', 'int_8', 'int_9',
'int_10', 'int_11', 'int_12', 'int_13', 'int_14',
'int_15'}.
compute yintemp = {yintemp, 'int_16', 'int_17',
'int_18', 'int_19', 'int_20', 'int_21', 'int_22',
'int_23', 'int_24', 'int_25', 'int_26', 'int_27',
'int_28'}.
compute yintemp = {yintemp, 'int_29', 'int_30',
'int_31', 'int_32', 'int_33', 'int_34', 'int_35',
'int_36', 'int_37', 'int_38', 'int_39', 'int_40'}.
compute yintemp = {yintemp, 'int_41', 'int_42',
'int_43', 'int_44', 'int_45', 'int_46', 'int_47',
'int_48', 'int_49', 'int_50', 'int_51', 'int_52'}.
compute yintemp = {yintemp, 'int_53', 'int_54',
'int_55', 'int_56', 'int_57', 'int_58', 'int_59',
'int_60', 'int_61', 'int_62', 'int_63', 'int_64'}.
compute
cntname={'(C1)';'(C2)';'(C3)';'(C4)';'(C5)';'(C6)';(
C7)';'(C8)';'(C9)';'(C10)';'(C11)';'(C12)';'(C13)';(
C14)';'(C15)';'(C16)';'(C17)'}.
compute
cntname={cntname;'(C18)';'(C19)';'(C20)';'(C21
)';'(C22)';'(C23)';'(C24)';'(C25)';'(C26)';'(C27)';'(
C28)';'(C29)';'(C30)';'(C31)'}.
compute
cntname={cntname;'(C32)';'(C33)';'(C34)';'(C35
)';'(C36)';'(C37)';'(C38)';'(C39)';'(C40)';'(C41)';(
C42)';'(C43)';'(C44)';'(C45)'}.
compute
cntname={cntname;'(C46)';'(C47)';'(C48)';'(C49

```



```

);'(C50)';'(C51)';'(C52)';'(C53)';'(C54)';'(C55)';'(
C56)';'(C57)';'(C58)';'(C59)}.

compute
cntname={cntname;'(C60)';'(C61)';'(C62)';'(C63
)';'(C64)';'(C65)';'(C66)';'(C67)';'(C68)';'(C69)';'(
C70)';'(C71)';'(C72)';'(C73)}.

compute
cntname={cntname;'(C74)';'(C75)';'(C76)';'(C77
)';'(C78)';'(C79)';'(C80)';'(C81)';'(C82)';'(C83)';'(
C84)';'(C85)';'(C86)';'(C87)}.

compute
cntname={cntname;'(C88)';'(C89)';'(C90)';'(C91
)';'(C92)';'(C93)';'(C94)';'(C95)';'(C96)';'(C97)';'(
C98)';'(C99)';'(C100)';'(C101)}.

compute
cntname={cntname;'(C102)';'(C103)';'(C104)';'(
C105)}.

compute apathnam={'a path'; 'a1 path'; 'a2
path'; 'a3 path'; 'a4 path'; 'a5 path'; 'a6 path';
'a7 path'; 'a8 path'; 'a9 path'; 'a10 path'}.

compute bpathnam={'b path'; 'b1 path'; 'b2
path'; 'b3 path'; 'b4path'; 'b5path'; 'b6 path'; 'b7
path'; 'b8 path'; 'b9 path'; 'b10 path'}.

compute modvnm = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.

compute modvnm2 = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.

compute mlab = { ' M1 ='; ' M2 ='; ' M3 ='; '
M4 ='; ' M5 ='; ' M6 ='; ' M7 ='; ' M8 ='; '
M9 ='; ' M10 ='.

compute m = make(n,nmeds,1).
compute mmat = make(16,nmeds,0).
compute ymat = make(8,nmeds,0).
compute deco = make(10,1,0).
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
compute iterate = abs(trunc( 10000 )).
compute converge = abs( 0.00000001 ).

```

```

compute boot = abs(trunc( 5000 )).
do if (ws=1 and 0 > 0).
compute note(notes,1)=17.
compute notes=notes+1.
end if.
compute mc=(abs(trunc( 0 ))*(1-ws)).
do if (mc > 0 and model > 5).
do if (boot = 0).
compute boot=mc.
end if.
compute mc=0.
compute note(notes,1)=12.
compute notes = notes+1.
end if.
compute adjust = 0.
do if (mc > 0 and (model > 3 and model < 6)).
compute boot=0.
compute bconoff=0.
end if.
do if (boot > 0 and mc > 0).
compute mc = 0.
end if.
compute savboot = 0.
compute saveboot = ( 0 = 1).
do if (ws=1 and saveboot=1).
compute saveboot=0.
compute note(notes,1) = 20.
compute notes = notes + 1.
compute normal=0.
end if.
do if (saveboot =1 and boot > 0 and model >
3).
compute savboot = 1.
end if.
do if (boot <> 0 or mc <> 0).
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
loop.

```

```

compute cilow = rnd(bootesz*(1-(conf/100))/2).
compute cihigh =
trunc((bootesz*(conf/100)+(bootesz*(1-
(conf/100))/2)))+1.
do if (cilow < 1 or cihigh > bootesz).
compute
bootesz=trunc((bootesz+1000)/1000)*1000.
compute adjust = 1.
end if.
end loop if (cilow gt 0 and cihigh le bootesz).
compute boot=bootesz.
do if (mc > 0).
compute mc = bootesz.
end if.
do if (boot > 0 and mc > 0 and (model > 3 and
model < 6)).
compute boot=0.
end if.
do if (adjust = 1 and boot > 0).
compute note(notes,1)=6.
compute notes = notes+1.
end if.
do if (adjust = 1 and mc > 0).
compute note(notes,1)=13.
compute notes = notes+1.
end if.
end if.
do if (model = 6 and nmeds > 1).
compute mmpaths =
make((nmeds+2),(nmeds+2),0).
do if (nmeds = 2).
compute indboot = make(boot+1, 3, 999).
else if (nmeds = 3).
compute indboot = make(boot+1, 7, 999).
else if (nmeds = 4).
compute indboot = make(boot+1, 15, 999).
end if.
compute indlbl = {'Total:','Ind1 :','Ind2 :','Ind3
:','Ind4 :','Ind5 :','Ind6 :','Ind7 :','Ind8 :','Ind9

```

```

:','Ind10:','Ind11:','Ind12:','Ind13:','Ind14:','
'Ind15:'.
compute indlbl2 = {'Ind1','Ind2','Ind3','Ind4';
'Ind5','Ind6','Ind7','Ind8','Ind9','Ind10';
'Ind11','Ind12','Ind13','Ind14','Ind15'}.
compute indces = make(boot+1, 4, 999).
end if.
do if (model < 4).
compute boot = 0.
compute cmat = make(10,1,0).
compute zmat = make(10,1,0).
end if.
compute nvarch = make(1,ncol(dat),0).
compute wmatch = 0.
compute zmatch = 0.
compute vmatch = 0.
compute qmatch = 0.
compute mmatch=0.
compute minprobe=0.
compute maxprobe=0.
loop i = 1 to ncol(vnames).
do if (vnames(:,i)=yname).
compute y = dat(:,i).
compute nvarch(1,i)=1.
compute yerr = 0.
do if ((yname = xname) or (yname = wname) or
(yname = zname) or (yname = vname) or
(yname = qname)).
compute errs = errs+1.
compute runerrs(errs,1)=17.
end if.
end if.
do if (vnames(:,i)=xname).
compute x = dat(:,i).
compute nvarch(1,i)=1.
compute xdich = 1.
compute xerr=0.
loop jj = 1 to n.
do if ((x(jj,1) <> cmin(x)) and (x(jj,1) <>
cmax(x))).

```

```

compute xdich = 0.
break.
end if.
end loop.
compute xmean = csum(x)/n.
do if (center = 1 and (model < 4 or xlist > 0)
and mcx = 0).
compute meanvec = make(n,1, xmean).
compute x = x-meanvec.
compute centvar = {centvar, xname}.
end if.
compute xmean = csum(x)/n.
compute tmp = x-(cons*xmean).
compute xsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
do if (xdich = 0).
compute quantc(1,6) = 1.
compute matx = {xmean-xsd; xmean;
xmean+xsd}.
do if ( 999 = 999 and quantile = 0).
do if ((matx(1,1) < cmin(x)) and model = 74).
compute matx(1,1)=cmin(x).
compute minprobe=1.
end if.
do if ((matx(3,1) > cmax(x)) and model = 74).
compute matx(3,1)=cmax(x).
compute maxprobe=1.
end if.
end if.
do if (quantile = 1).
compute quantd(1,6) = 1.
compute quantc(1,6) = 0.
compute tmp = x.
compute tmp(GRADE(x)) = x.
compute matx =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (xdich = 1).

```

```

compute matx = {cmin(x); cmax(x)}.
do if (model=74).
compute matx=cmin(x).
compute mod74dic=1.
end if.
end if.
do if ( 999 <> 999).
compute matx = 999.
compute quantd(1,6) = 0.
compute quantc(1,6) = 0.
end if.
end if.
do if (werr = 0 and wlist = 1).
do if (vnames(:,i)=wname).
compute werr = 0.
compute wmatch = 1.
compute w = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(w)/n).
compute w = w-meanvec.
compute centvar = {centvar, wname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute wmean = csum(w)/n.
compute tmp = w-(cons*wmean).
compute wsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute wdich = 1.
loop jj = 1 to n.
do if ((w(jj,1) <> cmin(w)) and (w(jj,1) <>
cmax(w))).
compute wdich = 0.
break.
end if.
end loop.
do if (model = 3).
compute jndich=wdich.
compute jnmin=cmin(w).
compute jnmax=cmax(w).
end if.

```

```

do if (wdich = 0).
compute matw = {wmean-wsd; wmean;
wmean+wsd}.
do if ( 999 = 999 and quantile = 0).
do if (matw(1,1) < cmin(w)).
compute matw(1,1)=cmin(w).
compute minprobe=1.
end if.
do if (matw(3,1) > cmax(w)).
compute matw(3,1)=cmax(w).
compute maxprobe=1.
end if.
end if.
compute quantc(1,1) = 1.
do if (quantile = 1).
compute quantd(1,1) = 1.
compute quantc(1,1) = 0.
compute tmp = w.
compute tmp(GRADE(w)) = w.
compute matw =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (wdich = 1).
compute matw = {cmin(w); cmax(w)}.
compute wvdich=1.
compute cmaxw=cmax(w).
compute cminw=cmin(w).
end if.
do if ( 999 <> 999).
compute matw = 999.
compute quantd(1,1) = 0.
compute quantc(1,1) = 0.
end if.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm(1,1)=wname.
compute modmatp(1,1) = 1.

```

```

end if.
end if.
do if (zerr = 0 and zlist = 1).
do if (vnames(:,i)=zname).
compute zerr = 0.
compute zmatch = 1.
compute z = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(z)/n).
compute z = z-meanvec.
compute centvar = {centvar, zname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute zmean = csum(z)/n.
compute tmp = z-(cons*zmean).
compute zsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute zdich = 1.
loop jj = 1 to n.
do if ((z(jj,1) <> cmin(z)) and (z(jj,1) <>
cmax(z))).
compute zdich = 0.
break.
end if.
end loop.
do if (zdich = 0).
compute matz = {zmean-zsd; zmean;
zmean+zsd}.
do if ( 999 = 999 and quantile = 0).
do if (matz(1,1) < cmin(z)).
compute matz(1,1)=cmin(z).
compute minprobe=1.
end if.
do if (matz(3,1) > cmax(z)).
compute matz(3,1)=cmax(z).
compute maxprobe=1.
end if.
end if.
compute quantc(1,2) = 1.
do if (quantile = 1).

```



```

compute quantd(1,2) = 1.
compute quantc(1,2) = 0.
compute tmp = z.
compute tmp(GRADE(z)) = z.
compute matz = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (zdich = 1).
compute matz = {cmin(z); cmax(z)}.
compute cmaxz=cmax(z).
compute cminz=cmin(z).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matz = 999.
compute quantd(1,2) = 0.
compute quantc(1,2) = 0.
end if.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm(1,2)=zname.
compute modmatp(1,2) = 1.
end if.
end if.
do if (verr = 0 and vlist = 1).
do if (vnames(:,i)=vname).
compute verr = 0.
compute vmatch = 1.
compute v = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(v)/n).
compute v = v-meanvec.
compute centvar = {centvar, vname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute vmean = csum(v)/n.
compute tmp = v-(cons*vmean).

```

```

compute vsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute vdich = 1.
loop jj = 1 to n.
do if ((v(jj,1) <> cmin(v)) and (v(jj,1) <>
cmax(v))).
compute vdich = 0.
break.
end if.
end loop.
do if (vdich = 0).
compute matv = {vmean-vsd; vmean;
vmean+vsd}.
do if ( 999 = 999 and quantile = 0).
do if (matv(1,1) < cmin(v)).
compute matv(1,1)=cmin(v).
compute minprobe=1.
end if.
do if (matv(3,1) > cmax(v)).
compute matv(3,1)=cmax(v).
compute maxprobe=1.
end if.
end if.
compute quantc(1,3) = 1.
do if (quantile = 1).
compute quantd(1,3) = 1.
compute quantc(1,3) = 0.
compute tmp = v.
compute tmp(GRADE(v)) = v.
compute matv = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (vdich = 1).
compute matv = {cmin(v); cmax(v)}.
compute wvdich=1.
compute cmaxv=cmax(v).
compute cminv=cmin(v).
end if.
do if ( 999 <> 999).

```

```

compute matv = 999.
compute quantd(1,3) = 0.
compute quantc(1,3) = 0.
end if.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm(1,3)=vname.
compute modmatp(1,3) = 1.
end if.
end if.
do if (qerr = 0 and qlist = 1).
do if (vnames(:,i)=qname).
compute qerr = 0.
compute qmatch = 1.
compute q = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(q)/n).
compute q = q-meanvec.
compute centvar = {centvar, qname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute qmean = csum(q)/n.
compute tmp = q-(cons*qmean).
compute qsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute qdich = 1.
loop jj = 1 to n.
do if ((q(jj,1) <> cmin(q)) and (q(jj,1) <>
cmax(q))).
compute qdich = 0.
break.
end if.
end loop.
do if (qdich = 0).
compute matq = {qmean-qsd; qmean;
qmean+qsd}.
do if ( 999 = 999 and quantile = 0).
do if (matq(1,1) < cmin(q)).
compute matq(1,1)=cmin(q).
compute minprobe=1.

```

```

end if.
do if (matq(3,1) > cmax(q)).
compute matq(3,1)=cmax(q).
compute maxprobe=1.
end if.
end if.
compute quantc(1,4) = 1.
do if (quantile = 1).
compute quantd(1,4) = 1.
compute quantc(1,4) = 0.
compute tmp = q.
compute tmp(GRADE(q)) = q.
compute matq = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (qdich = 1).
compute matq = {cmin(q); cmax(q)}.
compute cmaxq=cmax(q).
compute cminq=cmin(q).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matq = 999.
compute quantd(1,4) = 0.
compute quantc(1,4) = 0.
end if.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm(1,4)=qname.
compute modmatp(1,4) = 1.
end if.
end if.
do if (vnames(:,i)=cname).
compute cld = dat(:,i).
compute cvname = vnames(:,i).
compute nvarch(1,i)=1.
compute clsmtch = 1.
end if.

```

```

loop j = 1 to ncol(mnames).
do if (vnames(:,i)=mnames(1,j)).
compute mmatch = mmatch + 1.
compute m(:,j)=dat(:,i).
do if (center = 1 and nvarch(1,i) = 0 and mlist >
0 and mcm = 0).
compute meanvec = make(n,1,csum(m(:,j))/n).
compute m(:,j) = m(:,j)-meanvec.
compute centvar = {centvar, mnames(1,j)}.
end if.
compute nvarch(1,i)=1.
compute dichm = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute dichm = 0.
break.
end if.
end loop.
do if ((dichm = 1) and (olsdichm = 1)).
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichm = 1 and model > 3 and mcheck = 0
and oldsdichm = 0).
compute errs= errs+1.
compute runerrs(errs,1) = 1.
compute mcheck = 1.
end if.
do if ((model <= 3) and (ncol(mnames)=1)).
compute nmods = nmods + 1.
compute mmean = csum(m(:,j))/n.
compute tmp = m(:,j)-(cons*mmean).
compute msd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute mdich = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute mdich = 0.
break.

```

```

end if.
end loop.
do if (model = 1).
compute jndich=mdich.
compute jnmin=cmin(m(:,j)).
compute jnmax=cmax(m(:,j)).
end if.
do if (mdich = 0).
compute matm = {mmean-msd; mmean;
mmean+msd}.
do if ( 999 = 999 and quantile = 0).
do if (matm(1,1) < cmin(m(:,j))).
compute matm(1,1)=cmin(m(:,j)).
compute minprobe=1.
end if.
do if (matm(3,1) > cmax(m(:,j))).
compute matm(3,1)=cmax(m(:,j)).
compute maxprobe=1.
end if.
end if.
compute quantc(1,5) = 1.
do if (quantile = 1).
compute quantd(1,5) = 1.
compute quantc(1,5) = 0.
compute tmp = m(:,j).
compute tmp(GRADE(m(:,j))) = m(:,j).
compute matm = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (mdich = 1).
compute matm = {cmin(m); cmax(m)}.
end if.
do if ( 999 <> 999).
compute matm = 999.
compute quantd(1,5) = 0.
compute quantc(1,5) = 0.
end if.
compute modmatv(1,5)=nrow(matm).

```

```

compute modmat((1:nrow(matm)), 5) = matm.
compute modvnm(1,5)=mnames(1,j).
compute modmatp(1,5) = 1.
end if.
end if.
end loop.
end loop.
do if (minprobe=1).
compute note(notes,1) = 14.
compute notes = notes + 1.
end if.
do if (maxprobe=1).
compute note(notes,1) = 15.
compute notes = notes + 1.
end if.
do if (cname <> 'xxx' and clsmatch = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 23.
end if.
do if (cname <> 'xxx').
do if ((cname = zname) or (cname = vname)
or (cname = wname) or (cname = xname) or
(cname = yname) or (cname = qname)).
compute errs = errs+1.
compute runerrs(errs,1) = 24.
end if.
end if.
do if (wlist = 1 and werr = 0 and wmatch = 0).
compute werr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (zlist = 1 and zerr = 0 and zmatch = 0).
compute zerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (qlist = 1 and qerr = 0 and qmatch = 0).
compute qerr = 3.
compute errs = errs+1.

```

```

compute runerrs(errs,1) = 6.
end if.
do if (vlist = 1 and verr = 0 and vmatch = 0).
compute verr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (yerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 16.
end if.
do if (xerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 32.
end if.
do if (model = 6 and nmeds < 2).
compute errs = errs+1.
compute runerrs(errs,1) = 18.
end if.
do if (mmatch < ncol(mnames)).
compute errs=errs+1.
compute runerrs(errs,1) = 25.
end if.
do if (cname <> 'xxx').
compute cld = design(cld).
compute cluster = ncol(cld).
compute cld = cld(:,2:ncol(cld)).
compute clsdmy = ncol(cld).
do if (clsdmy > 19).
compute errs = errs+1.
compute runerrs(errs,1) = 26.
end if.
do if (mcx > 0 or mcm > 0).
compute errs = errs+1.
compute runerrs(errs,1) = 38.
end if.
end if.
compute dichy = 1.
loop jj = 1 to n.

```



```

do if ((y(jj,1) <> cmin(y)) and (y(jj,1) <>
cmax(y))).
compute dichy = 0.
break.
end if.
end loop.
do if ((dichy = 1) and (olstdichy = 1)).
compute dichy=0.
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichy = 1).
do if (toteff=1).
compute toteff=0.
compute note(notes,1) = 31.
compute notes = notes + 1.
end if.
compute jncrit=xp2*xp2.
do if (ws = 1).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 28.
end if.
end if.
compute ncovs = ncol(dat)-rsum(nvarch).
do if (effsize = 1 and covmy <> 0 and ncovs >
0 and model > 3 and model < 7).
compute note(notes,1) = 22.
compute notes = notes + 1.
compute effsize = 0.
end if.
do if (ws = 1 and ncovs < nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 29.
end if.
do if (ws = 1 and ncovs > nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 30.

```

```

end if.
end if.
do if (errs = 0).
do if (rsum(quantd) > 0).
compute note(notes,1) = 4.
compute notes = notes+1.
end if.
do if (rsum(quantc) > 0).
compute note(notes,1) = 5.
compute notes = notes+1.
end if.
do if (ncovs > 0).
compute c = make(n,ncovs,0).
compute cnames = {'x'}.
compute j = 1.
loop i = 1 to ncol(vnames).
do if (nvarch(1,i)) = 0.
compute c(:,j) = dat(:,i).
compute nvarch(1,i)=1.
compute j=j+1.
compute cnames = {cnames, vnames(:,i)}.
end if.
end loop.
compute cnames = cnames(1,2:ncol(cnames)).
do if (ws = 1).
compute covmean=csum(c)/n.
loop i = 1 to ncovs.
compute meanvec = make(n,1, covmean(1,i)).
compute c(:,i) = c(:,i)-meanvec.
end loop.
compute centvar = {centvar, cnames}.
compute x=c(:,1:ncovs).
do if (ncovs=nmeds).
compute ncovs=0.
end if.
end if.
end if.
compute names = {yname, xname, mnames,
wname, zname, vname, qname}.
do if (ncovs > 0).

```

```

compute names = {names, cnames}.
end if.
do if (dichy = 1 and effsize = 1).
compute note(notes,1) = 2.
compute notes = notes+1.
end if.
do if (model > 3 and model < 6).
compute indeff=make(nmeds,1,0).
compute indboot=make(boot+1,nmeds,999).
do if (mc > 0).
compute indboot=make(mc+1,nmeds,999).
end if.
do if (effsize = 1 and dichy = 0).
compute rmeff=make(boot+1,nmeds+1,999).
compute abpseff=make(boot+1,nmeds+1,999).
compute abcseff=make(boot+1,nmeds+1,999).
compute pmeff=make(boot+1,nmeds+1,999).
compute r245 = make(boot+1,1,999).
compute kappa2 = make(boot+1,1,999).
end if.
end if.
do if (model = 6 and effsize = 1 and dichy = 0).
compute
rmeff=make(boot+1,ncol(indboot),999).
compute
abpseff=make(boot+1,ncol(indboot),999).
compute
abcseff=make(boot+1,ncol(indboot),999).
compute
pmeff=make(boot+1,ncol(indboot),999).
end if.
do if (nmods > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm(1,tmp) = modvnm(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.

```

```

end loop.
compute modmat=modmat(:,1:nmods).
compute modvnm=modvnm(:,1:nmods).
compute modmatv=modmatv(:,1:nmods).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvals =
make((modmatv(1,1)*modprod(1,1)),
nmods,0).
loop i = 1 to nmods.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvals)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvals(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
do if (model = 74).
compute modvals=matx.
compute modvnm=xname.
end if.
compute vmat = make(8,nrow(modvals),0).
compute vmat(1,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute vmat(5,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute indeff=make(nrow(modvals),1,0).
do if (model <> 5).
compute indboot = make(((boot+1)*nmeds),
nrow(modvals), -99999999).

```

```

compute indbootp =
make(boot+1,(nmeds*nmods), -99999999).
do if (imm > 4 and imm < 8).
do if (imm = 7).
compute immmv1=matw.
compute immmv2=matv.
end if.
do if (imm = 5).
compute immmv1=matw.
compute immmv2=matz.
end if.
do if (imm = 6).
compute immmv1=matv.
compute immmv2=matq.
end if.
compute
indbootp1=make((boot+1),(nmeds*(nrow(immmv2))),-99999999).
compute
indbootp2=make((boot+1),(nmeds*(nrow(immmv1))),-99999999).
compute onesmv1=make(1,nrow(immmv1),1).
compute onesmv2=make(1,nrow(immmv2),1).
compute
nimvs=nrow(immmv1)+nrow(immmv2).
compute cmmval=0.
compute cmmlbs1={'xx'}.
compute cmmlbs2={'xx'}.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv2}.
compute
cmmlbt=make(nrow(immmv2),1,mnames(1,i)).
compute cmmlbs1={cmmlbs1;cmmlbt}.
end loop.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv1}.
compute
cmmlbt=make(nrow(immmv1),1,mnames(1,i)).
compute cmmlbs2={cmmlbs2;cmmlbt}.
end loop.

```

```

compute cmmval=cmmval(2:nrow(cmmval),:).
end if.
end if.
end if.
do if (nmods> 0).
loop i = 1 to ncol(modvals).
do if (modvnm(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm(1,i)=qname).
compute qcol = i.
end if.
end loop.
end if.
do if (dichy = 1).
compute omx = cmax(y).
compute omn = cmin(y).
compute y = (y = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute data = {cons,y,m,x}.
compute datamed = data.
compute datayed = data.
compute datanm = {'constant'; yname;
t(mnames); xname}.
compute datanmm = {'constant'; yname;
t(mnames); xname}.
compute datanmy = {'constant'; yname;
t(mnames); xname}.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
do if (model < 4 and errs = 0).
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' ', ' '}.
compute datayed = {datayed, x&*m}.

```

```

compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(1,i) = 1.
compute vmat(2,i) = modvals(i,1).
end loop.
compute mmat = make(16,nmeds,1).
end if.
do if (model = 2 or model = 3).
compute int1 = x&*w.
compute datayed = {datayed, w, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,2).
compute vmat(3,i) = modvals(i,1).
compute vmat(4,i) = modvals(i,1)*modvals(i,2).
end loop.
end if.
do if (model = 3).
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*m.
compute int2 = x&*w&*m.
compute datayed = {datayed, int1, int2}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' X', wname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (model = 4 or model = 5).
compute vmat = make(8,1,1).

```

```

end if.
compute yintkey2 = yintkey.
do if (wm = 1).
compute int1 = x&*w.
compute datamed = {datamed, w,int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmm = {datanmm; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,wcol).
end loop.
do if (zm = 1).
compute int1 = x&*z.
compute datamed = {datamed, z, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(3,i) = modvals(i,zcol).
end loop.
end if.
do if (wzm = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*z.
compute int2 = x&*w&*z.
compute datamed = {datamed, int1, int2}.
loop i = 1 to nrow(modvals).

```



```

compute vmat(4,i) =
modvals(i,wcol)*modvals(i,zcol).
end loop.
end if.
end if.
compute mdatacol=ncol(datamed).
compute mintkey = yintkey.
compute yintkey = {' ',' ',' ',' ',' ',' '}.
compute medints = intcnt-1.
do if (vy = 1 or xmy = 1).
compute mp = 1.
do if (xmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,1).
end loop.
end if.
do if (vy = 1).
compute datayed = {datayed, v}.
compute datanmy = {datanmy; vname}.
compute mmods = 1.
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,vcol).
end loop.
do if (qy = 1).
compute mp = 2.
compute datayed = {datayed,q}.
compute datanmy = {datanmy; qname}.
compute mmods = 2.
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,qcol).
end loop.
end if.
do if (vqy = 1).
compute mp = 3.
compute datayed = {datayed, v&*q}.
compute mmods = 3.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,vcol)*modvals(i,qcol).
end loop.

```

```

end if.
end if.
compute mints = make(n,(nmeds*mp),0).
loop i = 0 to (nmeds-1).
do if (i = 0 and vqy = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
vname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (xmy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))&*x.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', xname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (qy = 1).
compute mints(:,((i*mp)+2))=m(:,(i+1))&*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqy = 1).
compute mints(:,((i*mp)+3))=m(:,(i+1))&*v&*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.

```

```

compute intcnt = intcnt+1.
end if.
end if.
end loop.
compute datayed = {datayed, mints}.
end if.
compute mp = 1.
do if (wvmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,wcol)*modvals(i,vcol).
end loop.
end if.
compute mints2 = make(n,(nmeds*mp),0).
do if (wmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,wcol).
end loop.
do if (wy = 0 and model > 3).
compute datayed = {datayed, w}.
compute datanmy = {datanmy; wname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wvmy = 1).
compute datayed = {datayed,w*v}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints2(:,((i*mp)+1))= m(:,(i+1))*w.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wvmy = 1).

```

```

compute mints2(:,((i*mp)+2))=
m(:,(i+1))&*w&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints2}.
end if.
compute mp = 1.
do if (zmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,zcol).
end loop.
do if (wzmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,zcol)&*modvals(i,wcol).
end loop.
end if.
end if.
do if (zmy = 1).
compute mints3 = make(n,(nmeds*mp),0).
do if (zy = 0).
compute datayed = {datayed, z}.
compute datanmy = {datanmy; zname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wzmy = 1 and wzy = 0).
compute datayed = {datayed,w&*z}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints3(:,((i*mp)+1))= m(:,(i+1))&*z.

```

```

compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wzmy = 1).
compute mints3(:,((i*mp)+2))=
m(:,(i+1))&*w&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', zname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints3}.
end if.
compute decoc = 1.
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
do if (wy = 1 and model > 3).
compute datayed = {datayed, w, x&*w}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm2(1,1)=wname.
compute modmatp(1,1) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (zy = 1).
compute datayed = {datayed,z,x&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.

```

```

compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm2(1,2)=zname.
compute modmatp(1,2) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (wzy = 1).
compute datayed = {datayed,w&*z,x&*w&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vxy = 1).
compute datayed = {datayed, x&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm2(1,3)=vname.
compute modmatp(1,3) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (qxy = 1).
compute datayed = {datayed, x&*q}.

```

```

compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm2(1,4)=qname.
compute modmatp(1,4) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqxy = 1).
compute datayed = {datayed, x&*v&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, 'X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end if.
do if (wvxy = 1).
compute datayed = {datayed, x&*w&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, 'X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute modvalsd = 0.
compute ttt = rsum(modmatp).
do if (rsum(modmatp) > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).

```

```

compute modvnm2(1,tmp) = modvnm2(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:ttt).
compute modvnm2=modvnm2(:,1:ttt).
compute modmatv=modmatv(:,1:ttt).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvalsd =
make((modmatv(1,1)*modprod(1,1)), ttt,0).
loop i = 1 to ttt.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvalsd)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvalsd(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
end if.
do if (ttt > 0).
loop i = 1 to ncol(modvalsd).
do if (modvnm2(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm2(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm2(1,i)=vname).

```



```

compute vcol = i.
end if.
do if (modvnm2(1,i)=qname).
compute qcol = i.
end if.
end loop.
compute directv = make(nrow(modvalsd),1,1).
do if (wy = 1).
compute directv ={directv, modvalsd(:,wcol)}.
end if.
do if (zy = 1).
compute directv ={directv, modvalsd(:,zcol)}.
end if.
do if (wzy = 1).
compute directv ={directv,
(modvalsd(:,wcol)&*modvalsd(:,zcol))}.
end if.
do if (vxy = 1).
compute directv ={directv, modvalsd(:,vcol)}.
end if.
do if (qxy = 1).
compute directv ={directv, modvalsd(:,qcol)}.
end if.
do if (vqxy = 1).
compute directv ={directv,
(modvalsd(:,vcol)&*modvalsd(:,qcol))}.
end if.
do if (wvxy = 1).
compute directv ={directv,
(modvalsd(:,vcol)&*modvalsd(:,wcol))}.
end if.
end if.
compute ydatacol=ncol(datayed).
do if (ncovs > 0).
do if (covmy <> 2).
compute datamed = {datamed,c}.
end if.
do if (covmy <> 1).
compute datayed = {datayed,c}.
end if.

```

```

do if (covmy = 1 and model = 4 and mcx > 0).
compute datayed = {datayed,c}.
end if.
compute covmeans = csum(c)/n.
end if.
do if (cluster > 0).
compute datamed = {datamed,cld}.
compute datayed = {datayed, cld}.
compute cldmeans = csum(cld)/n.
end if.
compute mst = 3.
compute mnd = mst+nmeds-1.
compute ydatacol=ncol(datayed).
compute mdatacol=ncol(datamed).
do if (ncovs > 0).
compute datanmy = {datanmy; t(cnames)}.
do if (model > 3).
compute datanmm = {datanmm; t(cnames)}.
end if.
end if.
compute datanmy = {'constant';
datanmy(3:nrow(datanmy),1)}.
do if (model > 3).
compute datanmm = {'constant';
datanmm(3:nrow(datanmm),1)}.
end if.
compute amm = make(2,1,0).
compute abmm = make(2,1,0).
compute mnv = csum(datayed(:,2)/n).
compute mnv = make(n,1,mnv).
compute ssty = csum((datayed(:,2)-mnv)**2).
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
do if (ws=1).
compute stddevm=diag(sigma).
compute
stddevm=sqrt(stddevm(3:(nmeds+2),1)).
end if.
compute stddevy = sqrt(sigma(2,2)).

```

```

compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute r2xy =
(sigma(2,(3+nmeds))/(stddevy*stddevx))&**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))&**2.
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).
do if (model = 4 and nmeds = 1 and cluster = 0
and ncovs = 0).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
compute datatm = datamed.
compute dataty = datayed.
compute mdlms2 = { '1' ; yname; xname}.
compute mdlms = {'Model ='; ' Y ='; ' X ='
}.

```

```

do if (ws = 1).
compute mdlInms2 = { '1' ; yname}.
compute mdlInms = {'Model ='; '   Y ='}.
end if.
loop i = 1 to ncol(mnames).
compute mdlInms2 = {mdlInms2; mnames(1,i)}.
do if (i = 1 and ncol(mnames) = 1).
compute mdlInms = {mdlInms; '   M ='}.
else.
compute mdlInms = {mdlInms; mlab(i,1)}.
end if.
end loop.
do if (wname <> 'xxx').
compute mdlInms2 = {mdlInms2; wname}.
compute mdlInms = {mdlInms; '   W ='}.
end if.
do if (zname <> 'xxx').
compute mdlInms2 = {mdlInms2; zname}.
compute mdlInms = {mdlInms; '   Z ='}.
end if.
do if (vname <> 'xxx').
compute mdlInms2 = {mdlInms2; vname}.
compute mdlInms = {mdlInms; '   V ='}.
end if.
do if (qname <> 'xxx').
compute mdlInms2 = {mdlInms2; qname}.
compute mdlInms = {mdlInms; '   Q ='}.
end if.
do if (jn = 1 and model = 1 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
do if (jn = 1 and model = 3 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
compute mdpbe=0.
compute medte2=0.
do if ( 1 =4 and mcx > 0).
compute medte2=1.

```

```

end if.
do if (((mcx > 0) or (mcm > 0)) AND model =
1).
compute mdpbe=1.
compute dd={datayed(:,4),datayed(:,3)}.
do if (ncovs > 0).
compute dd={datayed(:,6:(5+ncovs)),dd}.
end if.
compute dd={datayed(:,2),dd}.
end if.
compute yes = ((nrow(modvals)=1) and
(contrast = 1) and (model = 3 or model = 2)).
compute yes2 = ((nrow(modvals)=1) and
(contrast = 1) and (model = 2)).
print/title = '***** PROCESS Procedure
for SPSS Release 2.16.3 *****'.
print/title = '      Written by Andrew F. Hayes,
Ph.D.    www.afhayes.com'.
print/title = '  Documentation available in
Hayes (2013).
www.guilford.com/p/hayes3'/space=0.
print mdlnms2/title =
*****
*****'/rnames = mdlnms/format =
a8.
do if (ncovs > 0).
print cnames/title = 'Statistical Controls:'/rlabels
= 'CONTROL='/format a8.
end if.
print n/title = 'Sample size'/format F10.0.
do if (cluster > 0).
print cluster/rnames = cvname/title =
'Clustering Variable and Number of Clusters'.
end if.
do if (model > 3 and ( 'random' <> 'random')).
compute seedt= 'random'.
print seedt/title='Custom'/format=A12/rlabel =
'Seed:'.
end if.
do if (mdpbe=0 and medte2 = 0).

```

```

loop bt = 1 to (boot+1).
do if (bt = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)),-999).
end if.
compute bootcoef = 0.
do if (bt > 1).
loop.
compute v=trunc(uniform(n,1)*n)+1.
compute datayed = dataty(v,:).
do if (ws=1).
compute
bmeans=csum(datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed)))/n.
compute cntmat=make(n,nmeds,1).
compute bmeans2=bmeans.
do if (nmeds > 1).
compute bmeans2=mddiag(bmeans).
end if.
compute cntmat=cntmat*bmeans2.
compute datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed))=datayed(:,(ncol(data
yed)-nmeds+1):ncol(datayed))-cntmat.
end if.
compute detcheck=det(t(datayed)*datayed).
compute rk=(detcheck <> 0).
do if (model > 3).
compute datamed = datatm(v,:).
compute detcheck=det(t(datamed)*datamed).
do if (rk=1).
compute rk=(detcheck <> 0).
end if.
end if.
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
compute temp=diag(sigma).
compute bad = bad+(1-rk).
compute false = 1.
end loop if (rk = 1).

```

```

compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).
do if (model = 4 and nmeds = 1 and ncovs = 0
and cluster = 0).
compute r2xy =
(sigma(2,4)/(stddevy*stddevx))**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))**2.
compute sstot = sigma(2,2)*(n-1).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
end if.
do if (model > 3).
loop im = 1 to nmeds.

```

```

compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant';
datanmm((2+nmeds):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
do if (model = 6).
do if (im = 1).
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
end if.
do if (im > 1).
compute xm={cons, datamed(:,3:(im+1)),
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant'; datanmm(2:im,1);
datanmm((mnd):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
compute
mmpaths((im+1),(2:im))=t(coeff(2:im,1)).
end if.
end if.
do if (ws = 1).
compute
coeff(1,1)=csum(datamed(:,(2+im)))/n.
end if.
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
do if (bt = 1).
compute resid=datamed(:,(2+im))-xm*coeff.
compute sse =cssq(resid).
compute mse = sse/(n-ncol(xm)).
compute mnv = csum(data(:,(2+im)))/n.
compute mnv = make(n,1,mnv).

```



```

compute sstm = csum((data(:,(2+im))-
mnv)**2).
compute k3 = nrow(coeff).
do if (hc3 = 1).
compute h = xm(:,1).
loop i3=1 to n.
compute h(i3,1)= xm(i3,:)*invXtX*t(xm(i3,:)).
end loop.
loop i3=1 to k3.
compute xm(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xm(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xm(:,i3) = sqrt(mse)&*xm(:,i3).
end loop.
end if.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute hccov=invXtX*t(xm)*xm*invXtX.
compute mcmats(im,im)=hccov(2,2).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*hccov*lmat)*((t(lmat)
)*coeff))/dfnum.
compute coeff = coeff(1:(nrow(coeff)-
clsdmy),1).
compute mccoef(im,1)=coeff(2,1).
compute standerr =
sqrt(diag(invXtX*t(xm)*xm*invXtX)).
compute standerr = standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1).
compute standerr(1,1)=stddevm(im,1)/sqrt(n).
end if.
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-
ncol(xm)))).

```

```

compute temp=(n-ncol(xm)).
do if (ws=1).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute temp=(n-1).
end if.
compute xd = abs(xp2).
compute temp = (temp* (exp((temp-
(5/6))*((xd/(temp-(2/3)+(.11/temp)))*(xd/(temp-
(2/3)+(.11/temp)))))-1)).
compute temp1 = coeff-
sqrt(abs(temp))*standerr.
compute temp2 =
coeff+sqrt(abs(temp))*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (ws=1).
compute op=op(1,:).
compute xnmn =
apathnam(((nmeds>1)+1+(im-1)),1).
end if.
compute sobel(im,1) = coeff(2,1).
compute sobel(im,2)=standerr(2,1).
compute temp = mnames(1,im).
compute r2full = 1-(sse/sstm).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute summ = {sqrt(r2full), r2full, mse,
Fratio, dfnum, dfden, pfr}.
do if (detail = 1).
print temp/title =
*****
*****'/rlabels = 'Outcome:'/format
= A8.
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels 'R',
'R-sq', 'MSE','F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.

```

```

print op/title = 'Model'/rnames = xnmn/clabels
'coeff', 'se', 't', 'p', 'LLCI', 'ULCI'/format = F10.4.
do if (covcoeff=1 and ws <> 1).
compute
hccovtmp=hccov(1:nrow(op),1:nrow(op)).
compute cnamestp=t(xnmn).
print hccovtmp/title='Covariance matrix of
regression parameter
estimates'/rnames=xnmn/cnames=cnamestp/f
ormat= F10.4.
end if.
do if (nmods > 0 and nrow(mintkey) > 1).
print mintkey/title = 'Product terms key:.'/format
= A8.
end if.
do if (ws = 1).
compute wsd f=n-1.
print wsd f/title=' '/rlabels = 'df = '/space=0.
end if.
end if.
end if.
compute ymat(1,im) = coeff(2,1).
do if (ws = 1).
compute ymat(1,im)=coeff(1,1).
end if.
do if (wm = 1).
compute ymat(2,im) = coeff(4,1).
do if (zm = 1).
compute ymat(3,im) = coeff(6,1).
do if (wzm = 1).
compute ymat(4, im) = coeff(8,1).
end if.
end if.
end if.
do if (model = 6).
compute mmpaths((im+1),1)=coeff((im+1),1).
end if.
end loop.
end if.
loop totlp = 1 to (1+(toteff*(bt = 1))).

```

```

do if (toteff = 1 and totlp =2).
compute xy =
{cons,datayed(:,(3+nmeds):ydatacol)}.
else.
compute xy={cons, datayed(:,3:ydatacol)}.
end if.
do if (dichy = 1).
compute pt2 =
make(nrow(datayed(:,2)),1,(csum(datayed(:,2))
/n)).
compute LL3 = datayed(:,2)*ln(pt2)+(1-
datayed(:,2))*ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xy),1,0).
compute LL1 = 0.
compute xy22=xy.
loop jjj = 1 to iterate.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.
compute coeff =
bt1+inv(t(xy22)*xy)*t(xy)*(datayed(:,2)-pt1).
compute pt1 = 1/(1+exp(-(xy*coeff))).
compute itprob = csum((pt1 <
.00000000000001) or (pt1 >
.99999999999999)).
do if (itprob = 0).
compute LL = datayed(:,2)*ln(pt1)+(1-
datayed(:,2))*ln(1-pt1).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
break.
end if.
compute bt1 = coeff.
compute LL1 = LL2.
end loop.
do if (jjj >= iterate and iterr = 0).
compute errs = errs+1.

```

```

compute runerrs(errs,1) = 22.
computer iterr = 1.
end if.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)&*pt1&*(1-pt1).
end loop.
compute covmat = inv(t(xy22)*xy).
release xy22.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
end if.
do if (dichy = 0).
compute invXtX = inv(t(xy)*xy).
compute coeff =invXtX*t(xy)*datayed(:,2).
do if (effsize=1 and model > 3 and model < 7
and (ncovs > 0 or clsdmy > 0)).
compute xysdy =
{cons,datayed(:,(4+nmeds):ydatacol)}.
compute coeffsd
=inv(t(xysdy)*xysdy)*t(xysdy)*datayed(:,2).
compute resid=datayed(:,2)-(xysdy*coeffsd).
compute sdycov =sqrt(csum(resid&*resid)/(n-
ncol(xysdy))).
compute xvaron=datayed(:,(3+nmeds)).
compute
coeffx2=inv(t(xysdy)*xysdy)*t(xysdy)*xvaron.
compute residx2=xvaron-(xysdy*coeffx2).
compute sdxcov
=sqrt(csum(residx2&*residx2)/(n-ncol(xysdy))).
end if.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4 and bt > 1).

```

```

compute resid=datayed(:,2)-xy*coeff.
compute sse = cssq(resid).
compute r2full = 1-(sse/sstot).
end if.
do if (bt =1).
compute resid=data(:,2)-xy*coeff.
compute k3 = nrow(coeff).
compute sse = cssq(resid).
compute mse = sse/(n-ncol(xy)).
compute
hcnot=coeff&/(sqrt(diag(mse*invXtX))).
do if (hc3 = 1).
compute h = xy(:,1).
loop i3=1 to n.
compute h(i3,1)= xy(i3,:)*invXtX*t(xy(i3,:)).
end loop.
loop i3=1 to k3.
compute xy(:,i3) = (resid(:,ncol(resid))&/(1-
h))*xy(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xy(:,i3) = sqrt(mse)*xy(:,i3).
end loop.
end if.
compute covmat = (invXtX*t(xy)*xy*invXtX).
end if.
end if.
do if (bt = 1).
do if (model = 2).
compute xy2={cons, datayed(:,3:ydatacol)}.
compute temp = ncol(xy2).
do if (temp > 6).
compute xy3=xy2(:,7:temp)}.
end if.
compute xy2={xy2(:,1:3), xy2(:,5)}.
do if (temp > 6).
compute xy2={xy2, xy3}.
release xy3.

```

```

end if.
compute invXtX = inv(t(xy2)*xy2).
compute coeff2 = invXtX*t(xy2)*datayed(:,2).
compute ssem2 = cssq(datayed(:,2)-xy2*coeff2).
release xy2.
end if.
compute standerr = sqrt(diag(covmat)).
do if (totlp = 1).
compute
mcmats((nmeds+1):ncol(mcmats),(nmeds+1):n
col(mcmats))=covmat(2:(1+nmeds),2:(1+nmed
s)).
end if.
compute standerr=standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1 and totp=2).
compute standerr(1,1)=stddevy/sqrt(n).
end if.
compute coeffplt = coeff.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*coeff)))/dfnum).
compute coeff=coeff(1:(nrow(coeff)-clsdmy),1).
do if (totlp=1).
compute
mccoeff((nmeds+1):nrow(mccoeff))=coeff(2:(1+
nmeds),1).
end if.
compute bbbb=coeff(2,1).
do if totp = 1.
compute deco(1,1)=2+nmeds.
compute deco = deco(1:decoc,1).
compute covdirt = make((nrow(covmat)-
clsdmy),(ncol(covmat)-clsdmy),0).
compute covdirt = covmat(deco,:).

```

```

compute covdir =
make(nrow(covdirt),ncol(covdirt),0).
compute covdir = covdirt(:,t(deco)).
compute deco=coeff(deco,1).
do if (ttt > 0).
compute sedir =
sqrt(diag(directv*covdir*t(directv))).
compute directv = directv*deco.
end if.
compute sobel(:,3)=coeff(2:(1+nmeds),1).
compute sobel(:,4) = standerr(2:(1+nmeds),1).
compute sobel2 = sobel*sobel.
do if (varorder <> 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)*sobel2(:,4)+sobel2(:,3)*sobel2(:,2)).
end if.
do if (varorder = 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)*sobel2(:,4)+sobel2(:,3)*sobel2(:,2)+sobel2(:,2)*sobel2(:,4)).
end if.
compute sobel(:,1) = sobel(:,1)*sobel(:,3).
compute sobel(:,3) = sobel(:,1)/sobel(:,2).
compute sobel(:,4) = 2*(1-
cdfnorm(abs(sobel(:,3)))).
end if.
do if (dichy = 0).
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
do if (ws= 1 and totlp=2).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute dfden=n-1.
end if.
compute cnms = {'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute op = {coeff, standerr, tratio, p}.
end if.
do if (dichy = 1).
compute tratio = (coeff&/standerr).

```



```

compute p = 2*(1-cdfnorm(abs(tratio))).
compute wald = tratio&*tratio.
compute cnms = {'coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute temp=coeff-abs(xp2)*standerr.
compute op = {coeff, standerr, tratio, p, temp}.
compute temp=coeff+abs(xp2)*standerr.
compute op = {op, temp}.
end if.
do if (detail = 1).
do if (totlp = 2).
print yname/title = '*****
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:'/format = A8.
end if.
do if (totlp <> 2).
print yname/title =
'*****
*****'/rlabels = 'Outcome:'/format
= A8.
end if.
end if.
do if (dichy = 1 and bt = 1 and totp = 1).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:'/cnames = nmsd/format = F9.2.
end if.
do if (dichy = 0).
compute r2full = 1-(sse/ssty).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute jndf=dfden.
compute wsdf=dfden.
compute xd = abs(xp2).
compute jncrit = (dfden* (exp((dfden-
(5/6))*((xd/(dfden-
(2/3)+(.11/dfden)))*(xd/(dfden-
(2/3)+(.11/dfden)))))-1)).
compute summ = {sqrt(r2full), r2full, mse,
fratio, dfnum, dfden, pfr}.

```

```

compute temp1=coeff-sqrt(jncrit)*standerr.
compute temp2=coeff+sqrt(jncrit)*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (detail = 1).
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels =
'R', 'R-sq', 'MSE' 'F', 'df1', 'df2', 'p'/format =
F10.4.
end if.
end if.
end if.
do if (dichy = 1).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(coeff)-
1)).
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
do if (detail = 1).
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format =
F10.4.
end if.
end if.
do if (totlp = 2).
compute datanmy={'constant';
datanmy((nmeds+2):nrow(datanmy),1)}.
end if.
do if (detail = 1).
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
do if (ws=1).
compute op2=op(1:(1+nmeds),:).

```

```

compute datanmy2 = {'c" path'; bpathnam(
(1+(nmeds>1)): ((1+(nmeds>1))+(nmeds-
1)),1)}.
do if (totlp=2).
compute datanmy2={'c path'}.
compute op2=op(1,:).
compute wsdf=n-1.
end if.
print op2/title = 'Model'/rnames =
datanmy2/cnames = cnms/format = F10.4.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
do if (ws <> 1).
print op/title = 'Model'/rnames =
datanmy/cnames = cnms/format = F10.4.
do if (covcoeff=1).
compute
covmattp=covmat(1:nrow(op),1:nrow(op)).
compute cnamestp=t(datanmy).
print covmattp/title='Covariance matrix of
regression parameter
estimates'/rnames=datanmy/cnames=cnamest
p/format= F10.4.
end if.
end if.
end if.
do if (ttt = 0 and totp = 1).
compute deco = op((nmeds+2),:).
do if (ws = 1).
compute deco=op(1,:).
end if.
end if.
do if (ttt = 0 and totp = 2).
compute decotot = op(2,:).
do if (ws = 1).
compute decotot=op(1,:).
end if.
end if.
do if (nmods > 0 and model > 4 and detail = 1
and nrow(yintkey) > 1)).

```

```

print yintkey/title = 'Product terms key: '/format
= A8.
end if.
do if (nmods > 0 and model < 4 and detail = 1).
print yintkey2/title = 'Product terms key: '/format
= A8.
do if ((model = 1 or model = 2) and dichy = 0).
compute temp={((op(4,3)**2)*(1-r2full))/dfden,
op(4,3)**2, 1, dfden, op(4,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(4,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(2,1).
do if (model = 2).
compute temp={temp;((op(6,3)**2)*(1-
r2full))/dfden, op(6,3)**2, 1, dfden, op(6,4)}.
do if (hc3=1).
compute temp(2,1)=((hcnot(6,1)**2)*(1-
r2full))/dfden.
end if.
compute frat2=(dfden*(r2full-(1-
(ssem2/ssty))))/(2*(1-r2full)).
do if (hc3=1).
compute lmat=make(nrow(coeffplt),2,0).
compute lmat(4,1)=1.
compute lmat(6,2)=1.
compute frat2 =
(t(t(lmat)*coeffplt)*inv(t(lmat)*covmat*lmat)*((t(l
mat)*coeffplt)))/2).
end if.
compute temp={temp;(r2full-(1-
(ssem2/ssty))),frat2,2,dfden,1-
fcd(frat2,2,dfden)}.
compute rnms={rnms; yintkey2(3,1),'Both'}.
end if.
print temp/title = 'R-square increase due to
interaction(s): '/rnames=rnms/clabels = 'R2-
chng', 'F', 'df1', 'df2', 'p'/format = F10.4.
end if.

```

```

do if (model = 3 and dichy = 0).
compute temp={((op(8,3)**2)*(1-r2full))/dfden,
op(8,3)**2, dfden, op(8,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(8,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(5,1).
print temp/title = 'R-square increase due to
three-way interaction:.'/rnames=rnms/clabels =
'R2-chng', 'F(1,df2)', 'df2', 'p'/format = F10.4.
end if.
end if.
end if.
do if (model = 6 and totlp = 1).
compute
mmpaths(nrow(mmpaths),1)=coeff(nrow(mmpa
ths),1).
compute
mmpaths(nrow(mmpaths),(2:(nmeds+1)))=t(co
eff(2:(nmeds+1),1)).
end if.
do if (totlp = 1).
loop im = 1 to nmeds.
do if (model < 4).
compute ymat(1,im) = coeff(3,1)*(1-yes).
compute ymat(2,im) = coeff(4,1).
compute cmat(1,im) = covmat(3,3).
compute cmat(2,im) = covmat(4,4).
compute cmat(5,im) = covmat(3,4).
compute jnb1=coeff(3,1).
compute jnb3=coeff(4,1).
compute jnsb1=covmat(3,3).
compute jnsb3=covmat(4,4).
compute jnsb1b3=covmat(3,4).
do if (model = 2 or model = 3).
compute ymat(3,im) = coeff(6,1).
compute cmat(3,im) = covmat(6,6).
compute cmat(6,im) = covmat(3,6).
compute cmat(8,im) = covmat(4,6).

```

```

end if.
do if (model = 3).
compute ymat(4,im) = coeff(8,1).
compute cmat(4,im) = covmat(8,8).
compute cmat(7,im) = covmat(3,8).
compute cmat(9,im) = covmat(4,8).
compute cmat(10,im) = covmat(6,8).
compute jnb1=coeff(4,1).
compute jnb3=coeff(8,1).
compute jnsb1=covmat(4,4).
compute jnsb3=covmat(8,8).
compute jnsb1b3=covmat(4,8).
end if.
end if.
do if (model > 3).
compute ymat(5,im) = coeff((1+im),1).
end if.
do if (xmy = 1).
compute ymat(6,im) = coeff((2+nmeds+im),1).
end if.
do if (vy = 1).
compute ymat(6,im) = coeff((3+nmeds+im),1).
end if.
do if (qy = 1 and vy = 1).
compute ymat(6,im) = coeff((5+nmeds+((im-1)*2)),1).
compute ymat(7,im) = coeff((6+nmeds+((im-1)*2)),1).
end if.
do if (vqy = 1).
compute ymat(6,im) = coeff((6+nmeds+((im-1)*3)),1).
compute ymat(7,im) = coeff((7+nmeds+((im-1)*3)),1).
compute ymat(8,im) = coeff((8+nmeds+((im-1)*3)),1).
end if.
do if (wmy = 1).
compute ymat(7,im) = coeff((3+nmeds+im-wy),1).

```

```

end if.
do if (wmy = 1 and vy = 1).
compute ymat(7,im) =coeff((4+(nmeds*2)+im-
wy),1).
end if.
do if (wmy = 1 and vy = 1 and wvmy = 1).
compute ymat(7,im) =
coeff((6+(nmeds*2)+((im-1)*2)-wy),1).
compute ymat(8,im) =
coeff((7+(nmeds*2)+((im-1)*2)-wy),1).
end if.
do if (wmy = 1 and zmy = 1).
compute ymat(6,im) = coeff((6-(wzy*3)+(wzm-
1)+(nmeds*2)+((im-1)*2)*wzm)+((im-1)*(1-
wzm))-((zy-wzy)*2),1).
compute ymat(7,im) = coeff((3-
zy+im+nmeds),1).
do if (wzmy = 1).
compute ymat(8,im) = coeff((7-
(wzy*3)+(nmeds*2)+((im-1)*2)),1).
end if.
end if.
do if (nmods > 0 and model <> 5).
loop indlp = 1 to nrow(modvals).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp)).
do if (model > 6).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp))*csum(ym
at(5:8,im)*vmat(5:8,indlp)).
end if.
end loop.
compute indboot((bt+(im-
1)*(boot+1)),:)=t(indeff).
do if (imm =1 or imm = 3).
compute
indbootp(bt,im)=ymat(2,im)*ymat(5,im).
do if (imm=3).

```

```

compute
indbootp(bt,(im+nmeds))=ymat(3,im)*ymat(5,i
m).
do if (zdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxz-cminz).
end if.
end if.
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
end if.
end if.
do if ((imm= 2 or imm = 4) or model = 74).
compute
indbootp(bt,im)=ymat(1,im)*ymat(6,im).
do if (imm=4).
compute
indbootp(bt,(im+nmeds))=ymat(1,im)*ymat(7,i
m).
do if (qdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxq-cminq).
end if.
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
end if.
end if.
do if (imm=7).
compute
indbootp(bt,im)=ymat(2,im)*ymat(6,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y

```



```

mat(2,im)*ymat(5,im)*onesmv2)+(ymat(2,im)*y
mat(6,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(6,im)*onesmv1)+(ymat(2,im)*y
mat(6,im)*t(immmv1)).
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxv-cminv).
end if.
end if.
do if (imm=5).
compute
indbootp(bt,im)=ymat(4,im)*ymat(5,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(4,im)*y
mat(5,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(3,im)*ymat(5,im)*onesmv1)+(ymat(4,im)*y
mat(5,im)*t(immmv1)).
do if (wdich=1).

```

```

compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).
end if.
do if (zdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxz-
cminz).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxz-cminz).
end if.
end if.
do if (imm=6).
compute
indbootp(bt,im)=ymat(1,im)*ymat(8,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(1,im)*ymat(6,im)*onesmv2)+(ymat(1,im)*y
mat(8,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(7,im)*onesmv1)+(ymat(1,im)*y
mat(8,im)*t(immmv1)).
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxv-cminv).

```

```

end if.
do if (qdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxq-
cminq).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxq-cminq).
end if.
end if.
do if ((model = 58 or model = 59) and
wvdich=1).
compute
indbootp(bt,im)=(ymat(1,im)*ymat(7,im)*(cmax
w-cminw)+(ymat(2,im)*ymat(5,im)*(cmaxw-
cminw)+(ymat(2,im)*ymat(7,im)*((cmaxw*cma
xw)-(cminw*cminw))).
end if.
end if.
do if (model = 4 or model = 5).
compute
indboot(bt,im)=csum(ymat(1:4,im)*vmat(1:4,1
))*csum(ymat(5:8,im)*vmat(5:8,1)).
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.000000000000001.
end if.
compute sumind=rsum(indboot(bt,:)).
do if (im=nmeds).
compute
pmeff(bt,2:(im+1))=indboot(bt,:)/(sumind+coeff(
(2+nmeds),1)).
end if.
compute
rmeff(bt,(im+1))=indboot(bt,im)/coeff((2+nmeds
),1).
compute
abpseff(bt,(im+1))=indboot(bt,im)/stddevy.

```

```

compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*stddevx.
do if (ncovs > 0 or clsdmy > 0).
compute
abpseff(bt,(im+1))=indboot(bt,im)/sdycov.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*sdxcov.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,1) = r2my-(r2full-r2xy).
compute abmmr = 1.
do if (indboot(bt,im) < 0).
compute abmmr = cmin(abmm).
end if.
do if (indboot(bt,im) > 0).
compute abmmr = cmax(abmm).
end if.
compute kappa2(bt,1)=indboot(bt,im)/abmmr.
compute tmp = indboot(bt,im)/abmmr.
end if.
end if.
do if (model = 6).
do if (nmeds = 2).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(4,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mm
paths(4,3).
compute
indboot(bt,3)=mmpaths(3,1)*mmpaths(4,3).
end if.
do if (nmeds = 3).
compute indboot(bt,1) =
mmpaths(2,1)*mmpaths(5,2).
compute indboot(bt,2) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3).
compute indboot(bt,3) =
mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4).

```

```

compute indboot(bt,4) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*m
mpaths(5,4).
compute indboot(bt,5) =
mmpaths(3,1)*mmpaths(5,3).
compute indboot(bt,6) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4).
compute indboot(bt,7) =
mmpaths(4,1)*mmpaths(5,4).
end if.
do if (nmeds = 4).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(6,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mm
paths(6,3).
compute
indboot(bt,3)=mmpaths(2,1)*mmpaths(4,2)*mm
paths(6,4).
compute
indboot(bt,4)=mmpaths(2,1)*mmpaths(5,2)*mm
paths(6,5).
compute
indboot(bt,5)=mmpaths(2,1)*mmpaths(3,2)*mm
paths(4,3)*mmpaths(6,4).
compute
indboot(bt,6)=mmpaths(2,1)*mmpaths(3,2)*mm
paths(5,3)*mmpaths(6,5).
compute
indboot(bt,7)=mmpaths(2,1)*mmpaths(4,2)*mm
paths(5,4)*mmpaths(6,5).
compute
indboot(bt,8)=mmpaths(2,1)*mmpaths(3,2)*mm
paths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,9)=mmpaths(3,1)*mmpaths(6,3).
compute
indboot(bt,10)=mmpaths(3,1)*mmpaths(4,3)*m
mpaths(6,4).

```

```

compute indboot(bt,11)
=mmpaths(3,1)*mmpaths(5,3)*mmpaths(6,5).
compute indboot(bt,12) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4)*m
mpaths(6,5).
compute indboot(bt,13) =
mmpaths(4,1)*mmpaths(6,4).
compute indboot(bt,14) =
mmpaths(4,1)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,15) =
mmpaths(5,1)*mmpaths(6,5).
end if.
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.00000000000001.
end if.
compute
pmeff(bt,:)=indboot(bt,:)/(rsum(indboot(bt,:))+m
mpaths(nrow(mmpaths),1)).
compute rmeff(bt,:) =
indboot(bt,:)/mmpaths(nrow(mmpaths),1).
compute abpseff(bt,:)=indboot(bt,:)/stddevy.
compute
abcseff(bt,:)=(stddevx*indboot(bt,:))/stddevy.
do if (ncovs > 0 or clsdmy > 0).
compute abpseff(bt,:)=indboot(bt,:)/sdycov.
compute
abcseff(bt,:)=(sdxcov*indboot(bt,:))/sdycov.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,:) = r2my-(r2full-r2xy).
end if.
end if.
end if.
end loop.
end if.
end loop.
do if (savboot = 1 and bt > 1).
compute bootstrp((bt-1),:)=bootcoef.

```

```

end if.
end loop.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
release datayed, dat, datamed, data, y, x,
datatm, dataty, mints2, cons, mnv, tmp.
do if (mc > 0).
compute x1 = sqrt(-
2*ln(uniform(mc,nrow(mcmats))))&*cos((2*3.14
159265358979)*uniform(mc,nrow(mcmats))).
compute x1=x1*chol(mcmats).
loop i=1 to nrow(x1).
compute x1(i,:)=x1(i,:)+t(mccoeff).
end loop.
loop i = 1 to nmeds.
compute
indboot(2:nrow(indboot),i)=x1(:,i)&*x1(:,(i+nme
ds)).
end loop.
end if.
do if (ttt = 0 and model > 3).
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
end if.
do if (model <> 74).
do if (dichy = 0).
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 't', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 't', 'p', 'LLCI', 'ULCI'/format =
F10.4.

```

```

else.
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 'Z', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 'Z', 'p', 'LLCI', 'ULCI' /format =
F10.4.
end if.
end if.
end if.
do if (ttt > 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
compute clbs = {modvnm2, 'Effect', 'SE', 't', 'p',
'LLCI', 'ULCI'}.
compute tratio = (directv&/sedir).
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy))))).
compute outp = {modvalsd, directv, sedir,
tratio, p}.
do if (dichy = 0).
compute temp1=directv-sqrt(jncrit)*sedir.
compute temp2=directv+sqrt(jncrit)*sedir.
compute outp = {outp, temp1, temp2}.
end if.
do if (dichy = 1).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute temp = directv-abs(xp2)*sedir.
compute outp = {outp, temp}.
compute temp = directv+abs(xp2)*sedir.
compute outp = {outp, temp}.
compute clbs = {modvnm2, 'Effect', 'SE', 'Z', 'p',
'LLCI', 'ULCI'}.
end if.
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.

```



```

print outp/title = 'Conditional direct effect(s) of
X on Y at values of the moderator(s)://'format =
F10.4 /cnames = clbs.
end if.
do if (nmods > 0 and model <> 5).
do if (model < 4).
print/title =
'*****'
*****'.
compute zmat(1,1) = 1*(1-yes).
compute cfse = make(nrow(modvals),1,0).
loop #m = 1 to nrow(modvals).
do if (model = 1).
compute zmat(2,1)=modvals(#m,1)**2.
compute zmat(5,1)=2*modvals(#m,1).
end if.
do if (model = 2 or model = 3).
compute zmat(2,1)=modvals(#m,2)**2.
compute zmat(3,1)=modvals(#m,1)**2.
compute
zmat(4,1)=(modvals(#m,1)**2)*(modvals(#m,2)
**2)*(1-yes2).
compute zmat(5,1)=2*modvals(#m,2)*(1-yes).
compute zmat(6,1)=2*modvals(#m,1)*(1-yes).
compute
zmat(7,1)=2*modvals(#m,1)*modvals(#m,2)*(1-
yes).
compute
zmat(8,1)=2*modvals(#m,1)*modvals(#m,2).
compute
zmat(9,1)=2*modvals(#m,1)*(modvals(#m,2)**
2)*(1-yes2).
compute
zmat(10,1)=2*(modvals(#m,1)**2)*modvals(#m
,2)*(1-yes2).
end if.
compute cfse(#m,1)=sqrt(csum(zmat&*cmat)).
end loop.
end if.
do if (nmods > 0).

```

```

compute clbs = {modvnm, 'Effect'}.
loop im = 1 to nmeds.
compute obs = t(indboot(1+(im-1)*(boot+1),:)).
compute outp = {modvals, obs}.
do if (model < 4).
compute tstat = obs&/cfse.
do if (dichy = 0).
compute pval = 2*(1-tcdf(abs(tstat), (n-
ncol(xy)))).
compute temp=obs-sqrt(jncrit)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp=obs+sqrt(jncrit)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 't', 'p', 'LLCI', 'ULCI'}.
compute jnclbs=clbs.
end if.
do if (dichy = 1).
compute pval = 2*(1-cdfnorm(abs(tstat))).
compute temp = obs-abs(xp2)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp = obs+abs(xp2)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute jnclbs=clbs.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indboot((1+(im-
1)*(boot+1)),:).
compute indboot2 = indboot((2+(im-
1)*(boot+1)):(1+(im-1)*(boot+1)+boot),:).
compute mnind = t(csum(indboot2)/boot).
compute stdind=t(sqrt((cssq(indboot2)-
((csum(indboot2)**2)/boot))/(boot-1))).
compute llci=make(1,ncol(indboot2),-999).
compute ulci=make(1,ncol(indboot2),-999).
loop #e = 1 to ncol(indboot2).

```

```

.
compute temp = indboot2(:,#e).
compute temp(GRADE( indboot2(:,#e) )) =
indboot2(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).

```

```

compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {modvals, obs, stdind, t(llci),
t(ulci)}.
compute clbs = {modvnm, 'Effect', 'Boot SE',
'BootLLCI', 'BootULCI'}.
end if.
compute mtemp = mnames(1, im).
compute rlbs =
make(nrow(modvals),1,mnames(1,im)).
do if (model < 4).
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
do if (yes=0).
print outp/title = 'Conditional effect of X on Y at
values of the moderator(s):'/cnames =
clbs/format = F10.4.
else.
compute outp=outp(:,3:ncol(outp)).
compute clbs=clbs(:,3:ncol(clbs)).
compute clbs(1,1)='Contrast'.
print outp/title = 'Contrast of conditional effects
of X on Y'/cnames = clbs/format = F10.4.
end if.
end if.
do if (model > 5 and mod74dic <> 1).

```

```

do if (im = 1).
print/title = 'Conditional indirect effect(s) of X on
Y at values of the moderator(s):'.
end if.
print outp/title = 'Mediator'/rnames =
rlbs/cnames = clbs/format = F10.4.
else if (mod74dic = 1 and model = 74).
do if (im = 1).
print/title = 'Indirect effect(s) of X on Y:'.
compute clbs3=clbs(1,2:ncol(clbs)).
end if.
compute outp3=outp(1,2:ncol(outp)).
print outp3/title = 'Mediator'/rnames =
rlbs/cnames = clbs3/format = F10.4.
end if.
end loop.
loop i = notes to 1 by -1.
do if (note(i,1) = 4 and yes=0).
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 5 and yes=0).
print/title = 'Values for quantitative moderators
are the mean and plus/minus one SD from
mean.'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 14 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    below the mean was replaced
with the minimum because one SD below the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.

```

```

do if (note(i,1) = 15 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    above the mean was replaced
with the maximum because one SD above the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.
end loop.
do if (model = 3 and yes=0).
compute jnvals=make(nrow(matw),7,0).
compute jnvals(:,1)=matw.
compute jnvals(:,2)=jnb1+jnb3*jnvals(:,1).
compute
jnvals(:,3)=sqrt(jnsb1+2*jnvals(:,1)*jnsb1b3+(jn
vals(:,1)*jnvals(:,1))*jnsb3).
compute jnvals(:,4)=jnvals(:,2)/jnvals(:,3).
do if (dichy = 0).
compute jnvals(:,5)=2*(1-tcdf(abs(jnvals(:,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(:,5)=2*(1-
cdfnorm(abs(jnvals(:,4)))).
end if.
compute jnvals(:,6)=jnvals(:,2)-
sqrt(jncrit)*jnvals(:,3).
compute
jnvals(:,7)=jnvals(:,2)+sqrt(jncrit)*jnvals(:,3).
compute clbs={clbs(:,1),clbs(:,3:ncol(clbs))}.
do if (coeffci = 0).
compute jnvals =jnvals(:,1:(ncol(jnvals)-2)).
end if.
print jnvals/title = 'Conditional effect of X*M
interaction at values of W:'/cnames
=clbs/format = F10.4.
end if.
do if (jn = 1 and (model = 1 or model = 3) and
jndich = 0 and yes=0).

```

```

print/title = '***** JOHNSON-
NEYMAN TECHNIQUE
*****'

compute ajn =(jncrit*jnsb3)-(jnb3*jnb3).
compute bjn = 2*((jncrit*jnsb1b3)-(jnb1*jnb3)).
compute cjn = (jncrit*jnsb1)-(jnb1*jnb1).
compute radarg = (bjn*bjn)-(4*ajn*cjn).
compute den = 2*ajn.
compute nrts = 0.
do if (radarg >= 0 and den <> 0).
compute x21 = (-bjn+sqrt(radarg))/den.
compute x22 = (-bjn-sqrt(radarg))/den.
compute roots = 0.
do if (x21 >= jnmin and x21 <= jnmax).
compute nrts = 1.
compute roots = {roots; x21}.
end if.
do if (x22 >= jnmin and x22 <= jnmax).
compute nrts = nrts + 1.
compute roots = {roots; x22}.
end if.
compute roots={roots,make(nrow(roots),2,0)}.
compute modtemp=m.
do if (model=3).
compute modtemp=w.
end if.
do if (nrts > 0).
compute roots = roots(2:nrow(roots),1:3).
compute roots(1,2)=(csum(modtemp <
roots(1,1))/n)*100.
compute roots(1,3)=(csum(modtemp >
roots(1,1))/n)*100.
do if (nrow(roots)=2).
compute roots(2,2)=(csum(modtemp <
roots(2,1))/n)*100.
compute roots(2,3)=(csum(modtemp >
roots(2,1))/n)*100.
end if.
print roots/title = 'Moderator value(s) defining
Johnson-Neyman significance

```

```

region(s)/clabels = 'Value', '% below', '%
above'/format = F10.4.
compute jnvals=make((21+nrts),7,0).
loop i= 0 to 20.
compute jnvals((i+1),1)=jnmin+(i*((jnmax-
jnmin)/20)).
end loop.
loop i = 1 to nrts.
loop j = 2 to nrow(jnvals).
do if ((roots(i,1) > jnvals((j-1),1)) and (roots(i,1)
< jnvals(j,1))).
compute
jnvals((j+1):(21+i),1)=jnvals(j:(20+i),1).
compute jnvals(j,1)=roots(i,1).
end if.
end loop.
end loop.
loop i = 1 to nrow(jnvals).
compute jnvals(i,2)=jnb1+jnb3*jnvals(i,1).
compute
jnvals(i,3)=sqrt(jnsb1+2*jnvals(i,1)*jnsb1b3+(jn
vals(i,1)*jnvals(i,1))*jnsb3).
compute jnvals(i,4)=jnvals(i,2)/jnvals(i,3).
do if (dichy = 0).
compute jnvals(i,5)=2*(1-tcdf(abs(jnvals(i,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(i,5)=2*(1-
cdfnorm(abs(jnvals(i,4)))).
end if.
compute jnvals(i,6)=jnvals(i,2)-
sqrt(jncrit)*jnvals(i,3).
compute
jnvals(i,7)=jnvals(i,2)+sqrt(jncrit)*jnvals(i,3).
end loop.
do if (model = 1).
print jnvals/title = 'Conditional effect of X on Y
at values of the moderator (M)'/cnames
=jnclbs/format = F10.4.

```



```

end if.
do if (model = 3).
compute
jnclbs={jnclbs(:,1),jnclbs(:,3:ncol(jnclbs))}.
print jnclbs/title = 'Conditional effect of X*M on
Y at values of the moderator (W)'/cnames
=jnclbs/format = F10.4.
end if.
end if.
do if (nrts = 0).
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
else.
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
end if.
end if.
do if (model < 4 and (plot = 1)).
compute dataplot =
make((nrow(modvals)*nrow(matx)),(ncol(modv
als)+1),0).
compute tmp = 1.
loop i = 1 to nrow(modvals).
loop j = 1 to nrow(matx).
compute dataplot(tmp,:)=matx(j,1),
modvals(i,:)}.
compute tmp=tmp+1.
end loop.
end loop.
compute dataplot = {dataplot,
make(nrow(dataplot),(1+dichy),0)}.
compute dataplo2 = make(nrow(dataplot),1,1).
do if (model = 1).
compute dataplo2 = {dataplo2, dataplot(:,2),
dataplot(:,1), (dataplot(:,1)*dataplot(:,2))}.
end if.

```

```

do if (model = 2 or model = 3).
compute dataplo2 = {dataplo2, dataplot(:,3),
dataplot(:,1), (dataplot(:,1)*dataplot(:,3)),
dataplot(:,2), (dataplot(:,1)*dataplot(:,2))}.
do if (model = 3).
compute dataplo2 = {dataplo2,
(dataplot(:,2)*dataplot(:,3)),
(dataplot(:,1)*dataplot(:,2)*dataplot(:,3))}.
end if.
end if.
loop i = 1 to nrow(dataplot).
compute tmp=dataplo2(i,:).
do if (ncovs > 0).
compute tmp = {tmp, covmeans}.
end if.
do if (cluster > 0).
compute tmp = {tmp, cldmeans}.
end if.
compute dataplot(i,(ncol(dataplot)-
(dichy)))=tmp*coeffplt.
do if (dichy = 1).
compute
dataplot(i,(ncol(dataplot)))=exp(tmp*coeffplt)/(1
+exp(tmp*coeffplt)).
end if.
end loop.
compute clbs = {xname, modvnm, 'yhat'}.
do if (dichy = 1).
compute clbs = {xname, modvnm, 'ln(odds)',
'prob'}.
end if.
print/title =
*****
*****!

```

```

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (model = 1).
do if (dichy = 0).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Technological_Advan
Financial_Performance.' /space=1.
else if (dichy = 1).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Technological_Advan
logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH Financial_Performance BY
Technological_Advan.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva

```

```

n WITH Financial_Performance BY
Plan_Orga_Pract.' /space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH logodds BY Technological_Advan.'
/space=0.
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH prob BY Technological_Advan.'
/space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Plan_Orga_Pract.'
/space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Plan_Orga_Pract.' /space=0.
end if.
end if.
end if.
do if (model = 2 or model = 3).
do if (dichy = 0).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Technological_Advan
Financial_Performance.' /space=1.
end if.
do if (dichy = 1).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Technological_Advan
logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.

```

```

do if (dichy = 0).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH Financial_Performance BY
Technological_Advan/PANEL ROWVAR=.'
/space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH Financial_Performance BY
Plan_Orga_Pract/PANEL ROWVAR=.'
/space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH logodds BY
Technological_Advan/PANEL ROWVAR=.'
/space=0.
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH prob BY Technological_Advan/PANEL
ROWVAR=.' /space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Plan_Orga_Pract/PANEL
ROWVAR=.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Plan_Orga_Pract/PANEL
ROWVAR=.' /space=0.
end if.
end if.
end if.

```

```

do if (ncovs > 0).
print/title = '* Estimates are based on setting
covariates to their sample means.'.
end if.
end if.
end if.
do if ((imm > 0) or (model = 74 and
mod74dic=0) or ((model = 58 or model = 59)
and wvdich=1))).
do if (imm > 4 and imm < 8).
compute
indbootp={indbootp(:,1:nmeds),indboop1,indbo
op2}.
end if.
compute obsprod = t(indbootp(1,:)).
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indbootp(1,:).
compute indbootp = indbootp(2:(boot+1),:).
compute mnindp = t(csum(indbootp)/boot).
compute stdindp=t(sqrt((cssq(indbootp)-
((csum(indbootp)**2)/boot))/(boot-1))).
compute llcip=make(1,ncol(indbootp),-999).
compute ulcip=make(1,ncol(indbootp),-999).
loop #e = 1 to ncol(indbootp).
.
compute temp = indbootp(:,#e).
compute temp(GRADE( indbootp(:,#e) )) =
indbootp(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.

```

```

end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcip(1,#e)=llcit.
compute ulcip(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.

```

```

end loop.
compute outp = {obsprod, stdindp, t(llcip),
t(ulcip)}.
compute clbs = {'Effect', 'SE(Boot)', 'BootLLCI',
'BootULCI'}.
do if (model = 8 or model = 12).
print/title='-----'.
compute outpimm=outp(1:nmeds,:).
print/title = 'Indirect effect of highest order
product:.'/space=0.
print outpimm/title = 'Mediator'/cnames =
clbs/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
compute clbs3 = {'Index', 'SE(Boot)',
'BootLLCI', 'BootULCI'}.
do if (imm < 5).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=outp((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
end loop.
do if (wvdich=1 or zqdich=1).

```



```

print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****!
compute outpimm=outp(1:nmeds,:).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
compute
outpimm=outp((nmeds+1):nrow(outp),:).
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),clbs3}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3(1,1)=modvnm(1,1).
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.

```

```

print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
end if.
do if (boot = 0).
do if (model = 8 or model = 12).
print/title='-----'.
compute obsimm=obsprod(1:nmeds,:).
print/title = 'Indirect effect of highest order
product:.'/space=0.
print obsimm/title = 'Mediator'/clabels =
'Effect'/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
do if ((imm > 0 and imm < 5) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.

```

```

end if.
compute outpimm=obsprod((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
end loop.
end if.
do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'.
compute outpimm=obsprod(1:nmeds,1).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
compute
outpimm=obsprod((nmeds+1):nrow(obsprod),:)
.
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),'Index'}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm((((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3={modvnm(1,1),'Index'}.
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.

```

```

print {modvnm(1,k4)}/title =
'Moderator:'/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
end if.
end if.
end if.
compute conmake=0.
compute concols = 0.
do if ((model > 3 and model < 7) and (contrast
> 0) and nmods = 0 and nmeds > 1 ).
compute concols =
((ncol(indboot)*(ncol(indboot)-1))/2).
compute indcon=make(nrow(indboot),concols,-
999).
compute conkey = { ' ', ' ', ' ', ' ' }.
compute temp=1.
compute conmake=1.
loop i = 1 to (ncol(indboot)-1).
loop j = (i+1) to (ncol(indboot)).
compute indcon(:,temp)=indboot(:,i)-
indboot(:,j).
do if (contrast = 2).
compute indcon(:,temp)=abs(indboot(:,i))-
abs(indboot(:,j)).
end if.
do if (model <> 6).
compute conkey={conkey; mnames(1,i),
'minus', mnames(1,j)}.
end if.
do if (model = 6).
compute conkey={conkey; indlbl2(i,1), 'minus',
indlbl2(j,1)}.
end if.
compute temp=temp+1.
end loop.
end loop.
end if.

```

```

do if (model = 4 or model = 5).
compute clbs = {'Effect'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obs = {csum(obs); obs}.
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute rlbs = {rlbs;
cntname(1:ncol(indcon),1)}.
end if.
compute outp = obs.
compute outp2=outp.
do if (effsize = 1 and dichy = 0).
compute
pmeff(:,1)=rsum(pmeff(:,2:ncol(pmeff))).
compute
rmeff(:,1)=rsum(rmeff(:,2:ncol(rmeff))).
compute
abpseff(:,1)=rsum(abpseff(:,2:ncol(abpseff))).
compute
abcseff(:,1)=rsum(abcseff(:,2:ncol(abcseff))).
compute eff = {pmeff, rmeff, abpseff, abcseff}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute eff = {eff, r245}.
compute r245obs = {r245(1,1);r245(1,1)}.
compute kappa2ob = kappa2(1,1).
end if.
compute pmobs = t(pmeff(1,1:(nmeds+1))).
compute rmobs = t(rmeff(1,1:(nmeds+1))).
compute psobs = t(abpseff(1,1:(nmeds+1))).
compute csobs = t(abcseff(1,1:(nmeds+1))).
do if (contrast = 0).
compute outp2 = {obs, psobs, csobs, pmobs,
rmobs}.
end if.
do if (contrast > 0).
compute obs2=obs(1:nrow(psobs),:).
compute outp2 = {obs2, psobs, csobs, pmobs,
rmobs}.

```

```

end if.
compute clbs = {'ab', 'ab_ps', 'ab_cs', 'ab/c',
'ab/c'}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute outp2 = {outp2, r245obs}.
compute clbs = {clbs, 'R-sq_med'}.
do if (boot = 0 and mc = 0).
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
do if (boot = 0 and mc = 0).
do if (nmeds = 1).
compute outp2 = outp2(2,:).
compute rlbs = rlbs(2,1).
end if.
do if (model=5).
compute outp2=outp2(:,1:3).
end if.
print outp2/title = 'Indirect effect(s) of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1 and nmeds >
1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0 or mc > 0).
compute indboot = {rsum(indboot), indboot}.
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
compute ones = make(bootsz,1,1).

```

```

do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.

compute estmte=indboot(1,:).
compute indboot = indboot(2:(bootsz+1),:).
compute mnind = t(csum(indboot)/bootsz).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/bootsz))/(bootsz-1))).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).

.

compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.

```

```

compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
do if (effsize = 1 and dichy = 0 and boot > 0).
compute estmte=eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).
.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.

```



```

compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.

```

```

compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
end if.
end if.
do if (boot > 0 or mc > 0).
compute outp = {obs, stdind, t(llci), t(ulci)}.
do if (nmeds = 1).
compute outp = outp(2,:).
compute rlbs = rlbs(2,1).
end if.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
do if (mc > 0).
compute clbs = {'Effect', 'MC SE', 'MC LLCI',
'MC ULCI'}.
end if.
print outp/title = 'Indirect effect of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (dichy = 0 and effsize = 1 and boot > 0).
compute outp = {psobs,
stdindf((2*(nmeds+1)+1):(3*(nmeds+1)),1),
t(llcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))),
t(ulcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Partially standardized indirect
effect of X on Y'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute outp = {csobs,
stdindf((3*(nmeds+1)+1):(4*(nmeds+1)),1),

```

```

t(llcif(1,(3*(nmeds+1)+1):(4*(nmeds+1)))),
t(ulcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))}.
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Completely standardized
indirect effect of X on Y'/rnames = rlbs/cnames
= clbs/format = F10.4.
compute outp = {pmobs,
stdindf(1:(nmeds+1),1),t(llcif(1,1:(nmeds+1))),
t(ulcif(1,1:(nmeds+1))))}.
do if (model = 4).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
compute outp = {rmobs, stdindf(((nmeds+1)+1)
:(2*(nmeds+1)) ,1),
t(llcif(1,((nmeds+1)+1):(2*(nmeds+1)))),
t(ulcif(1,((nmeds+1)+1):(2*(nmeds+1))))}.
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
do if (nmeds = 1 and cluster = 0 and ncovs = 0
and model = 4).
compute r245obs = r245obs(1,1).
compute outp = {r245obs,
stdindf((4*(nmeds+1)+1):(4*(nmeds+1)+1),1),
t(llcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1))),
t(ulcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1))))}.
print outp/title = 'R-squared mediation effect
size (R-sq_med)/rnames = rlbs/cnames =
clbs/format = F10.4.
compute note(notes,1) = 32.
compute notes = notes + 1.

```

```

end if.
end if.
end if.
end if.
do if (normal = 1).
do if (nmeds = 1).
print sobel/title = 'Normal theory tests for
indirect effect'/clabels = 'Effect', 'se', 'Z',
'p'/format = F10.4.
end if.
do if (nmeds > 1).
compute rlbs2 = rlbs(2:nrow(rlbs),1).
print sobel/title = 'Normal theory tests for
specific indirect effects'/rnames = rlbs2/clabels
= 'Effect', 'se', 'Z', 'p'/format = F10.4.
end if.
end if.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
do if (model = 6).
compute clbs = {'eff'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obsfsum=csum(obs).
compute obs = {obsfsum; obs}.
compute indlbl = indlbl(1:nrow(obs),1).
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute indlbl = {indlbl;
cntname(1:ncol(indcon),1)}.
end if.
compute obs2=obs.
do if (boot = 0).
do if (dichy = 0 and effsize = 1).

```

```

compute clbs = {'eff', 'eff_ps', 'eff_cs', 'eff/c',
'eff/c'}.
do if (ncovs = 0).
compute obs = {obs, obs/stddevy,
(obs*stddevx/stddevy), obs/ctot,
obs/mmpaths(nrow(mmpaths),1)}.
end if.
do if (ncovs > 0 or clsdmy > 0).
compute obs = {obs, obs/sdycov,
(obs*sdxcov/sdycov),
obs/(obsfsum+mmpaths(nrow(mmpaths),1)),
obs/mmpaths(nrow(mmpaths),1)}.
end if.
compute obs2=obs.
do if (contrast > 0).
compute obs2 = obs(1:(nrow(obs)-concols),:).
end if.
end if.
print obs2/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute indboot = {rsum(indboot), indboot}.
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(boot+1),:).
compute mnind = t(csum(indboot)/boot).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/boot))/(boot-1))).

```

```

compute temp = nrow(indboot).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).

.

compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.

```

```

compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute obs = {obs, stdind, t(llci), t(ulci)}.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
print obs/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (effsize = 1 and dichy = 0).
compute indboot=indboot(:,1:(ncol(indboot)-
concols)).
compute eff = {rsum(abpseff), abpseff,
rsum(abcseff), abcseff, rsum(pmeff), pmeff,
rsum(rmeff), rmeff}.
compute effobs = eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).

```

```

.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(effobs(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).

```



```

compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute temp2 = stdindf(1:ncol(indboot),1).
compute temp3 = effobs(:,1:ncol(indboot)).
compute templow = llcif(1,1:ncol(indboot)).
compute temphi = ulcif(1,1:ncol(indboot)).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Partially standardized indirect
effect of X on Y'/cnames = clbs/rnames =
indlbl/format = F10.4.
compute temp2 =
stdindf((ncol(indboot)+1):(2*ncol(indboot)),1).
compute temp3 =
effobs(:,(ncol(indboot)+1):(2*ncol(indboot))).
compute templow =
llcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute temphi =
ulcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Completely standardized
indirect effect of X on Y'/cnames = clbs/rnames
= indlbl/format = F10.4.
compute temp2 =
stdindf((2*(ncol(indboot))+1):(3*ncol(indboot)),1
).

```

```

compute temp3 =
effobs(.,(2*(ncol(indboot))+1):(3*ncol(indboot)))
.
compute templow =
llcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute temphi =
ulcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
compute temp =
eff(.,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temp2 =
stdindf((3*(ncol(indboot))+1):(4*ncol(indboot)),1
).
compute temp3 =
effobs(.,(3*(ncol(indboot))+1):(4*ncol(indboot)))
.
compute templow =
llcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temphi =
ulcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
end if.
end if.
do if (nmeds = 2).
compute effkey = {xname, '->', mnames(1,1), '-
>', yname, ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, ' ', ' '}.
compute effkey = {indlbl(2:4,1), effkey}.
end if.

```

```

do if (nmeds = 3).
compute effkey = {xname, '->', mnames(1,1), '->', yname, ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->', mnames(1,1), '->', mnames(1,2), '->', yname, ',', ' '}.
compute effkey = {effkey; xname, '->', mnames(1,1), '->', mnames(1,3), '->', yname, ',', ' '}.
compute effkey = {effkey; xname, '->', mnames(1,1), '->', mnames(1,2), '->', mnames(1,3), '->', yname}.
compute effkey = {effkey; xname, '->', mnames(1,2), '->', yname, ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->', mnames(1,2), '->', mnames(1,3), '->', yname, ',', ' '}.
compute effkey = {effkey; xname, '->', mnames(1,3), '->', yname, ',', ',', ',', ' '}.
compute effkey = {indlbl(2:8,1), effkey}.
end if.

do if (nmeds = 4).
compute effkey = {xname, '->', mnames(1,1), '->', yname, ',', ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->', mnames(1,1), '->', mnames(1,2), '->', yname, ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->', mnames(1,1), '->', mnames(1,3), '->', yname, ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->', mnames(1,1), '->', mnames(1,4), '->', yname, ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->', mnames(1,1), '->', mnames(1,2), '->', mnames(1,3), '->', yname, ',', ' '}.
compute effkey = {effkey; xname, '->', mnames(1,1), '->', mnames(1,2), '->', mnames(1,4), '->', yname, ',', ' '}.

```

```

compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', mnames(1,4), '->', yname}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, ' ', ' ', ' ', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', mnames(1,3), '->', yname, '
', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', mnames(1,4), '->', yname, '
', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,3), '->', yname, ' ', ' ', ' ', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,3), '->', mnames(1,4), '->', yname, '
', ' ', ' ', ' '}.
compute effkey = {effkey; xname, '->',
mnames(1,4), '->', yname, ' ', ' ', ' ', ' ', ' ', ' '}.
compute effkey = {indlbl(2:16,1), effkey}.
end if.
print effkey/title = 'Indirect effect key'/format =
A8.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
end if.
DO IF (medte2=1).
compute
name=t(mdlnms2(2:nrow(mdlnms2),1)).

```

```

compute tmpdat=datayed(:,2:ncol(datayed)).
compute dd=tmpdat(:,1).
compute
dd={dd,tmpdat(:,(2+nmeds)),tmpdat(:,2:(1+nmeds))}.
compute stratify=0.
do if (ncovs > 0).
compute cname = cnames.
compute
dd={dd,tmpdat(:,(3+nmeds):(2+nmeds+ncovs))}.
end if.
compute method = 1.
compute samples = boot.
do if (mc > 0).
compute samples = mc.
compute method=2.
end if.
compute omnibus = 1.
compute booterr = 0.
do if (mcx < 0 or mcx > 4).
compute mcx = 1.
end if.
compute ovals = ncol(design(dd(:,1))).
compute nte = notes.
compute criterr = 0.
compute xskip = 0.
compute dumok = 0.
compute badend = 0.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute priorlo = -9999999.
compute priorhi = 9999999.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute reps = samples.
do if (contrast = 1).

```

```

compute note(nte,1) = 26.
compute nte=nte+1.
end if.
do if (effsize = 1).
compute note(nte,1) = 27.
compute nte=nte+1.
end if.
do if (normal = 1).
compute note(nte,1) = 28.
compute nte=nte+1.
end if.
do if (method = 2).
do if (omnibus = 1).
compute note(nte,1) = 25.
compute nte=nte+1.
end if.
compute samples = 0.
do if (reps < 1000).
compute reps = 1000.
end if.
end if.
do if (method = 1).
compute samples = reps.
end if.
compute temp = ncol(dd).
compute n = nrow(dd).
compute ones = make(n,1,1).
compute temp=dd(:,2).
compute nxd = ncol(temp).
compute nx = nxd.
do if (nx = 1 and mcx > 0).
compute temp = dd.
compute temp(GRADE(dd(:,2)),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,2)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute mnvls = cmin(t(nnvls)).
do if (mnvls < 2).
compute errs = errs + 1.

```

```

compute runerrs(errs,1) = 37.
compute criterr = 1.
end if.
do if (nvls > 9).
compute errs = errs+1.
compute runerrs(errs,1) = 36.
compute criterr = 1.
end if.
do if (criterr = 0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,2).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,2) <> nnvls((temp-1),1)).
compute nnvls(temp,1)=dd(i,2).
compute temp = temp+1.
end if.
end loop.
do if (mcx > 0).
compute x = dummy(:,2:ncol(dummy)).
compute nx = ncol(x).
compute minus1 = make(1,ncol(x),-1).
compute xdes=make((nx+1),3,0).
compute xdes(1,1)=dd(1,2).
compute xdes(1,2)=1.
compute temp = 2.
loop k = 2 to n.
do if (dd(k,2) <> dd((k-1),2)).
compute xdes(temp,2) = k.
compute xdes(temp,1) = dd(k,2).
compute xdes((temp-1),3) = k-1.
compute temp=temp+1.
end if.
end loop.
compute xdes((temp-1),3)=n.
compute xdes = {xdes, (xdes(:,3)-xdes(:,2)+1)}.
do if (stratify <> 0).
compute note(nte,1) = 7.
compute nte=nte+1.

```

```

end if.
do if (mcx = 4).
loop k = 1 to n.
do if (rsum(x(k,:)) = 0).
compute x(k,:) = minus1.
end if.
end loop.
end if.
do if (mcx = 2 or mcx = 3).
compute note(nte,1)=4.
compute nte = nte+1.
loop k = 1 to n.
do if (rsum(x(k,:)) > 0).
loop i = 1 to ncol(x).
do if (x(k,i) = 0).
compute x(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcx = 3).
compute conmat1={-8,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1; 0,0,0,0,0,-
3,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.
loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,:)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute x(k,:)=conmat1((rsum(x(k,:))+1),:).
end loop.
end if.
end if.
compute xdata = x.
compute xskip = 1.
compute xname = ddd(1,1:nx).

```



```

compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcx = 4).
compute dummat(1,:) = minus1.
end if.
do if (mcx = 2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcx = 3).
compute dummat=conmat1.
end if.
compute dummat={nnvls, dummat}.
compute xname2={name(1,2), xname}.
end if.
end if.
end if.
compute nm=nmeds.
compute r2b = make(nm,1,-999).
compute indeff = make((nx*nm),1,-999).
compute cov =
make((nx*nm+nm),(nx*nm+nm),0).
compute mns=make(nrow(cov),1,0).
compute ny = ncol(dd)-nm-nxd-ncovs.
compute y = dd(:,1).
compute ydata = y.
compute yname = name(1,1).
do if (xskip = 0).
compute x = dd(:,2:(1+nxd)).
compute xdata = x.
compute xname = name(1,2:(1+nxd)).
compute indlbs = t(xname).
end if.
compute m = dd(:,(2+nxd):(1+nxd+nm)).
compute mdata = m.

```

```

compute mlab = { ' M1 = ';' ' M2 = ';' ' M3 = ';' '
M4 = ';' ' M5 = ';' ' M6 = ';' ' M7 = ';' ' M8 = ';' '
M9 = ';' ' M10 = '}.
compute mlab = {mlab; ' M11 = ';' ' M12 = ';' '
M13 = ';' ' M14 = ';' ' M15 = '}.
compute mname =
name(1,(2+nxd):(1+nxd+nm)).
do if (ncovs > 0).
compute c =
dd(:,(2+nxd+nm):(1+nxd+nm+ncovs)).
compute x = {x, c}.
compute xdata = x.
compute cdata = c.
compute xname = {xname, cname}.
end if.
compute nms={yname; t(mname);
t(xname(1,1:(ncol(xname)-ncovs)))}.
do if (dumok = 1).
compute nms={yname; t(mname); name(1,2)}.
end if.
do if (nm < 16).
do if (dumok = 1).
print dummat/title = 'Coding of categorical X
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
compute
boots=make((samples+1),((nx+1)*nm),-999).
do if (errs = 0 and criterr = 0).
loop k = 1 to (samples+1).
do if (k = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef=0.
do if (k > 1).
compute u=trunc(uniform(n,1)*n)+1.
compute xdata = x(u,:).
compute mdata = m(u,:).

```

```

compute ydata = y(u,:).
do if (ncovs > 0).
compute cdata = c(u,:).
end if.
end if.
loop i = 1 to nm.
compute start = 1+(i-1)*nx.
compute xmat = {ones, xdata}.
do if (covmy=2).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = mdata(:,i).
compute amat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*amat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-(((1-r2)*(n-1))/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute r2cha = r2.
compute adjr2cha = adjr2.
do if (ncovs > 0 and covmy <> 2).
compute xmat = {ones, cdata}.
compute atmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*atmat)**2).
compute r2step1 = (sstotal-ssresid2)/sstotal.
compute r2cha = r2-r2step1.
compute adjr2cha = adjr2- (1-(((1-r2step1)*(n-
1))/(n-ncol(xmat)))).
compute xmat = {ones, xdata}.
end if.
compute bootcoef={bootcoef,t(amat)}.
compute r2b(i,1)=adjr2cha.
do if (k = 1).

```

```

compute hcinvtX=inv(t( xmat )' xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,1)*hcinvtX*t( xmat
(i3,1)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvtX*t( xmat )' xmat
*hcinvtX
.
compute tratio = amat&/sqrt(diag(covmat)).
compute rnms = mname(1,i).
print rnms/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.
compute lmat = ident(nrow(amat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*amat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*amat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/'format = F10.4
/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'.

```

```

compute op = {amat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*(xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute cov(start:(start+(nx-
1)),start:(start+(nx-
1)))=covmat(2:(ncol(covmat)-(ncovs*(covmy <>
2))),2:(ncol(covmat)-(ncovs*(covmy <> 2)))).
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns(start:(start+(nx-
1)),1)=amat(2:(1+nx)).
end loop.
compute xmat = {ones,mdata,xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (k = 1).
print yname/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.

```

```

do if (ovals = 2).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:'/cnames = nmsd/format = F9.2.
end if.
end if.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(yamat)/n)).
compute LL3 = yamat*&ln(pt2)+(1-yamat)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= yamat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).

```

```

compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*ulp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute bootcoef={bootcoef,t(bmat)}.
compute LL1 = LL2.
compute covmat=varb.
end if.
do if (ovals <> 2).
compute bmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*bmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-(((1-r2)*(n-1))/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute bootcoef={bootcoef,t(bmat)}.
end if.
do if (k = 1).
do if (ovals = 2).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.

```

```

compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).
compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms2/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute
opdirsum=outp((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
end if.
do if (ovals <> 2).
.
compute hcinvXtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtXt*( xmat
(i3,:)).

```



```

end loop.
loop i3=1 to k3.
compute xmat(:,i3) = ( resid(:,ncol( resid
))&/(1-h))&* xmat(:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat(:,i3) = sqrt( mse )&* xmat(:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(bmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/(ncol(xmat)-1).
compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat((2+nmeds):(2+nmeds+(nx-1)),:)=
lmat2.
compute fratiodr =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/nx.
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute df2=(n-ncol(xmat)).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), df2, pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = bmat&/sqrt(diag(covmat)).
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute op = {bmat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat)))) }.

```

```

compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp))))*(xp2/(temp-
(2/3)+(.11/temp)))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms2/format =
F10.4.
compute
opdirsum=op((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
compute xmat = {ones,mdata}.
do if (ncovs > 0).
compute xmat = {xmat, cdata}.
end if.
compute btmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*btmat)**2).
compute r2direct = r2-((sstotal-
ssresid2)/sstotal).
compute pfr = 1-fcdf(fratiodr,(nx),(n-1-nx-nm-
ncovs)).
compute r2direct = {r2direct, fratiodr, nx, (n-1-
nx-nm-ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms2/cnames=rnms2/form
at= F10.4.
end if.
end if.

```

```

do if (k = 1 and toteff = 1).
print yname/title = '*****'
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:.'/format = A8.
compute xmat = {ones, xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(ymat)/n)).
compute LL3 = ymat*&ln(pt2)+(1-ymat*)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= ymat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.

```

```

compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute LL1 = LL2.
compute covmat=varb.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(bmat)-
1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms = {'Constant'; t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).

```

```

compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute optotsum=outp(2:(1+nx),:).
end if.
do if (ovals <> 2).
compute cmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*cmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).

.

compute hcinvtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).

```

```

loop i3=1 to k3.
compute xmat(:,i3) = sqrt( mse )&* xmat(:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(cmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = cmatrix/sqrt(diag(covmat)).
compute op = {cmatrix, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/names = rnms/format =
F10.4.
compute optotsum=op(2:(1+nx),:).
compute ssresid2 = ssresid.
compute r2total = r2.

```

```

compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat(2:(1+nx),:) = lmat2.
compute fratio3 =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/nx.
do if (ncovs > 0).
compute xmat = {ones, cdata}.
compute ctmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*ctmat)**2).
compute r2total = r2-((sstotal-ssresid2)/sstotal).
end if.
compute pfr = 1-fcdf(fratio3,(nx),(n-1-nx-
ncovs)).
compute r2tot2 = {r2total, fratio3, nx, (n-1-nx-
ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns((nrow(mns)-
nm+1):nrow(mns),1)=bmat(2:(1+nm),1).
compute tmp = 1.
loop i = 1 to (nm*nx).
compute
boots(k,i)=mns(i,1)*mns(((nm*nx)+tmp),1).
do if ((i/nx)=trunc(i/nx)).
compute
boots(k,((nm*nx)+tmp))=r2b(tmp,1)*mns(((nm*
nx)+tmp),1).
compute tmp = tmp+1.
end if.
end loop.
do if (savboot = 1 and k > 1).

```

```

compute bootstrp((k-1),:)=bootcoef.
end if.
end loop.
do if (method = 1).
compute llci=make(1,ncol(boots),-999).
compute ulci=make(1,ncol(boots),-999).
compute indeff = t(boots(1,1:(nm*nx))).
compute indeff22=t(boots(1,:)).
compute omni = t(boots(1,((nm*nx)+1) :
ncol(boots))).
do if (nrow(boots) > 1).
compute boots = boots(2:nrow(boots),:).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
loop i = 1 to ncol(boots).
.
compute temp = boots(:,i).
compute temp(GRADE( boots(:,i) )) = boots(:,i).
compute badlo = 0.
compute badhi = 0.
do if ( indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(indeff22(i,1)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2)))+1.

```



```

do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,i)=llcit.
compute ulci(1,i)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute llci = t(llci).
compute ulci= t(ulci).
compute llci={llci,ulci}.
end if.
end if.
do if (method = 2).
compute tmp=1.
loop i = 1 to (nm*nx).
compute
indeff(i,1)=mns(i,1)&*mns(((nm*nx)+tmp),1).

```

```

do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = sqrt(-
2*ln(uniform(reps,nrow(cov))))&*cos((2*3.1415
9265358979)*uniform(reps,nrow(cov))).
compute boots=boots*chol(cov).
loop i = 1 to ncol(boots).
compute boots(:,i)=boots(:,i)+mns(i,1).
end loop.
compute tmp=1.
loop i = 1 to (nm*nx).
compute
boots(:,i)=boots(:,i)&*boots(:,((nm*nx)+tmp)).
compute temp = boots(:,i).
compute temp(GRADE(boots(:,i))) = boots(:,i).
compute boots(:,i) = temp.
do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = boots(:,1:(nm*nx)).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
compute llci =
t(((boots(cilow,:));(boots(cihigh,:)))).
end if.
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
compute rnmssum=rnms(2:(1+nx),1).
do if (ovals <> 2).
print optotsum/title = 'Relative total effects of X
of Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.

```

```

print r2tot2/title = 'Omnibus test of total effect of
X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print optotsum/title = 'Relative total effects of X
on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
end if.
compute
rnmssum=rnms2((2+nmeds):(1+nmeds+nx),1).
do if (ovals <> 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.
print r2direct/title = 'Omnibus test of direct
effect of X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
do if (omnibus = 1 and (method = 1 or method
= 3)).
compute indlbs = {indlbs; 'Omnibus'}.
end if.
loop i = 1 to nm.
compute op = mname(1,i).
print op/title = 'Relative indirect effect(s) of X on
Y through:.'/format = A8/space=0.
do if (method = 2).
compute clbs = {'Effect', 'SE MC', 'MC LLCI',
'MC ULCI'}.
compute op = {indeff, bootse(1:(nm*nx)), lici}.
end if.

```

```

do if (method = 1).
compute clbs = {'Effect', 'SE(boot)', 'LLCI',
'ULCI'}.
compute op = indeff.
do if (samples > 0).
compute op = {indeff, bootse(1:(nm*nx),:),
llci(1:(nm*nx),:)}).
end if.
end if.
compute temp = op((((i-1)*nx)+1):(nx*i),:).
do if ((method = 1) and omnibus = 1).
do if (samples = 0).
compute temp={temp;omni(i,1)}.
end if.
do if (samples > 0).
compute temp = {temp; omni(i,1),
bootse(((nm*nx)+i),:), llci(((nm*nx)+i),:)}).
end if.
end if.
print temp/title = ' '/rnames = indlbs/cnames =
clbs/format = F10.4 /space=0.
print/title = '-----'.
end loop.
end if.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
END IF.
DO IF (mdpbe=1).
compute nm={yname,xname,mnames}.
do if (ncovs > 0).

```

```

compute
nm={yname,cnames,xname,mnames}.
end if.

compute tpx=xname.
compute x = dd(:,2:ncol(dd)).
compute tpy = yname.
compute tempy= dd(:,1).
compute n = nrow(dd).
compute nx=1.
compute criterr=0.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute ddd1 = {'int_1', 'int_2', 'int_3', 'int_4',
'int_5', 'int_6', 'int_7', 'int_8', 'int_9'}.
compute dumok = 0.
compute mcfoc=mcx.
compute mcmmod=mcm.
compute mcloc=(mcfoc>0).
compute con = make(n,1,1).
compute ncovs=ncol(x)-2.
do if (mcfoc > 0 or mcmmod > 0).
compute temp = dd.
compute temp(GRADE(dd(:,(ncol(dd)-
mcloc))),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,ncol(dd)-
mcloc)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute toosmall=rsum(nnvls < 2).
compute mnvls = cmin(t(nnvls)).
do if ((rsum(nnvls < 2)) > 0).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=37.
end if.
do if (nvls > 10).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=36.

```

```

end if.
do if (criterr=0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,ncol(dd)-mcloc).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,ncol(dd)-mcloc) <> nnvls((temp-
1),1)).
compute nnvls(temp,1)=dd(i,ncol(dd)-mcloc).
compute temp = temp+1.
end if.
end loop.
compute dummy = dummy(:,2:ncol(dummy)).
do if (mcfoc=4 or mcmmod=4).
compute minus1=make(1,ncol(dummy),-1).
loop k = 1 to n.
do if (rsum(dummy(k,:)) = 0).
compute dummy(k,:)=minus1.
end if.
end loop.
end if.
do if (mcfoc = 2 or mcfoc = 3) or (mcmmod = 2 or
mcmmod = 3)).
loop k = 1 to n.
do if (rsum(dummy(k,:)) > 0).
loop i = 1 to ncol(dummy).
do if (dummy(k,i) = 0).
compute dummy(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcfoc = 3 or mcmmod=3).
compute conmat1={-8,1,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1,1; 0,0,0,0,0,-
3,1,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.

```

```

loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,:)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute
dummy(k,:)=conmat1((rsum(dummy(k,:))+1),:).
end loop.
end if.
end if.
compute nx = ncol(dummy).
compute xname = ddd(1,1:nx).
compute xdata = dummy.
compute xname = ddd(1,1:nx).
compute xname2={nm(1,(ncol(nm)-mcloc)),
xname}.
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcfoc = 2 or mcmmod=2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcfoc = 3 or mcmmod=3).
compute dummat=conmat1.
end if.
do if (mcfoc = 4 or mcmmod = 4).
compute dummat(1,:)=minus1.
end if.
compute dummat={nnvls, dummat}.
end if.
end if.
do if (criterr=0).
compute y = dd(:,1).
compute ovals = ncol(design(dd(:,1))).

```

```

compute itprob = 0.
do if (ovals = 2).
compute omx = cmax(y(:,1)).
compute omn = cmin(y(:,1)).
compute y(:,1) = (y(:,1) = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute sstotal = csum((y-(csum(y)/n))&**2).
compute outv = t(nm(1,1)).
compute mdtr = nm(1,ncol(dd)).
compute fciv = nm(1,(ncol(dd)-1)).
do if (mcfoc <> 0 or mcmod <> 0).
compute inter=make(n,nx,0).
loop i = 1 to nx.
compute inter(:,i)=dummy(:,i)&*dd(:,ncol(dd)-
(1-mcloc)).
end loop.
do if (ncovs = 0).
compute x={con,dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
do if (ncovs > 0).
compute
x={con,dd(:,2:(1+ncovs)),dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={t(nm(1,2:(1+ncovs)));
nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
end if.
compute xmns = csum(x)&/n.
compute focvals = {1,0,0}.
compute highwarn = 0.
compute lowwarn = 0.

```



```

do if (ncol(x) > (2*nx+2)).
compute covmns = {1, xmns(1,2:(ncol(x)-1-
(2*nx)))}.
compute focvals = {covmns,0,0}.
end if.
compute dfres = n-ncol(x).
do if (ovals = 2).
compute pt2 = make(n,1,(csum(y)/n)).
compute LL3 = y*&ln(pt2)+(1-y)&ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(x),1,0).
compute LL1 = 0.

.

compute pt1lp= pt1.
compute xlp= x.
compute ylp= y.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).

```

```

compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
else if (ovals <> 2).
compute invXtX=inv(t(x)*x).
compute b = invXtX*t(x)*y.
compute resid = y-(x*b).
compute ssresid=csum(resid*resid).
compute msresid=ssresid/dfres.
compute varb = msresid*invXtX.
compute
seint=sqrt(varb(nrow(varb),nrow(varb))).
compute k3 = nrow(b).
do if (hc3 = 1).
compute xhc=x.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)= xhc(i3,:)*invXtX*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).
end loop.
compute varb=(invXtX*t(xhc)*xhc*invXtX).
end if.
compute seb = sqrt(diag(varb)).
compute r2 = 1-(ssresid/sstotal).

```

```

do if (mcfoc <> 0 or mcmod <> 0).
compute ytp=y.
compute xtp=x(:,1:(ncol(x)-nx)).
compute btp = inv(t(xtp)*xtp)*t(xtp)*ytp.
compute residt = ytp-(xtp*btp).
compute ssresidt=csum(residt*residt).
compute r2noint = 1-(ssresidt/sstotal).
end if.
compute pr = ncol(x)-1.
compute lmat = ident(nrow(b)).
compute lmat = lmat(:,2:ncol(lmat)).
compute f =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
pr).
compute pf = 1-fcdf(f,pr,dfres).
compute pf = {sqrt(r2),r2,msresid,f,pr,dfres,pf}.
compute xd=abs(xp2).
end if.
do if (itprob = 0).
compute tstat = b&/seb.
do if (ovals <> 2).
compute p = 2*(1-tcdf(abs(tstat), dfres)).
else if (ovals = 2).
compute p = 2*(1-cdfnorm(abs(tstat))).
end if.
compute outp = {b,seb,tstat,p}.
compute nms = {'constant'; nms; 'interact'}.
do if (ovals <> 2).
compute tval = (dfres* (exp((dfres-
(5/6))*((xd/(dfres-(2/3)+(.11/dfres)))*(xd/(dfres-
(2/3)+(.11/dfres)))))-1)).
compute outp={outp,(b-
sqrt(tval)&*seb),(b+sqrt(tval)&*seb)}.
end if.
compute bb = tval.
do if (dumok = 1).
do if (mcfoc > 0).
print dummat/title = 'Coding of categorical X
variable for analysis:'/cnames = xname2/format
= F5.2.

```

```

else if (mcmod > 0).
print dummat/title = 'Coding of categorical M
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
print yname/title =
*****
*****'/rlabels = 'Outcome: '/format
= A8.
do if (ovals = 2).
compute nmsd = {outv, 'Analysis'}.
print rcd/title = 'Coding of binary Y for
analysis: '/cnames = nmsd/format = F10.4.
compute outp = {outp,(outp(:,1)-
sqrt(tval)&*outp(:,2)),(outp(:,1)+sqrt(tval)&*outp
(:,2))}.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
print outp/title = 'Logistic Regression
Model'/rnames = nms/clabels 'Coeff' 'se' 'Z' 'p'
'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
compute varbtmp=varb.
compute btmp=b.

```

```

compute LL2f=LL2.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=x(:,1:(ncol(x)-nx)).
compute bt1 = make(ncol(xlp),1,0).

.

compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(y-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)*&ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.

```

```

compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL2f.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(LLdiff,nx).
compute rcha={LLdiff,nx,pchi}.
end if.
do if (ovals <> 2).
print pf/title = 'Model Summary'/clabels = 'R' 'R-
sq' 'MSE' 'F' 'df1' 'df2' 'p'/format F10.4.
compute lmat=make(nrow(b),1,0).
compute lmat(nrow(lmat),1)=1.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
1).
compute
rcha=((b(nrow(b),1)/seint)*(b(nrow(b),1)/seint))*
(1-r2)/dfres.
compute rcha = {rcha, fcha, 1, dfres,
outp((pr+1),4)}.
print outp/title = 'Model'/rnames = nms/clabels
'coeff' 'se' 't' 'p' 'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
do if (mcfoc > 0 or mcmod > 0).
compute rcha=r2-r2noint.
compute lmat=make((nrow(b)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
nx).
compute pvalr2c=1-fcdf(fcha,nx,dfres).

```

```

compute rcha = {rcha, fcha, nx, dfres,pvalr2c}.
end if.
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute intkey = {'a', 'b', 'c', 'd', 'e'}.
loop i = 1 to nx.
compute intkey={intkey; ddd1(1,i), ' : ', ddd(1,i),
' X ', nm(1,ncol(dd)-(1-mcloc))}.
end loop.
compute intkey=intkey(2:nrow(intkey),:).
print intkey/title='Product terms key:.'/format =
a8.
end if.
do if (ovals <> 2).
print rcha/title = 'R-square increase due to
interaction:.'/clabels 'R2-chng' 'F' 'df1' 'df2'
'p'/format F10.4.
else if (ovals = 2).
print rcha/title = 'Likelihood ratio test for
interaction:.'/clabels 'Chi-sq' 'df' 'p'/format F10.4.
end if.
print/title =
*****
*****'.
compute mdvar = x(:,(ncol(x)-1)).
do if (mcfoc = 0 and mcmmod = 0).
compute g1 = b((ncol(x)-2),1).
compute g3 = b(ncol(x),1).
compute vg1 = varb((ncol(x)-2),(ncol(x)-2)).
compute vg3 = varb(ncol(x),ncol(x)).
compute covg1g3 = varb((ncol(x)-2), ncol(x)).
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute mdvar = x(:,(ncol(x)-(2*nx))).
end if.
compute mdmin = cmin(mdvar).
compute mdmax = cmax(mdvar).
compute fvar = x(:,(ncol(x)-2)).
compute nval = ncol(design(mdvar)).
compute fvmin = cmin(fvar).

```

```

compute fvmax = cmax(fvar).
do if (mmodval = 999).
compute mnmd = csum(mdvar)/n.
compute tmp = make(n,1,mnmd).
compute sdmd = sqrt(csum(((mdvar-
tmp)**2))/(n-1)).
compute probeval = {mnmd-sdmd; mnmd;
mnmd+sdmd}.
do if (probeval(1,1) < mdmin).
compute lowwarn = 1.
compute probeval(1,1)=mdmin.
end if.
do if (probeval(nrow(probeval),1) > mdmax).
compute highwarn = 1.
compute probeval(nrow(probeval),1)=mdmax.
end if.
do if (quantile = 1).
compute tmp = mdvar.
compute tmp(GRADE(mdvar(:,1)),:)= mdvar.
compute mdvar = tmp.
compute
probeval={mdvar(trunc(0.1*n),1);mdvar(trunc(0
.25*n),1);mdvar(trunc(0.5*n),1);mdvar(trunc(0.7
5*n),1);mdvar(trunc(0.9*n),1)}.
end if.
end if.
do if (nval = 2).
compute probeval = make(2,1,0).
compute probeval(1,1)=cmin(mdvar).
loop i = 1 to n.
do if (mdvar(i,1) <> probeval(1,1)).
compute probeval(2,1) = mdvar(i,1).
BREAK.
end if.
end loop.
end if.
do if (mmodval <> 999).
compute probeval = mmodval.
end if.
compute outp = make(nrow(probeval),7,0).

```



```

do if (mcfoc > 0 or mcmmod > 0).
compute focvals=make(1,ncol(x)+1,1).
do if (mcfoc > 0).
print/title = 'Conditional effect of X on Y at
values of the moderator:'.
print/title = ' '/space=0.
compute rnn2=mdtr.
compute matt=make(nx,6,0).
end if.
loop jj=1 to nrow(probeval).
do if (mcfoc > 0).
loop ii=1 to nx.
compute g1=b((ncol(x)-(2*nx)+ii),1).
compute g3=b((ncol(x)-nx+ii),1).
compute vg1=varb((ncol(x)-(2*nx)+ii),(ncol(x)-
(2*nx)+ii)).
compute vg3=varb((ncol(x)-nx+ii),(ncol(x)-
nx+ii)).
compute covg1g3=varb((ncol(x)-
(2*nx)+ii),(ncol(x)-nx+ii)).
compute x2 = probeval(jj,1).
compute w1 = g1+g3*x2.
compute varw1 =
vg1+(2*x2*covg1g3)+((x2*x2)*vg3).
compute sew1 = sqrt(varw1).
compute t1 = w1/sew1.
do if (ovals <> 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-tcdf(abs(t1), dfres)).
compute cnms = {'Coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)= {w1,sew1,t1,p,llci,ulci}.
else if (ovals = 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-cdfnorm(abs(t1))).
compute cnms = {'Coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)= {w1,sew1,t1,p,llci,ulci}.

```

```

end if.
end loop.
compute rnms=t(xname).
compute mdvalpr=probeval(jj,1).
print mdvalpr/title = 'Moderator
value: '/rnames=rnn2/format= F10.4 /space=0.
print matt/title='
'/cnames=cnms/rnames=rnms/format= F10.4
/space=0.
compute xprob=x(:,(ncol(x)-(2*nx)))-mdvalpr.
loop kk = 1 to nx.
compute xprob={xprob,
(xprob(:,1)&*x(:,(ncol(x)-(2*nx)+kk)))}.
end loop.
compute xprob={x(:,1:((ncol(x)-(2*nx))-
1)),xprob}.
do if (ovals <> 2).
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute residc = y-(xprob*bmultc).
compute ssresidc=csum(residc&*residc).
compute r2c = r2-(1-(ssresidc/sstotal)).
compute fcha2=(dfres*r2c)/(nx*(1-r2)).
do if (hc3 = 1).
loop kk = 1 to nx.
compute xprob={xprob, x(:,(ncol(x)-
(2*nx)+kk))}.
end loop.
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute k3 = nrow(bmultc).
compute xhc=xprob.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)=
xhc(i3,1)*inv(t(xprob)*xprob)*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).

```

```

end loop.
compute
varbc=(inv(t(xprob)*xprob)*t(xhc)*xhc*inv(t(xprob)*xprob)).
compute lmat=make((nrow(bmultc)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha2 =
(t(t(lmat)*bmultc)*inv(t(lmat)*varbc*lmat)*((t(lmat)*bmultc)))/nx).
end if.
compute pvalr2cc=1-fcdf(fcha2,nx,dfres).
compute rcha2 = {r2c, fcha2, nx,
dfres,pvalr2cc}.
print rcha2/title = 'Test of equality of conditional
means at this value of the moderator'/clabels
'R2-chng' 'F' 'df1' 'df2' 'p'/format F10.4.
else if (ovals = 2).
compute btmp=b.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=xprob.
compute bt1 = make(ncol(xlp),1,0).

.
compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.

```

```

end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*ulp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL4.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(lldiff,nx).
compute rcha2 = {LLdiff, nx,pchi}.
print rcha2/title = 'Test of equality of log odds
conditioned on this moderator value'/clabels
'Chi-sq' 'df' 'p'/format F10.4.
end if.
end if.
compute
tttt=make(nrow(dummat),1,probeval(jj,1)).
compute tttt={tttt,dummat(:,2:ncol(dummat))}.
loop kkk=1 to nx.
compute tttt={tttt,tttt(:,1)*tttt(:,1+kkk)}.
end loop.
compute ones=make(nrow(dummat),1,1).
do if (ncol(x) > (2*nx+2)).

```

```

compute
covmnmat=make(nrow(tttt),ncol(covmns),0).
loop kkk=1 to nrow(tttt).
compute covmnmat(kkk,:)=covmns.
end loop.
compute tttt={covmnmat,tttt}.
else.
compute tttt={ones,tttt}.
end if.
compute focvals={focvals;dummat(:,1),tttt}.
do if (mcfoc > 0).
compute yhat={dummat(:,1),(tttt*b)}.
compute cmnms={fciv, 'yhat'}.
do if (ovals <> 2).
print yhat/title = 'Estimated conditional means
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (ovals = 2).
print yhat/title = 'Estimated conditional log odds
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (jj <> nrow(probeval)).
print/title='-----'/space=0.
end if.
end if.
end loop.
compute focvals=focvals(2:nrow(focvals),:).
compute yhat=focvals(:,2:ncol(focvals))*b.
compute
focvals={focvals(:,1),focvals(:,(ncol(focvals)-
(2*nx))),yhat}.
compute cnms={fciv,mdtr,'yhat'}.
do if (mcmod > 0).
compute cnms={mdtr,fciv,'yhat'}.
end if.
end if.
do if (mcmod > 0).
compute outp=make((nx+1),7,0).

```

```

compute bcatm={b((ncol(x)-
(2*nx)),1);b((ncol(x)-nx+1):ncol(x),1)}.
compute outp(:,1)=dummat(:,1).
compute bcatcov=varb((ncol(x)-
nx):ncol(x),(ncol(x)-nx):ncol(x)).
compute bcatcov(1,1)=varb((ncol(x)-
(2*nx)),(ncol(x)-(2*nx))).
compute
bcatcov(2:nrow(bcatcov),1)=varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx)).
compute
bcatcov(1,2:nrow(bcatcov))=t(varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx))).
loop i = 1 to nrow(dummat).
compute
catmval={1,dummat(i,2:ncol(dummat))}.
compute condeff=catmval*bcatm.
compute
condse=sqrt(catmval*bcatcov*t(catmval)).
compute outp(i,2:3)={condeff,condse}.
end loop.
compute outp(:,4)=outp(:,2)&/outp(:,3).
compute outp(:,5) = 2*(1-tcdf(abs(outp(:,4))),
dfres)).
compute outp(:,6) = (outp(:,2)-
sqrt(tval)&*outp(:,3)).
compute outp(:,7) =
(outp(:,2)+sqrt(tval)&*outp(:,3)).
do if (ovals <> 2).
compute cnmms = {xname2(1,1), 'coeff', 'se',
't', 'p', 'LLCI', 'ULCI'}.
end if.
do if (ovals = 2).
compute outp(:,5) = 2*(1-
cdfnorm(abs(outp(:,4)))).
compute cnmms = {xname2(1,1), 'coeff', 'se',
'Z', 'p', 'LLCI', 'ULCI'}.
end if.

```

```

print outp/title = 'Conditional Effect of Focal
Predictor in Groups Defined by the Moderator
Variable:.'/cnames=cnmms/format = F10.4.
end if.
do if (nval > 2 and (mmodval = 999) and
mcmmod = 0).
do if (quantile <> 1).
print/title = 'Moderator values are the sample
mean and plus/minus one SD from mean'.
else.
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
end if.
do if (highwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD above the mean was'.
print/title = 'replaced with the maximum
because one SD above the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
do if (lowwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD below the mean was'.
print/title = 'replaced with the minimum
because one SD below the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
end if.
do if (nval = 2 and (mmodval = 999)).
print/title = 'The moderator variable is
dichotomous'.
end if.
do if (mmodval <> 999 and mcmmod = 0 and
((mmodval < mdmin) or (mmodval > mdmax))).
print/title = 'Warning: Moderator value specified
is outside of the range of the data'.
end if.

```

```

do if (plot = 1).
do if (mcfoc = 0 and mcmmod=0).
compute cnms = {t(nms((ncol(x)-2):(ncol(x)-
1),1)), 'yhat'}.
end if.
do if (ovals = 2).
compute prob =
exp(focvals(:,3))/(1+exp(focvals(:,3))).
compute focvals = {focvals, prob}.
compute cnms = {cnms, 'prob'}.
end if.
print/title =
*****
*****!

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (ovals = 2).
do if (mcfoc=0).
print/title = 'DATA LIST
FREE/Technological_Advan Plan_Orga_Pract
logodds prob.' /space=1.
end if.
do if (mcmmod=0).

```



```

print/title = 'DATA LIST
FREE/Plan_Orga_Pract Technological_Advan
logodds prob.' /space=1.
end if.
else if (ovals <> 2).
do if (mcfoc=0).
print/title = 'DATA LIST
FREE/Technological_Advan Plan_Orga_Pract
Financial_Performance.' /space=1.
end if.
do if (mcmmod=0).
print/title = 'DATA LIST
FREE/Plan_Orga_Pract Technological_Advan
Financial_Performance.' /space=1.
end if.
end if.
print/title = 'BEGIN DATA.' /space=0.
print focvals/title = ' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (ovals <> 2 and mcmmod=0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH Financial_Performance BY
Plan_Orga_Pract.' /space=0.
end if.
do if (ovals <> 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH Financial_Performance BY
Technological_Advan.' /space=0.
end if.
do if (ovals = 2 and mcmmod=0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Plan_Orga_Pract.'
/space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Plan_Orga_Pract.' /space=0.
end if.

```

```

do if (ovals = 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH logodds BY Technological_Advan.'
/space=0.
print/title =
'GRAPH/SCATTERPLOT=Plan_Orga_Pract
WITH prob BY Technological_Advan.'
/space=0.
end if.
do if (ncovs > 0).
print/title = 'NOTE: For data above, covariates
are set to their sample means.'.
end if.
end if.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
end if.
END IF.
end if.
do if (bad > 0).
compute note(notes,1) = 9.
compute notes = notes + 1.
end if.
print/title = '***** ANALYSIS
NOTES AND WARNINGS
*****'.
loop i = 1 to errs.
do if (runerrs(i,1) = 1).
print/title = 'ERROR: One of your declared
mediators is dichotomous. This procedure
can"t be used.'.
end if.

```

```

do if (runerrs(i,1) = 2).
print/title = 'ERROR: For model 6, this
procedure limits the number of mediators to
four.'.
end if.
do if (runerrs(i,1) = 3).
print/title = 'ERROR: For models 1, 2, and 3,
only a single variable can be listed in the M
list.'.
end if.
do if (runerrs(i,1) = 4).
print/title = 'ERROR: You requested a model
involving W but did not provide a valid W
variable name.'.
end if.
do if (runerrs(i,1) = 5).
print/title = 'ERROR: You requested a model
involving Z but did not provide a valid Z
variable name.'.
end if.
do if (runerrs(i,1) = 6).
print/title = 'ERROR: You requested a model
involving Q but did not provide a valid Q
variable name.'.
end if.
do if (runerrs(i,1) = 7).
print/title = 'ERROR: You requested a model
involving V but did not provide a valid V
variable name.'.
end if.
do if (runerrs(i,1) = 8).
print/title = 'ERROR: You specified a W
variable for a model that does not need it'.
end if.
do if (runerrs(i,1) = 9).
print/title = 'ERROR: You specified a Z variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 10).

```

```
print/title = 'ERROR: You specified a Q variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 11).
print/title = 'ERROR: You specified a V variable
for a model that does not need it.'.
end if.
do if (runerrs(i,1) = 12).
print/title = 'ERROR: The variable specified for
W has already been assigned.'.
end if.
do if (runerrs(i,1) = 13).
print/title = 'ERROR: The variable specified for
Z has already been assigned.'.
end if.
do if (runerrs(i,1) = 14).
print/title = 'ERROR: The variable specified for
Q has already been assigned.'.
end if.
do if (runerrs(i,1) = 15).
print/title = 'ERROR: The variable specified for
V has already been assigned.'.
end if.
do if (runerrs(i,1) = 16).
print/title = 'ERROR: You did not provide a
valid Y variable name.'.
end if.
do if (runerrs(i,1) = 17).
print/title = 'ERROR: The variable specified for
Y has already been assigned.'.
end if.
do if (runerrs(i,1) = 18).
print/title = 'ERROR: Model 6 requires more
than one mediator.'.
end if.
do if (runerrs(i,1) = 19).
print/title = 'ERROR: You have not specified a
valid model number.'.
end if.
do if (runerrs(i,1) = 20).
```

```
print/title = 'ERROR: At least one and only one
variable must be listed for X.'.
end if.
do if (runerrs(i,1) = 21).
print/title = 'ERROR: At least one and only one
variable must be listed for Y.'.
end if.
do if (runerrs(i,1) = 22).
print/title = 'ERROR: Iteration didn"t converge
to a solution. Interpret results with caution.'.
end if.
do if (runerrs(i,1) = 23).
print/title = 'ERROR: You specified a clustering
variable that does not exist in your variable
list.'.
end if.
do if (runerrs(i,1) = 24).
print/title = 'ERROR: You specified a clustering
variable that has already been assigned.'.
end if.
do if (runerrs(i,1) = 25).
print/title = 'ERROR: One or more of your M
variables is not listed in the variables list.'.
end if.
do if (runerrs(i,1) = 26).
print/title = 'ERROR: A maximum of 20 cluster
units is allowed. Use multilevel modeling
instead.'.
end if.
do if (runerrs(i,1) = 27).
print/title = 'ERROR: One of the variables in
your model is a constant.'.
end if.
do if (runerrs(i,1) = 28).
print/title = 'ERROR: Dichotomous Y is not
permitted with WS option.'.
end if.
do if (runerrs(i,1) = 29).
print/title = 'ERROR: Insufficient number of
variables in vars= list when using WS option.'.
```

```

end if.
do if (runerrs(i,1) = 30).
print/title = 'ERROR: Too many variables in
vars= list when using WS option. Covariates
not allowed.'.
end if.
do if (runerrs(i,1) = 31).
print/title = 'ERROR: mmodval and wmodval
can"t both be set to zero with contrast option.'.
end if.
do if (runerrs(i,1) = 32).
print/title = 'ERROR: You did not provide a
valid X variable name.'.
end if.
do if (runerrs(i,1) = 33).
print/title = 'ERROR: PROCESS requires all
variable names to be eight characters or
fewer.'.
print/title = '      Please shorten variable names
and reexecute.'/space=0.
end if.
do if (runerrs(i,1)=34).
print/title = 'ERROR: X and M cannot both be
specified as multicategorical.'.
end if.
do if (runerrs(i,1)=35).
print/title = 'ERROR: Multicategorical variable
specification available only for models 1 and
4.'.
end if.
do if (runerrs(i,1)=36).
print/title = 'ERROR: Categorical variables
cannot have more than 10 categories.'.
end if.
do if (runerrs(i,1)=37).
print/title = 'ERROR: Each group must have at
least two cases.'.
end if.
do if (runerrs(i,1)=38).

```

```

print/title = 'ERROR: Cluster option not
available with multicategorical variables.'.
end if.
do if (runerrs(i,1)=39).
print/title = 'ERROR: A maximum of 10
variables is allowed in the M= list.'.
end if.
do if (runerrs(i,1) = 41).
print/title = 'ERROR: MCM option not permitted
in model 4.'.
end if.
end loop.
do if (errs = 0).
do if (boot > 1 or mc > 0).
do if (bconoff = 1 and boot > 1).
print boot/title = 'Number of bootstrap samples
for bias corrected bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (bconoff = 0 and boot > 1).
print boot/title = 'Number of bootstrap samples
for percentile bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (mc > 1).
print mc/title = 'Number of samples for Monte
Carlo confidence intervals:'/format = F8.0.
end if.
do if (booterr = 1).
compute badend = badend(1,2:ncol(badend)).
print badend/title = 'WARNING: Bootstrap CI
endpoints below not trustworthy. Decrease
confidence or increase bootstraps'/format =
F10.4.
end if.
end if.
print conf/title = 'Level of confidence for all
confidence intervals in output:'/format = F8.2.
do if ((center = 1 or ws=1) and (ncol(centvar) >
1)).

```

```

compute centvar = centvar(1,2:ncol(centvar)).
print centvar/title = 'NOTE: The following
variables were mean centered prior to
analysis:' /format = a8.
end if.
loop i = 1 to notes.
do if (note(i,1) = 1).
print/title = 'NOTE: Confidence level restricted
to between 50 and 99.9999%. 95% confidence
is provided in output'.
end if.
do if (note(i,1) = 2).
print/title = 'NOTE: Effect size measures for
indirect effects are not available for models
with dichotomous outcomes'.
end if.
do if (note(i,1) = 3).
print/title = 'NOTE: All standard errors for
continuous outcome models are based on the
HC3 estimator'.
end if.
do if (note(i,1) = 6).
print/title = 'NOTE: The number of bootstrap
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 7).
print/title = 'NOTE: The Johnson-Neyman
method is available only for Models 1 and 3'.
end if.
do if (note(i,1) = 8).
print/title = 'NOTE: The Johnson-Neyman
method cannot be used with a dichotomous
moderator'.
end if.
do if (note(i,1)=9).
print bad/title = 'NOTE: Some bootstrap
samples had to be replaced. The number of
such replacements was:'.
end if.

```



```

do if (note(i,1) = 11).
print nmiss/title = 'NOTE: Some cases were
deleted due to missing data. The number of
such cases was:'.
end if.
do if (note(i,1) = 12).
print /title = 'NOTE: Monte Carlo method
available only for models 4 and 5.
Bootstrapping was used instead.'.
end if.
do if (note(i,1) = 13).
print/title = 'NOTE: The number of Monte Carlo
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 19).
print/title = 'NOTE: Effect sizes not available for
within-subject analyses.'.
end if.
do if (note(i,1) = 16).
print/title = 'NOTE: Normal theory tests not
available for within-subject analyses.'.
end if.
do if (note(i,1) = 17).
print/title = 'NOTE: Monte Carlo confidence
intervals not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 18 and warnrep=0).
print/title = 'WARNING: You have requested
OLS estimation with a dichotomous criterion.'.
print/title = 'Interpret model coefficients and
inferential statistics with caution.'/space=0.
compute warnrep=1.
end if.
do if (note(i,1) = 20).
print/title = 'NOTE: Saving of bootstrap
estimates not available for within-subject
analyses.'.
end if.

```

```
do if (note(i,1) = 22).
print/title = 'NOTE: Effect size option with
covariates requires covariates in models of M
and Y.'.
end if.
do if (note(i,1) = 23).
print/title = 'NOTE: Johnson-Neyman method
not available with multicategorical X'.
end if.
do if (note(i,1) = 24).
print/title = 'NOTE: MMODVAL option not
available with a multicategorical moderator'.
end if.
do if (note(i,1) = 25).
print/title = 'NOTE: Omnibus test for relative
indirect effects is not available with the MC
option.'.
end if.
do if (note(i,1) = 26).
print/title = 'NOTE: CONTRAST option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 27).
print/title = 'NOTE: EFFSIZE option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 28).
print/title = 'NOTE: NORMAL option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 31).
print/title = 'NOTE: The TOTAL option is not
available when Y is dichotomous.'.
end if.
do if (note(i,1) = 32).
print/title = 'NOTE: Kappa-squared is disabled
from output as of version 2.16.'.
end if.
do if (note(i,1) = 33).
```

| | | |
|-----------|----------------|---|
| | | print/title = 'NOTE: COVMY option ignored in models 1, 2, and 3.' |
| | | end if. |
| | | end loop. |
| | | end if. |
| | | end matrix. |
| Resources | Processor Time | 00:00:00.98 |
| | Elapsed Time | 00:00:00.98 |

Run MATRIX procedure:

***** PROCESS Procedure for SPSS Release 2.16.3 *****

Written by Andrew F. Hayes, Ph.D. www.afhayes.com
Documentation available in Hayes (2013). www.guilford.com/p/hayes3

Model = 1
Y = Financia
X = Plan_Org
M = Technolo

Sample size
856

Outcome: Financia

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|-----------|--------|----------|-------|
| .9258 | .8572 | .1734 | 1704.7087 | 3.0000 | 852.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|--------|-------|---------|-------|--------|--------|
| constant | -.3717 | .1160 | -3.2034 | .0014 | -.5994 | -.1440 |
| Technolo | .6559 | .0452 | 14.5084 | .0000 | .5672 | .7447 |
| Plan_Org | .5411 | .0437 | 12.3960 | .0000 | .4554 | .6268 |
| int_1 | -.0289 | .0108 | -2.6748 | .0076 | -.0500 | -.0077 |

Product terms key:

int_1 Plan_Org X Technolo

R-square increase due to interaction(s):

| | R2-chng | F | df1 | df2 | p |
|-------|---------|--------|--------|----------|-------|
| int_1 | .0012 | 7.1547 | 1.0000 | 852.0000 | .0076 |

Conditional effect of X on Y at values of the moderator(s):

| Technolo | Effect | se | t | p | LLCI | ULCI |
|----------|--------|----|---|---|------|------|
|----------|--------|----|---|---|------|------|

| | | | | | | |
|--------|-------|-------|---------|-------|-------|-------|
| 2.7372 | .4621 | .0282 | 16.3963 | .0000 | .4068 | .5174 |
| 3.8107 | .4311 | .0289 | 14.9225 | .0000 | .3744 | .4878 |
| 4.8843 | .4001 | .0338 | 11.8341 | .0000 | .3338 | .4665 |

Values for quantitative moderators are the mean and plus/minus one SD from mean.

Values for dichotomous moderators are the two values of the moderator.

***** JOHNSON-NEYMAN TECHNIQUE *****

There are no statistical significance transition points within the observed range of the moderator.

Data for visualizing conditional effect of X on Y

Paste text below into a SPSS syntax window and execute to produce plot.

DATA LIST FREE/Plan_Orga_Pract Technological_Advan Financial_Performance.
BEGIN DATA.

| | | |
|--------|--------|--------|
| 2.8484 | 2.7372 | 2.7400 |
| 3.8575 | 2.7372 | 3.2063 |
| 4.8666 | 2.7372 | 3.6726 |
| 2.8484 | 3.8107 | 3.3559 |
| 3.8575 | 3.8107 | 3.7910 |
| 4.8666 | 3.8107 | 4.2260 |
| 2.8484 | 4.8843 | 3.9718 |
| 3.8575 | 4.8843 | 4.3756 |
| 4.8666 | 4.8843 | 4.7794 |

END DATA.

GRAPH/SCATTERPLOT=Plan_Orga_Pract WITH Financial_Performance BY
Technological_Advan.

***** ANALYSIS NOTES AND WARNINGS *****

Level of confidence for all confidence intervals in output:
95.00

----- END MATRIX -----

restore.

Matrix

Notes

| | | |
|----------------|--------------------------------|----------------------|
| Output Created | | 20-OCT-2020 16:07:00 |
| Comments | | |
| Input | Active Dataset | DataSet1 |
| | Filter | <none> |
| | Weight | <none> |
| | Split File | <none> |
| | N of Rows in Working Data File | 856 |

Syntax

```
matrix.  
get dat/file = */variables =  
Financial_Performance Opra_Pract  
Technological_Advan /names =  
vnames/missing = 9999.  
compute ninit = nrow(dat).  
get dat/file = */variables =  
Financial_Performance Opra_Pract  
Technological_Advan /names =  
vnames/missing = omit.  
get tmp/file = */variables =  
Financial_Performance /names =  
ynames/missing = omit.  
get tmp2/file = */variables = Opra_Pract  
/names = xnames/missing = omit.  
get tmp3/file = */variables =  
Technological_Advan /names =  
mnames/missing = omit.  
get tmp4/file = */variables = w999999t z999999t  
v999999t q999999t.  
compute wname=tmp(1,1).  
do if (wname = ' ').  
compute wname = 'xxx'.  
end if.  
compute zname=tmp(1,2).  
do if (zname = ' ').  
compute zname = 'xxx'.  
end if.  
compute vname=tmp(1,3).  
do if (vname = ' ').  
compute vname = 'xxx'.  
end if.  
compute qname=tmp(1,4).  
do if (qname = ' ').  
compute qname = 'xxx'.  
end if.  
compute n = nrow(dat).  
compute p0=-.322232431088.  
compute p1 = -1.  
compute p2 = -.342242088547.
```

```

compute p3 = -.0204231210245.
compute p4 = -.0000453642210148.
compute q0 = .0993484626060.
compute q1 = .588581570495.
compute q2 = .531103462366.
compute q3 = .103537752850.
compute q4 = .0038560700634.
compute badend = 0.
compute priorlo = -9999999.
compute priorhi = 9999999.
compute criterr = 0.
compute cluster = 0.
compute clsdmy = 0.
compute jndich = 0.
compute wvdich=0.
compute zqdich=0.
compute mod74dic=0.
compute warnrep=0.
compute contrast = trunc( 0 ).
do if (contrast > 2 or contrast < 0).
compute contrast = 0.
end if.
compute booterr = 0.
compute effsize = ( 0 = 1).
compute note = make(10,1,0).
compute notes = 1.
compute iterr = 0.
compute clsmtn = 0.
compute quantile = ( 0 = 1).
compute mmodval= 999.
compute jn = ( 1 = 1).
compute covcoeff=( 0 =1).
compute center = ( 0 = 1).
compute detail = ( 1 = 1).
compute coeffci = ( 1 = 1).
compute conf = 95.
compute longname = ( 1 =1).
compute bconoff=( 0 <> 1).
compute covmy = trunc( 0 ).
do if (covmy < 0 or covmy > 2).

```

```

compute covmy = 0.
end if.
do if (trunc(conf) >= 100 or (trunc(conf) <=
50)).
compute conf = 95.
compute note(notes,1) = 1.
compute notes = notes + 1.
end if.
do if (n < ninit).
compute nmiss = ninit-n.
compute note(notes,1) = 11.
compute notes = notes + 1.
end if.
compute errs = 0.
compute quantd = {0,0,0,0,0,0}.
compute quantc = {0,0,0,0,0,0}.
compute mcheck = 0.
compute ttt = 0.
compute plot = trunc( 1 ).
compute runerrs = make(50,1,0).
compute model = trunc( 1 ).
compute normal = 0.
compute ws= ( 0 =1).
compute olsdichm= ( 0 =1).
compute olsdichy= ( 0 =1).
compute mcx=trunc( 0 ).
compute mcm=trunc( 0 ).
do if (ws = 1).
do if (effsize = 1).
compute note(notes,1) = 19.
compute notes = notes + 1.
compute effsize=0.
end if.
do if (normal=1).
compute note(notes,1) = 16.
compute notes = notes + 1.
compute normal=0.
end if.
end if.
do if (model <> 4).

```



```

compute ws = 0.
end if.
do if (model > 0 and model < 4 and covmy <>
0).
compute covmy=0.
compute note(notes,1) = 33.
compute notes = notes + 1.
end if.
print/space=0.
do if (jn = 1 and model <> 1 and model <> 3).
compute note(notes,1) = 7.
compute notes = notes + 1.
end if.
do if ((model > 76) or (model < 1)).
compute model = 77.
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 19.
end if.
compute itprob=0.
do if (mcx < 0).
compute mcx=0.
end if.
do if (mcx > 4).
compute mcx=0.
end if.
do if (mcm < 0).
compute mcm=0.
end if.
do if (mcm > 4).
compute mcm=0.
end if.
do if (mcx > 0 and mcm > 0).
compute errs=errs+1.
compute runerrs(errs,1)=34.
compute criterr=1.
end if.
do if ((model <> 1 and model <> 4) and (mcx >
0 or mcm > 0)).
compute errs=errs+1.

```

```

compute runerrs(errs,1)=35.
compute criterr=1.
end if.
do if ((model = 4) and (mcm > 0)).
compute errs=errs+1.
compute runerrs(errs,1)=41.
compute criterr=1.
end if.
do if ((mcx > 0) and jn <> 0).
compute jn=0.
compute note(notes,1) = 23.
compute notes = notes + 1.
end if.
do if (mmodval <> 999 and mcm > 0).
compute note(notes,1) = 24.
compute notes = notes + 1.
end if.
compute toteff = 0.
compute toteff = (( 0 = 1)*( 1 = 4 or 1 = 6)).
compute varorder = 2.
compute hc3 = ( 0 <> 0).
compute cname = 'xxx'.
compute centvar = {'xxx'}.
compute nmeds = ncol(mnames).
do if (longname=0).

```

```
compute criterr = 1.  
compute errs = errs+1.  
compute runerrs(errs,1) = 33.
```

1

2

3

end if.

compute modelm =

```
{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,2;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,3;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,5;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,6;  
1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,7;  
1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,8;  
1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,9;  
1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,3,10;  
1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,5,11;  
1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,5,12;  
1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,5,13;  
0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,2,14;  
0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,2,15;  
0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,4,16;  
0,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,4,17;  
0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,6,18;  
0,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,6,19;  
0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,6,20;
```

1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,7,21;
1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,7,22;
1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,23;
1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,0,24; 1,1,1,1

,0,0,0,0,0,0,0,0,0,0,0,0,0,0,25;
1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,0,26;
1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,0,27;
1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,7,28;
1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,7,29;
1,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,30;
1,1,0,1,0,0,1,1,0,1,0,0,0,0,0,0,31;
1,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,32;
1,1,1,1,0,0,1,1,1,1,0,0,0,0,0,0,33;
1,1,1,1,0,0,1,0,0,1,0,0,0,0,0,0,34;
1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,35;
1,0,0,1,1,0,0,0,0,0,1,1,0,0,0,0,0,36;
1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,37;
1,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,38;
1,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,39;
1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,40;
1,0,0,1,1,0,1,0,0,1,1,0,0,0,0,0,0,41;
1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,42;
1,0,0,1,1,1,1,0,0,1,1,1,0,0,0,0,0,43;
1,0,0,1,1,1,1,0,0,1,0,0,0,0,0,0,44;
1,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,45;
1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,46;
1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,47;
1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,48;
1,1,0,1,1,0,1,1,0,0,0,

0,0,0,0,0,0,0,0,49;
1,1,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,50;
1,1,1,1,1,0,1,1,1,0,0,0,0,0,0,0,0,51;
1,1,1,1,1,0,0,0,0,1,1,0,0,0,0,0,0,52;
1,1,0,1,1,1,1,1,0,0,0,0,0,0,0,0,53;
1,1,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,54;
1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,55;
1,1,1,1,1,1,0,0,0,1,1,1,0,0,0,0,0,56;
1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,57;

```
1,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,58;  
1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,59;  
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,60;  
1,1,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,61;  
1,1,0,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,62;  
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,0,0,0,0,63;  
1,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,64;  
1,0,0,1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,65;  
1,0,0,1,0,0,0,0,0,1,0,0,1,0,0,0,0,0,0,66;  
1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,67;  
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,68;  
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,69;  
1,0,0,1,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,70;  
1,0,0,1,0,0,1,0,0,1,0,0,1,1,1,0,0,0,0,71;  
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,1,1,0,0,72;  
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,1,1,0,0
```

```
,73; 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,74;  
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,75;  
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,1,0,0,0,76;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,77}.
```

```
compute wm = modelm(model, 1).
```

```
compute zm = modelm(model, 2).
```

```
compute wzm = modelm(model, 3).
```

```
compute vy = modelm(model, 4).
```

```
compute qy = modelm(model, 5).
```

```
compute vqy = modelm(model, 6).
```

```
compute wy = modelm(model, 7).
```

```
compute zy = modelm(model, 8).
```

```
compute wzy = modelm(model, 9).
```

```
compute vxy = modelm(model, 10).
```

```
compute qxy = modelm(model, 11).
```

```
compute vqxy = modelm(model, 12).
```

```
compute wmy = modelm(model, 13).
```

```
compute wvmy = modelm(model, 14).
```

```
compute wvxy = modelm(model, 15).
```

```
compute zmy = modelm(model,16).
```

```
compute wzmy = modelm(model,17).
```

```
compute xmy = modelm(model,18).
```

```
compute imm=modelm(model,19).
```

```

do if (model = 74).
compute vdich=0.
end if.
do if (ncol(xname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 20.
compute criterr = 1.
end if.
do if (ncol(mnames) > 10).
compute errs = errs+1.
compute runerrs(errs,1) = 39.
compute criterr = 1.
end if.
do if (ncol(yname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 21.
compute criterr = 1.
end if.
do if (( 999 =0) and ( 999 =0) and (model=3 or
model=2) and (contrast=1)).
compute errs = errs+1.
compute runerrs(errs,1) = 31.
compute criterr = 1.
end if.
compute xlist = (wm or zm or wzm or wy or zy
or wzy or vxy or qxy or vqxy or wvxy or xmy).
compute mlist = (vy or qy or vqy or zmy or wmy
or wzmy or xmy or (model < 4)).
compute bad=0.
do if (criterr = 0).
compute werr = 0.
compute verr = 0.
compute qerr = 0.
compute zerr = 0.
compute yerr = 1.
compute xerr = 1.
compute wlist = (wm or wzm or wy or wzy or
wm or wvmy or wvmy or wvxy or wzmy).
do if (wlist = 1 and wname = 'xxx').
compute werr = 1.

```



```

compute wlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (wlist = 1 and ((wname = qname) or
(wname = vname) or (wname = zname) or
(wname = xname) or (wname = yname))).
compute werr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 12).
end if.
compute zlist = (zm or wzm or zy or wzy or
zmy or wzmy).
do if (zlist = 1 and zname = 'xxx').
compute zerr = 1.
compute zlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (zlist = 1 and ((zname = qname) or
(zname = vname) or (zname = wname) or
(zname = xname) or (zname = yname))).
compute zerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 13.
end if.
compute qlist = (qy or vqy or qxy or vqxy).
do if (qlist = 1 and qname = 'xxx').
compute qerr = 1.
compute qlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (qlist = 1 and ((qname = zname) or
(qname = vname) or (qname = wname) or
(qname = xname) or (qname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 14.
end if.

```

```

compute vlist = (vy or vqy or vxy or vqxy or
wvmy or wvxy).
do if (vlist = 1 and vname = 'xxx').
compute verr = 1.
compute vlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (vlist = 1 and ((vname = zname) or
(vname = qname) or (vname = wname) or
(vname = xname) or (vname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 15.
end if.
do if (wlist = 0 and wname <> 'xxx').
compute werr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 8.
end if.
do if (zlist = 0 and zname <> 'xxx').
compute zerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 9.
end if.
do if (qlist = 0 and qname <> 'xxx').
compute qerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 10.
end if.
do if (vlist = 0 and vname <> 'xxx').
compute verr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 11.
end if.
do if (hc3 = 1).
compute note(notes,1) = 3.
compute notes = notes+1.
end if.
compute alpha2 = (1-(conf/100))/2.

```

```

compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute cons = make(n,1,1).
compute temp = (n*sscp(dat))-
(t(csum(dat))*(csum(dat))).
compute temp = temp/(n*(n-1)).
compute temp = csum((((diag(temp))=0)).
compute temp2=1.
do if (ws = 1).
compute temp2 = (n*sscp(tmp2))-
(t(csum(tmp2))*(csum(tmp2))).
compute temp2 = temp2/(n*(n-1)).
end if.
do if ((temp > 0 and ws = 0) or (temp > 0 and
ws = 1 and temp2 > 0)).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 27.
end if.
compute nmeds = ncol(mnames).
compute mcmats=ident(nmeds*2).
compute mccoeff=make(nmeds*2,1,0).
compute sobel = make(nmeds,4,-999).
do if (model = 6 and nmeds > 4).
compute errs = errs+1.
compute runerrs(errs,1)=2.
end if.
do if (model < 4 and nmeds > 1).
compute errs = errs+1.
compute runerrs(errs,1) = 3.
end if.
compute nmods = (model = 74).
compute bad = 0.
compute intcnt = 1.
compute modvals = 0.
compute modvalsd = 0.
compute yintemp = {'int_1', 'int_2', 'int_3',
'int_4', 'int_5', 'int_6', 'int_7', 'int_8', 'int_9',

```

```

'int_10', 'int_11', 'int_12', 'int_13', 'int_14',
'int_15'}.
compute yintemp = {yintemp, 'int_16', 'int_17',
'int_18', 'int_19', 'int_20', 'int_21', 'int_22',
'int_23', 'int_24', 'int_25', 'int_26', 'int_27',
'int_28'}.
compute yintemp = {yintemp, 'int_29', 'int_30',
'int_31', 'int_32', 'int_33', 'int_34', 'int_35',
'int_36', 'int_37', 'int_38', 'int_39', 'int_40'}.
compute yintemp = {yintemp, 'int_41', 'int_42',
'int_43', 'int_44', 'int_45', 'int_46', 'int_47',
'int_48', 'int_49', 'int_50', 'int_51', 'int_52'}.
compute yintemp = {yintemp, 'int_53', 'int_54',
'int_55', 'int_56', 'int_57', 'int_58', 'int_59',
'int_60', 'int_61', 'int_62', 'int_63', 'int_64'}.
compute
cntname={'(C1)';'(C2)';'(C3)';'(C4)';'(C5)';'(C6)';'(
C7)';'(C8)';'(C9)';'(C10)';'(C11)';'(C12)';'(C13)';(
C14)';'(C15)';'(C16)';'(C17)'}.
compute
cntname={cntname;'(C18)';'(C19)';'(C20)';'(C21
)';'(C22)';'(C23)';'(C24)';'(C25)';'(C26)';'(C27)';(
C28)';'(C29)';'(C30)';'(C31)'}.
compute
cntname={cntname;'(C32)';'(C33)';'(C34)';'(C35
)';'(C36)';'(C37)';'(C38)';'(C39)';'(C40)';'(C41)';(
C42)';'(C43)';'(C44)';'(C45)'}.
compute
cntname={cntname;'(C46)';'(C47)';'(C48)';'(C49
)';'(C50)';'(C51)';'(C52)';'(C53)';'(C54)';'(C55)';(
C56)';'(C57)';'(C58)';'(C59)'}.
compute
cntname={cntname;'(C60)';'(C61)';'(C62)';'(C63
)';'(C64)';'(C65)';'(C66)';'(C67)';'(C68)';'(C69)';(
C70)';'(C71)';'(C72)';'(C73)'}.
compute
cntname={cntname;'(C74)';'(C75)';'(C76)';'(C77
)';'(C78)';'(C79)';'(C80)';'(C81)';'(C82)';'(C83)';(
C84)';'(C85)';'(C86)';'(C87)'}.

```

```

compute
cntname={cntname;'(C88)';'(C89)';'(C90)';'(C91
)';'(C92)';'(C93)';'(C94)';'(C95)';'(C96)';'(C97)';'(
C98)';'(C99)';'(C100)';'(C101)'}.
compute
cntname={cntname;'(C102)';'(C103)';'(C104)';'(
C105)'}.
compute apathnam={'a path'; 'a1 path'; 'a2
path'; 'a3 path'; 'a4 path'; 'a5 path'; 'a6 path';
'a7 path'; 'a8 path'; 'a9 path'; 'a10 path'}.
compute bpathnam={'b path'; 'b1 path'; 'b2
path'; 'b3 path'; 'b4path'; 'b5path'; 'b6 path'; 'b7
path'; 'b8 path'; 'b9 path'; 'b10 path'}.
compute modvnm = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute modvnm2 = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute mlab = { ' M1 ='; ' M2 ='; ' M3 ='; '
M4 ='; ' M5 ='; ' M6 ='; ' M7 ='; ' M8 ='; '
M9 ='; ' M10 ='}.
compute m = make(n,nmeds,1).
compute mmat = make(16,nmeds,0).
compute ymat = make(8,nmeds,0).
compute deco = make(10,1,0).
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
compute iterate = abs(trunc( 10000 )).
compute converge = abs( 0.00000001 ).
compute boot = abs(trunc( 5000 )).
do if (ws=1 and 0 > 0).
compute note(notes,1)=17.
compute notes=notes+1.
end if.
compute mc=(abs(trunc( 0 ))*(1-ws)).
do if (mc > 0 and model > 5).
do if (boot = 0).
compute boot=mc.
end if.

```

```

compute mc=0.
compute note(notes,1)=12.
compute notes = notes+1.
end if.
compute adjust = 0.
do if (mc > 0 and (model > 3 and model < 6)).
compute boot=0.
compute bconoff=0.
end if.
do if (boot > 0 and mc > 0).
compute mc = 0.
end if.
compute savboot = 0.
compute saveboot = ( 0 = 1).
do if (ws=1 and saveboot=1).
compute saveboot=0.
compute note(notes,1) = 20.
compute notes = notes + 1.
compute normal=0.
end if.
do if (saveboot =1 and boot > 0 and model >
3).
compute savboot = 1.
end if.
do if (boot <> 0 or mc <> 0).
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
loop.
compute cilow = rnd(bootsz*(1-(conf/100))/2).
compute cihigh =
trunc((bootsz*(conf/100)+(bootsz*(1-
(conf/100))/2)))+1.
do if (cilow < 1 or cihigh > bootsz).
compute
bootsz=trunc((bootsz+1000)/1000)*1000.
compute adjust = 1.
end if.
end loop if (cilow gt 0 and cihigh le bootsz).

```

```

compute boot=bootsz.
do if (mc > 0).
compute mc = bootsz.
end if.
do if (boot > 0 and mc > 0 and (model > 3 and
model < 6)).
compute boot=0.
end if.
do if (adjust = 1 and boot > 0).
compute note(notes,1)=6.
compute notes = notes+1.
end if.
do if (adjust = 1 and mc > 0).
compute note(notes,1)=13.
compute notes = notes+1.
end if.
end if.
do if (model = 6 and nmeds > 1).
compute mmpaths =
make((nmeds+2),(nmeds+2),0).
do if (nmeds = 2).
compute indboot = make(boot+1, 3, 999).
else if (nmeds = 3).
compute indboot = make(boot+1, 7, 999).
else if (nmeds = 4).
compute indboot = make(boot+1, 15, 999).
end if.
compute indlbl = {'Total:','Ind1 :','Ind2 :','Ind3
:','Ind4 :','Ind5 :','Ind6 :','Ind7 :','Ind8 :','Ind9
:','Ind10:','Ind11:','Ind12:','Ind13:','Ind14:','
Ind15:'}.
compute indlbl2 = {'Ind1','Ind2','Ind3','Ind4';
'Ind5','Ind6','Ind7','Ind8','Ind9','Ind10';
'Ind11','Ind12','Ind13','Ind14','Ind15'}.
compute indces = make(boot+1, 4, 999).
end if.
do if (model < 4).
compute boot = 0.
compute cmat = make(10,1,0).
compute zmat = make(10,1,0).

```

```

end if.
compute nvarch = make(1,ncol(dat),0).
compute wmatch = 0.
compute zmatch = 0.
compute vmatch = 0.
compute qmatch = 0.
compute mmatch=0.
compute minprobe=0.
compute maxprobe=0.
loop i = 1 to ncol(vnames).
do if (vnames(:,i)=yname).
compute y = dat(:,i).
compute nvarch(1,i)=1.
compute yerr = 0.
do if ((yname = xname) or (yname = wname) or
(yname = zname) or (yname = vname) or
(yname = qname)).
compute errs = errs+1.
compute runerrs(errs,1)=17.
end if.
end if.
do if (vnames(:,i)=xname).
compute x = dat(:,i).
compute nvarch(1,i)=1.
compute xdich = 1.
compute xerr=0.
loop jj = 1 to n.
do if ((x(jj,1) <> cmin(x)) and (x(jj,1) <>
cmax(x))).
compute xdich = 0.
break.
end if.
end loop.
compute xmean = csum(x)/n.
do if (center = 1 and (model < 4 or xlist > 0)
and mcx = 0).
compute meanvec = make(n,1, xmean).
compute x = x-meanvec.
compute centvar = {centvar, xname}.
end if.

```



```

compute xmean = csum(x)/n.
compute tmp = x-(cons*xmean).
compute xsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
do if (xdich = 0).
compute quantc(1,6) = 1.
compute matx = {xmean-xsd; xmean;
xmean+xsd}.
do if ( 999 = 999 and quantile = 0).
do if ((matx(1,1) < cmin(x)) and model = 74).
compute matx(1,1)=cmin(x).
compute minprobe=1.
end if.
do if ((matx(3,1) > cmax(x)) and model = 74).
compute matx(3,1)=cmax(x).
compute maxprobe=1.
end if.
end if.
do if (quantile = 1).
compute quantd(1,6) = 1.
compute quantc(1,6) = 0.
compute tmp = x.
compute tmp(GRADE(x)) = x.
compute matx =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (xdich = 1).
compute matx = {cmin(x); cmax(x)}.
do if (model=74).
compute matx=cmin(x).
compute mod74dic=1.
end if.
end if.
do if ( 999 <> 999).
compute matx = 999.
compute quantd(1,6) = 0.
compute quantc(1,6) = 0.
end if.

```

```

end if.
do if (werr = 0 and wlist = 1).
do if (vnames(:,i)=wname).
compute werr = 0.
compute wmatch = 1.
compute w = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(w)/n).
compute w = w-meanvec.
compute centvar = {centvar, wname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute wmean = csum(w)/n.
compute tmp = w-(cons*wmean).
compute wsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute wdich = 1.
loop jj = 1 to n.
do if ((w(jj,1) <> cmin(w)) and (w(jj,1) <>
cmax(w))).
compute wdich = 0.
break.
end if.
end loop.
do if (model = 3).
compute jndich=wdich.
compute jnmin=cmin(w).
compute jnmax=cmax(w).
end if.
do if (wdich = 0).
compute matw = {wmean-wsd; wmean;
wmean+wsd}.
do if ( 999 = 999 and quantile = 0).
do if (matw(1,1) < cmin(w)).
compute matw(1,1)=cmin(w).
compute minprobe=1.
end if.
do if (matw(3,1) > cmax(w)).
compute matw(3,1)=cmax(w).
compute maxprobe=1.

```

```

end if.
end if.
compute quantc(1,1) = 1.
do if (quantile = 1).
compute quantd(1,1) = 1.
compute quantc(1,1) = 0.
compute tmp = w.
compute tmp(GRADE(w)) = w.
compute matw =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (wdich = 1).
compute matw = {cmin(w); cmax(w)}.
compute wvdich=1.
compute cmaw=cmax(w).
compute cminw=cmin(w).
end if.
do if ( 999 <> 999).
compute matw = 999.
compute quantd(1,1) = 0.
compute quantc(1,1) = 0.
end if.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm(1,1)=wname.
compute modmatp(1,1) = 1.
end if.
end if.
do if (zerr = 0 and zlist = 1).
do if (vnames(:,i)=zname).
compute zerr = 0.
compute zmatch = 1.
compute z = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(z)/n).
compute z = z-meanvec.
compute centvar = {centvar, zname}.

```

```

end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute zmean = csum(z)/n.
compute tmp = z-(cons*zmean).
compute zsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute zdich = 1.
loop jj = 1 to n.
do if ((z(jj,1) <> cmin(z)) and (z(jj,1) <>
cmax(z))).
compute zdich = 0.
break.
end if.
end loop.
do if (zdich = 0).
compute matz = {zmean-zsd; zmean;
zmean+zsd}.
do if ( 999 = 999 and quantile = 0).
do if (matz(1,1) < cmin(z)).
compute matz(1,1)=cmin(z).
compute minprobe=1.
end if.
do if (matz(3,1) > cmax(z)).
compute matz(3,1)=cmax(z).
compute maxprobe=1.
end if.
end if.
compute quantc(1,2) = 1.
do if (quantile = 1).
compute quantd(1,2) = 1.
compute quantc(1,2) = 0.
compute tmp = z.
compute tmp(GRADE(z)) = z.
compute matz = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (zdich = 1).
compute matz = {cmin(z); cmax(z)}.

```

```

compute cmaxz=cmax(z).
compute cminz=cmin(z).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matz = 999.
compute quantd(1,2) = 0.
compute quantc(1,2) = 0.
end if.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm(1,2)=zname.
compute modmatp(1,2) = 1.
end if.
end if.
do if (verr = 0 and vlist = 1).
do if (vnames(:,i)=vname).
compute verr = 0.
compute vmatch = 1.
compute v = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(v)/n).
compute v = v-meanvec.
compute centvar = {centvar, vname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute vmean = csum(v)/n.
compute tmp = v-(cons*vmean).
compute vsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute vdich = 1.
loop jj = 1 to n.
do if ((v(jj,1) <> cmin(v)) and (v(jj,1) <>
cmax(v))).
compute vdich = 0.
break.
end if.
end loop.
do if (vdich = 0).

```

```

compute matv = {vmean-vsd; vmean;
vmean+vsd}.
do if ( 999 = 999 and quantile = 0).
do if (matv(1,1) < cmin(v)).
compute matv(1,1)=cmin(v).
compute minprobe=1.
end if.
do if (matv(3,1) > cmax(v)).
compute matv(3,1)=cmax(v).
compute maxprobe=1.
end if.
end if.
compute quantc(1,3) = 1.
do if (quantile = 1).
compute quantd(1,3) = 1.
compute quantc(1,3) = 0.
compute tmp = v.
compute tmp(GRADE(v)) = v.
compute matv = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (vdich = 1).
compute matv = {cmin(v); cmax(v)}.
compute wvdich=1.
compute cmaxv=cmax(v).
compute cminv=cmin(v).
end if.
do if ( 999 <> 999).
compute matv = 999.
compute quantd(1,3) = 0.
compute quantc(1,3) = 0.
end if.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm(1,3)=vname.
compute modmatp(1,3) = 1.
end if.
end if.

```

```

do if (qerr = 0 and qlist = 1).
do if (vnames(:,i)=qname).
compute qerr = 0.
compute qmatch = 1.
compute q = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(q)/n).
compute q = q-meanvec.
compute centvar = {centvar, qname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute qmean = csum(q)/n.
compute tmp = q-(cons*qmean).
compute qsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute qdich = 1.
loop jj = 1 to n.
do if ((q(jj,1) <> cmin(q)) and (q(jj,1) <>
cmax(q))).
compute qdich = 0.
break.
end if.
end loop.
do if (qdich = 0).
compute matq = {qmean-qsd; qmean;
qmean+qsd}.
do if ( 999 = 999 and quantile = 0).
do if (matq(1,1) < cmin(q)).
compute matq(1,1)=cmin(q).
compute minprobe=1.
end if.
do if (matq(3,1) > cmax(q)).
compute matq(3,1)=cmax(q).
compute maxprobe=1.
end if.
end if.
compute quantc(1,4) = 1.
do if (quantile = 1).
compute quantd(1,4) = 1.
compute quantc(1,4) = 0.

```

```

compute tmp = q.
compute tmp(GRADE(q)) = q.
compute matq = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (qdich = 1).
compute matq = {cmin(q); cmax(q)}.
compute cmaxq=cmax(q).
compute cminq=cmin(q).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matq = 999.
compute quantd(1,4) = 0.
compute quantc(1,4) = 0.
end if.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm(1,4)=qname.
compute modmatp(1,4) = 1.
end if.
end if.
do if (vnames(:,i)=cname).
compute cld = dat(:,i).
compute cvname = vnames(:,i).
compute nvarch(1,i)=1.
compute clsmtch = 1.
end if.
loop j = 1 to ncol(mnames).
do if (vnames(:,i)=mnames(1,j)).
compute mmatch = mmatch + 1.
compute m(:,j)=dat(:,i).
do if (center = 1 and nvarch(1,i) = 0 and mlist >
0 and mcm = 0).
compute meanvec = make(n,1,csum(m(:,j))/n).
compute m(:,j) = m(:,j)-meanvec.
compute centvar = {centvar, mnames(1,j)}.
end if.

```



```

compute nvarch(1,i)=1.
compute dichm = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute dichm = 0.
break.
end if.
end loop.
do if ((dichm = 1) and (olsdichm = 1)).
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichm = 1 and model > 3 and mcheck = 0
and olsdichm = 0).
compute errs= errs+1.
compute runerrs(errs,1) = 1.
compute mcheck = 1.
end if.
do if ((model <= 3) and (ncol(mnames)=1)).
compute nmods = nmods + 1.
compute mmean = csum(m(:,j))/n.
compute tmp = m(:,j)-(cons*mmean).
compute msd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute mdich = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute mdich = 0.
break.
end if.
end loop.
do if (model = 1).
compute jndich=mdich.
compute jnmin=cmin(m(:,j)).
compute jnmax=cmax(m(:,j)).
end if.
do if (mdich = 0).
compute matm = {mmean-msd; mmean;
mmean+msd}.

```

```

do if ( 999 = 999 and quantile = 0).
do if (matm(1,1) < cmin(m(:,j))).
compute matm(1,1)=cmin(m(:,j)).
compute minprobe=1.
end if.
do if (matm(3,1) > cmax(m(:,j))).
compute matm(3,1)=cmax(m(:,j)).
compute maxprobe=1.
end if.
end if.
compute quantc(1,5) = 1.
do if (quantile = 1).
compute quantd(1,5) = 1.
compute quantc(1,5) = 0.
compute tmp = m(:,j).
compute tmp(GRADE(m(:,j))) = m(:,j).
compute matm = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (mdich = 1).
compute matm = {cmin(m); cmax(m)}.
end if.
do if ( 999 <> 999).
compute matm = 999.
compute quantd(1,5) = 0.
compute quantc(1,5) = 0.
end if.
compute modmatv(1,5)=nrow(matm).
compute modmat((1:nrow(matm)), 5) = matm.
compute modvnm(1,5)=mnames(1,j).
compute modmatp(1,5) = 1.
end if.
end if.
end loop.
end loop.
do if (minprobe=1).
compute note(notes,1) = 14.
compute notes = notes + 1.

```

```

end if.
do if (maxprobe=1).
compute note(notes,1) = 15.
compute notes = notes + 1.
end if.
do if (cname <> 'xxx' and clsmatch = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 23.
end if.
do if (cname <> 'xxx').
do if ((cname = zname) or (cname = vname)
or (cname = wname) or (cname = xname) or
(cname = yname) or (cname = qname)).
compute errs = errs+1.
compute runerrs(errs,1) = 24.
end if.
end if.
do if (wlist = 1 and werr = 0 and wmatch = 0).
compute werr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (zlist = 1 and zerr = 0 and zmatch = 0).
compute zerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (qlist = 1 and qerr = 0 and qmatch = 0).
compute qerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (vlist = 1 and verr = 0 and vmatch = 0).
compute verr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (yerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 16.

```

```

end if.
do if (xerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 32.
end if.
do if (model = 6 and nmeds < 2).
compute errs = errs+1.
compute runerrs(errs,1) = 18.
end if.
do if (mmatch < ncol(mnames)).
compute errs=errs+1.
compute runerrs(errs,1) = 25.
end if.
do if (cname <> 'xxx').
compute cld = design(cld).
compute cluster = ncol(cld).
compute cld = cld(:,2:ncol(cld)).
compute clsdmy = ncol(cld).
do if (clsdmy > 19).
compute errs = errs+1.
compute runerrs(errs,1) = 26.
end if.
do if (mcx > 0 or mcm > 0).
compute errs = errs+1.
compute runerrs(errs,1) = 38.
end if.
end if.
compute dichy = 1.
loop jj = 1 to n.
do if ((y(jj,1) <> cmin(y)) and (y(jj,1) <>
cmax(y))).
compute dichy = 0.
break.
end if.
end loop.
do if ((dichy = 1) and (olsdichy = 1)).
compute dichy=0.
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.

```

```

do if (dichy = 1).
do if (toteff=1).
compute toteff=0.
compute note(notes,1) = 31.
compute notes = notes + 1.
end if.
compute jncrit=xp2*xp2.
do if (ws = 1).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 28.
end if.
end if.
compute ncovs = ncol(dat)-rsum(nvarch).
do if (effsize = 1 and covmy <> 0 and ncovs >
0 and model > 3 and model < 7).
compute note(notes,1) = 22.
compute notes = notes + 1.
compute effsize = 0.
end if.
do if (ws = 1 and ncovs < nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 29.
end if.
do if (ws = 1 and ncovs > nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 30.
end if.
end if.
do if (errs = 0).
do if (rsum(quantd) > 0).
compute note(notes,1) = 4.
compute notes = notes+1.
end if.
do if (rsum(quantc) > 0).
compute note(notes,1) = 5.
compute notes = notes+1.
end if.

```

```

do if (ncovs > 0).
compute c = make(n,ncovs,0).
compute cnames = {'x'}.
compute j = 1.
loop i = 1 to ncol(vnames).
do if (nvarch(1,i)) = 0.
compute c(:,j) = dat(:,i).
compute nvarch(1,i)=1.
compute j=j+1.
compute cnames = {cnames, vnames(:,i)}.
end if.
end loop.
compute cnames = cnames(1,2:ncol(cnames)).
do if (ws = 1).
compute covmean=csum(c)/n.
loop i = 1 to ncovs.
compute meanvec = make(n,1, covmean(1,i)).
compute c(:,i) = c(:,i)-meanvec.
end loop.
compute centvar = {centvar, cnames}.
compute x=c(:,1:ncovs).
do if (ncovs=nmeds).
compute ncovs=0.
end if.
end if.
end if.
compute names = {yname, xname, mnames,
wname, zname, vname, qname}.
do if (ncovs > 0).
compute names = {names, cnames}.
end if.
do if (dichy = 1 and effsize = 1).
compute note(notes,1) = 2.
compute notes = notes+1.
end if.
do if (model > 3 and model < 6).
compute indeff=make(nmeds,1,0).
compute indboot=make(boot+1,nmeds,999).
do if (mc > 0).
compute indboot=make(mc+1,nmeds,999).

```

```

end if.
do if (effsize = 1 and dichy = 0).
compute rmeff=make(boot+1,nmeds+1,999).
compute abpseff=make(boot+1,nmeds+1,999).
compute abcseff=make(boot+1,nmeds+1,999).
compute pmeff=make(boot+1,nmeds+1,999).
compute r245 = make(boot+1,1,999).
compute kappa2 = make(boot+1,1,999).
end if.
end if.
do if (model = 6 and effsize = 1 and dichy = 0).
compute
rmeff=make(boot+1,ncol(indboot),999).
compute
abpseff=make(boot+1,ncol(indboot),999).
compute
abcseff=make(boot+1,ncol(indboot),999).
compute
pmeff=make(boot+1,ncol(indboot),999).
end if.
do if (nmods > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm(1,tmp) = modvnm(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:nmods).
compute modvnm=modvnm(:,1:nmods).
compute modmatv=modmatv(:,1:nmods).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.

```

```

compute modvals =
make((modmatv(1,1)*modprod(1,1)),
nmods,0).
loop i = 1 to nmods.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvals)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvals(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
do if (model = 74).
compute modvals=matx.
compute modvnm=xname.
end if.
compute vmat = make(8,nrow(modvals),0).
compute vmat(1,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute vmat(5,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute indeff=make(nrow(modvals),1,0).
do if (model <> 5).
compute indboot = make(((boot+1)*nmeds),
nrow(modvals), -99999999).
compute indbootp =
make(boot+1,(nmeds*nmods), -99999999).
do if (imm > 4 and imm < 8).
do if (imm = 7).
compute immmv1=matw.
compute immmv2=matv.
end if.
do if (imm = 5).
compute immmv1=matw.
compute immmv2=matz.
end if.

```



```

do if (imm = 6).
compute immmv1=matv.
compute immmv2=matq.
end if.
compute
indboop1=make((boot+1),(nmeds*(nrow(immmv2))),-9999999).
compute
indboop2=make((boot+1),(nmeds*(nrow(immmv1))),-9999999).
compute onesmv1=make(1,nrow(immmv1),1).
compute onesmv2=make(1,nrow(immmv2),1).
compute
nimvs=nrow(immmv1)+nrow(immmv2).
compute cmmval=0.
compute cmmlbs1={'xx'}.
compute cmmlbs2={'xx'}.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv2}.
compute
cmmlbt=make(nrow(immmv2),1,mnames(1,i)).
compute cmmlbs1={cmmlbs1;cmmlbt}.
end loop.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv1}.
compute
cmmlbt=make(nrow(immmv1),1,mnames(1,i)).
compute cmmlbs2={cmmlbs2;cmmlbt}.
end loop.
compute cmmval=cmmval(2:nrow(cmmval),:).
end if.
end if.
end if.
do if (nmods> 0).
loop i = 1 to ncol(modvals).
do if (modvnm(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm(1,i)=zname).
compute zcol = i.

```

```

end if.
do if (modvnm(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm(1,i)=qname).
compute qcol = i.
end if.
end loop.
end if.
do if (dichy = 1).
compute omx = cmax(y).
compute omn = cmin(y).
compute y = (y = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute data = {cons,y,m,x}.
compute datamed = data.
compute datayed = data.
compute datanm = {'constant'; yname;
t(mnames); xname}.
compute datanmm = {'constant'; yname;
t(mnames); xname}.
compute datanmy = {'constant'; yname;
t(mnames); xname}.
compute yintkey = {' ',' ',' ',' ',' ',' '}.
do if (model < 4 and errs = 0).
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' ', ' '}.
compute datayed = {datayed, x&*m}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(1,i) = 1.
compute vmat(2,i) = modvals(i,1).
end loop.
compute mmat = make(16,nmeds,1).
end if.
do if (model = 2 or model = 3).
compute int1 = x&*w.

```

```

compute datayed = {datayed, w, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,2).
compute vmat(3,i) = modvals(i,1).
compute vmat(4,i) = modvals(i,1)*modvals(i,2).
end loop.
end if.
do if (model = 3).
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*m.
compute int2 = x&*w&*m.
compute datayed = {datayed, int1, int2}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' X', wname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (model = 4 or model = 5).
compute vmat = make(8,1,1).
end if.
compute yintkey2 = yintkey.
do if (wm = 1).
compute int1 = x&*w.
compute datamed = {datamed, w,int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmm = {datanmm; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).

```

```

compute vmat(2,i) = modvals(i,wcol).
end loop.
do if (zm = 1).
compute int1 = x*z.
compute datamed = {datamed, z, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(3,i) = modvals(i,zcol).
end loop.
end if.
do if (wzm = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w*z.
compute int2 = x*w*z.
compute datamed = {datamed, int1, int2}.
loop i = 1 to nrow(modvals).
compute vmat(4,i) =
modvals(i,wcol)*modvals(i,zcol).
end loop.
end if.
end if.
compute mdatacol=ncol(datamed).
compute mintkey = yintkey.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
compute medints = intcnt-1.
do if (vy = 1 or xmy = 1).
compute mp = 1.

```

```

do if (xmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,1).
end loop.
end if.
do if (vy = 1).
compute datayed = {datayed, v}.
compute datanmy = {datanmy; vname}.
compute mmods = 1.
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,vcol).
end loop.
do if (qy = 1).
compute mp = 2.
compute datayed = {datayed,q}.
compute datanmy = {datanmy; qname}.
compute mmods = 2.
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,qcol).
end loop.
end if.
do if (vqy = 1).
compute mp = 3.
compute datayed = {datayed, v*&q}.
compute mmods = 3.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,vcol)*modvals(i,qcol).
end loop.
end if.
end if.
compute mints = make(n,(nmeds*mp),0).
loop i = 0 to (nmeds-1).
do if (i = 0 and vqy = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
vname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.

```

```

do if (vy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (xmy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))*x.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', xname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (qy = 1).
compute mints(:,((i*mp)+2))=m(:,(i+1))*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqy = 1).
compute mints(:,((i*mp)+3))=m(:,(i+1))*v*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end loop.
compute datayed = {datayed, mints}.
end if.
compute mp = 1.
do if (wvmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).

```

```

compute vmat(8,i) =
modvals(i,wcol)*modvals(i,vcol).
end loop.
end if.
compute mints2 = make(n,(nmeds*mp),0).
do if (wmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,wcol).
end loop.
do if (wy = 0 and model > 3).
compute datayed = {datayed, w}.
compute datanmy = {datanmy; wname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wvmy = 1).
compute datayed = {datayed,w&*v}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints2(:,((i*mp)+1))= m(:,(i+1))&*w.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wvmy = 1).
compute mints2(:,((i*mp)+2))=
m(:,(i+1))&*w&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints2}.
end if.

```

```

compute mp = 1.
do if (zmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,zcol).
end loop.
do if (wzmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,zcol)&*modvals(i,wcol).
end loop.
end if.
end if.
do if (zmy = 1).
compute mints3 = make(n,(nmeds*mp),0).
do if (zy = 0).
compute datayed = {datayed, z}.
compute datanmy = {datanmy; zname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wzmy = 1 and wzy = 0).
compute datayed = {datayed,w&*z}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints3(:,((i*mp)+1))= m(:,(i+1))&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wzmy = 1).
compute mints3(:,((i*mp)+2))=
m(:,(i+1))&*w&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', zname}.

```



```

compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints3}.
end if.
compute decoc = 1.
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
do if (wy = 1 and model > 3).
compute datayed = {datayed, w, x&*w}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm2(1,1)=wname.
compute modmatp(1,1) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (zy = 1).
compute datayed = {datayed,z,x&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm2(1,2)=zname.
compute modmatp(1,2) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.

```

```

do if (wzy = 1).
compute datayed = {datayed,w&*z,x&*w&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vxy = 1).
compute datayed = {datayed, x&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm2(1,3)=vname.
compute modmatp(1,3) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (qxy = 1).
compute datayed = {datayed, x&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm2(1,4)=qname.
compute modmatp(1,4) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.

```

```

compute intcnt = intcnt+1.
do if (vqxy = 1).
compute datayed = {datayed, x&*v&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, 'X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end if.
do if (wvxy = 1).
compute datayed = {datayed, x&*w&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, 'X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute modvalsd = 0.
compute ttt = rsum(modmatp).
do if (rsum(modmatp) > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm2(1,tmp) = modvnm2(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:ttt).
compute modvnm2=modvnm2(:,1:ttt).
compute modmatv=modmatv(:,1:ttt).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.

```

```

loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvalsd =
make((modmatv(1,1)*modprod(1,1)), ttt,0).
loop i = 1 to ttt.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvalsd)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvalsd(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
end if.
do if (ttt > 0).
loop i = 1 to ncol(modvalsd).
do if (modvnm2(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm2(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm2(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm2(1,i)=qname).
compute qcol = i.
end if.
end loop.
compute directv = make(nrow(modvalsd),1,1).
do if (wy = 1).
compute directv ={directv, modvalsd(:,wcol)}.
end if.

```

```

do if (zy = 1).
compute directv = {directv, modvalsd(:,zcol)}.
end if.
do if (wzy = 1).
compute directv = {directv,
(modvalsd(:,wcol)*modvalsd(:,zcol))}.
end if.
do if (vxy = 1).
compute directv = {directv, modvalsd(:,vcol)}.
end if.
do if (qxy = 1).
compute directv = {directv, modvalsd(:,qcol)}.
end if.
do if (vqxy = 1).
compute directv = {directv,
(modvalsd(:,vcol)*modvalsd(:,qcol))}.
end if.
do if (wvxy = 1).
compute directv = {directv,
(modvalsd(:,vcol)*modvalsd(:,wcol))}.
end if.
end if.
compute ydatacol = ncol(datayed).
do if (ncovs > 0).
do if (covmy <> 2).
compute datamed = {datamed,c}.
end if.
do if (covmy <> 1).
compute datayed = {datayed,c}.
end if.
do if (covmy = 1 and model = 4 and mcx > 0).
compute datayed = {datayed,c}.
end if.
compute covmeans = csum(c)/n.
end if.
do if (cluster > 0).
compute datamed = {datamed,cld}.
compute datayed = {datayed, cld}.
compute cldmeans = csum(cld)/n.
end if.

```

```

compute mst = 3.
compute mnd = mst+nmeds-1.
compute ydatacol=ncol(datayed).
compute mdatacol=ncol(datamed).
do if (ncovs > 0).
compute datanmy = {datanmy; t(cnames)}.
do if (model > 3).
compute datanmm = {datanmm; t(cnames)}.
end if.
end if.
compute datanmy = {'constant';
datanmy(3:nrow(datanmy),1)}.
do if (model > 3).
compute datanmm = {'constant';
datanmm(3:nrow(datanmm),1)}.
end if.
compute amm = make(2,1,0).
compute abmm = make(2,1,0).
compute mnv = csum(datayed(:,2)/n).
compute mnv = make(n,1,mnv).
compute ssty = csum((datayed(:,2)-mnv)**2).
compute sigma = (n*sscp(datayed)-
(t(csum(datayed))*(csum(datayed)))).
compute sigma = sigma/(n*(n-1)).
do if (ws=1).
compute stddevm=diag(sigma).
compute
stddevm=sqrt(stddevm(3:(nmeds+2),1)).
end if.
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute r2xy =
(sigma(2,(3+nmeds))/(stddevy*stddevx))**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))**2.
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).

```

```

do if (model = 4 and nmeds = 1 and cluster = 0
and ncovs = 0).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
compute datatm = datamed.
compute dataty = datayed.
compute mdlrms2 = { '1' ; yname; xname}.
compute mdlrms = {'Model ='; ' Y ='; ' X ='
}.
do if (ws = 1).
compute mdlrms2 = { '1' ; yname}.
compute mdlrms = {'Model ='; ' Y ='}.
end if.
loop i = 1 to ncol(mnames).
compute mdlrms2 = {mdlrms2; mnames(1,i)}.
do if (i = 1 and ncol(mnames) = 1).
compute mdlrms = {mdlrms; ' M ='}.
else.

```

```

compute mdlInms = {mdlInms; mlab(i,1)}.
end if.
end loop.
do if (wname <> 'xxx').
compute mdlInms2 = {mdlInms2; wname}.
compute mdlInms = {mdlInms; '  W = '}.
end if.
do if (zname <> 'xxx').
compute mdlInms2 = {mdlInms2; zname}.
compute mdlInms = {mdlInms; '  Z = '}.
end if.
do if (vname <> 'xxx').
compute mdlInms2 = {mdlInms2; vname}.
compute mdlInms = {mdlInms; '  V = '}.
end if.
do if (qname <> 'xxx').
compute mdlInms2 = {mdlInms2; qname}.
compute mdlInms = {mdlInms; '  Q = '}.
end if.
do if (jn = 1 and model = 1 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
do if (jn = 1 and model = 3 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
compute mdpbe=0.
compute medte2=0.
do if ( 1 =4 and mcx > 0).
compute medte2=1.
end if.
do if (((mcx > 0) or (mcm > 0)) AND model =
1).
compute mdpbe=1.
compute dd={datayed(:,4),datayed(:,3)}.
do if (ncovs > 0).
compute dd={datayed(:,6:(5+ncovs)),dd}.
end if.
compute dd={datayed(:,2),dd}.

```



```

end if.
compute yes = ((nrow(modvals)=1) and
(contrast = 1) and (model = 3 or model = 2)).
compute yes2 = ((nrow(modvals)=1) and
(contrast = 1) and (model = 2)).
print/title = '***** PROCESS Procedure
for SPSS Release 2.16.3 *****'.
print/title = '      Written by Andrew F. Hayes,
Ph.D.    www.afhayes.com'.
print/title = '  Documentation available in
Hayes (2013).
www.guilford.com/p/hayes3'/space=0.
print mdlnms2/title =
*****
*****'/rnames = mdlnms/format =
a8.
do if (ncovs > 0).
print cnames/title = 'Statistical Controls: '/rlabels
= 'CONTROL=' /format a8.
end if.
print n/title = 'Sample size' /format F10.0.
do if (cluster > 0).
print cluster/rnames = cvname/title =
'Clustering Variable and Number of Clusters'.
end if.
do if (model > 3 and ( 'random' <> 'random')).
compute seedt= 'random'.
print seedt/title='Custom' /format=A12/rlabel =
'Seed:'.
end if.
do if (mdpbe=0 and medte2 = 0).
loop bt = 1 to (boot+1).
do if (bt = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef = 0.
do if (bt > 1).
loop.
compute v=trunc(uniform(n,1)*n)+1.

```

```

compute datayed = dataty(v,:).
do if (ws=1).
compute
bmeans=csum(datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed)))/n.
compute cntmat=make(n,nmeds,1).
compute bmeans2=bmeans.
do if (nmeds > 1).
compute bmeans2=mdia(bmeans).
end if.
compute cntmat=cntmat*bmeans2.
compute datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed))=datayed(:,(ncol(data
yed)-nmeds+1):ncol(datayed))-cntmat.
end if.
compute detcheck=det(t(datayed)*datayed).
compute rk=(detcheck <> 0).
do if (model > 3).
compute datamed = datatm(v,:).
compute detcheck=det(t(datamed)*datamed).
do if (rk=1).
compute rk=(detcheck <> 0).
end if.
end if.
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
compute temp=diag(sigma).
compute bad = bad+(1-rk).
compute false = 1.
end loop if (rk = 1).
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nme
ds)).
do if (model = 4 and nmeds = 1 and ncovs = 0
and cluster = 0).

```

```

compute r2xy =
(sigma(2,4)/(stddevy*stddevx))&**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))&**2.
compute sstot = sigma(2,2)*(n-1).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
end if.
do if (model > 3).
loop im = 1 to nmeds.
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant';
datanmm((2+nmeds):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
do if (model = 6).

```

```

do if (im = 1).
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
end if.
do if (im > 1).
compute xm={cons, datamed(:,3:(im+1)),
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant'; datanmm(2:im,1);
datanmm((mnd):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
compute
mmpaths((im+1),(2:im))=t(coeff(2:im,1)).
end if.
end if.
do if (ws = 1).
compute
coeff(1,1)=csum(datamed(:,(2+im)))/n.
end if.
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
do if (bt = 1).
compute resid=datamed(:,(2+im))-xm*coeff.
compute sse =cssq(resid).
compute mse = sse/(n-ncol(xm)).
compute mnv = csum(data(:,(2+im)))/n.
compute mnv = make(n,1,mnv).
compute sstm = csum((data(:,(2+im))-
mnv)**2).
compute k3 = nrow(coeff).
do if (hc3 = 1).
compute h = xm(:,1).
loop i3=1 to n.
compute h(i3,1)= xm(i3,:)*invXtX*t(xm(i3,:)).
end loop.

```

```

loop i3=1 to k3.
compute xm(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xm(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xm(:,i3) = sqrt(mse)&*xm(:,i3).
end loop.
end if.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute hccov=invXtX*t(xm)*xm*invXtX.
compute mcmats(im,im)=hccov(2,2).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*hccov*lmat)*((t(lmat)
)*coeff))/dfnum.
compute coeff = coeff(1:(nrow(coeff)-
clsdmy),1).
compute mccoeff(im,1)=coeff(2,1).
compute standerr =
sqrt(diag(invXtX*t(xm)*xm*invXtX)).
compute standerr = standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1).
compute standerr(1,1)=stddevm(im,1)/sqrt(n).
end if.
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-
ncol(xm)))).
compute temp=(n-ncol(xm)).
do if (ws=1).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute temp=(n-1).
end if.
compute xd = abs(xp2).

```

```

compute temp = (temp* (exp((temp-
(5/6))*((xd/(temp-(2/3)+(.11/temp)))*(xd/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = coeff-
sqrt(abs(temp))*standerr.
compute temp2 =
coeff+sqrt(abs(temp))*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (ws=1).
compute op=op(1,:).
compute xnmn =
apathnam(((nmeds>1)+1+(im-1)),1).
end if.
compute sobel(im,1) = coeff(2,1).
compute sobel(im,2)=standerr(2,1).
compute temp = mnames(1,im).
compute r2full = 1-(sse/sstm).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute summ = {sqrt(r2full), r2full, mse,
Fratio, dfnum, dfden, pfr}.
do if (detail = 1).
print temp/title =
*****
*****'/rlabels = 'Outcome:'/format
= A8.
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels 'R',
'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
print op/title = 'Model'/rnames = xnmn/clabels
'coeff', 'se', 't', 'p', 'LLCI', 'ULCI'/format = F10.4.
do if (covcoeff=1 and ws <> 1).
compute
hccovtmp=hccov(1:nrow(op),1:nrow(op)).
compute cnamestp=t(xnmn).

```

```

print hccovtmp/title='Covariance matrix of
regression parameter
estimates'/rnames=xmnm/cnames=cnamestp/f
ormat= F10.4.
end if.
do if (nmods > 0 and nrow(mintkey) > 1).
print mintkey/title = 'Product terms key:'/format
= A8.
end if.
do if (ws = 1).
compute wsdf=n-1.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
end if.
end if.
compute ymat(1,im) = coeff(2,1).
do if (ws = 1).
compute ymat(1,im)=coeff(1,1).
end if.
do if (wm = 1).
compute ymat(2,im) = coeff(4,1).
do if (zm = 1).
compute ymat(3,im) = coeff(6,1).
do if (wzm = 1).
compute ymat(4, im) = coeff(8,1).
end if.
end if.
end if.
do if (model = 6).
compute mmpaths((im+1),1)=coeff((im+1),1).
end if.
end loop.
end if.
loop totlp = 1 to (1+(toteff*(bt = 1))).
do if (toteff = 1 and totlp =2).
compute xy =
{cons,datayed(:,(3+nmeds):ydatacol)}.
else.
compute xy={cons, datayed(:,3:ydatacol)}.
end if.

```

```

do if (dichy = 1).
compute pt2 =
make(nrow(datayed(:,2)),1,(csum(datayed(:,2))
/n)).
compute LL3 = datayed(:,2)*ln(pt2)+(1-
datayed(:,2))*ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xy),1,0).
compute LL1 = 0.
compute xy22=xy.
loop jjj = 1 to iterate.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.
compute coeff =
bt1+inv(t(xy22)*xy)*t(xy)*(datayed(:,2)-pt1).
compute pt1 = 1/(1+exp(-(xy*coeff))).
compute itprob = csum((pt1 <
.00000000000001) or (pt1 >
.99999999999999)).
do if (itprob = 0).
compute LL = datayed(:,2)*ln(pt1)+(1-
datayed(:,2))*ln(1-pt1).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
break.
end if.
compute bt1 = coeff.
compute LL1 = LL2.
end loop.
do if (jjj >= iterate and iterr = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
computer iterr = 1.
end if.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.

```



```

compute covmat = inv(t(xy22)*xy).
release xy22.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
end if.
do if (dichy = 0).
compute invXtX = inv(t(xy)*xy).
compute coeff =invXtX*t(xy)*datayed(:,2).
do if (effsize=1 and model > 3 and model < 7
and (ncovs > 0 or clsdmy > 0)).
compute xysdy =
{cons,datayed(:,(4+nmeds):ydatacol)}.
compute coeffsd
=inv(t(xysdy)*xysdy)*t(xysdy)*datayed(:,2).
compute resid=datayed(:,2)-(xysdy*coeffsd).
compute sdycov =sqrt(csum(resid&*resid)/(n-
ncol(xysdy))).
compute xvaron=datayed(:,(3+nmeds)).
compute
coeffx2=inv(t(xysdy)*xysdy)*t(xysdy)*xvaron.
compute residx2=xvaron-(xysdy*coeffx2).
compute sdxcov
=sqrt(csum(residx2&*residx2)/(n-ncol(xysdy))).
end if.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4 and bt > 1).
compute resid=datayed(:,2)-xy*coeff.
compute sse = cssq(resid).
compute r2full = 1-(sse/sstot).
end if.
do if (bt =1).
compute resid=data(:,2)-xy*coeff.

```

```

compute k3 = nrow(coeff).
compute sse = cssq(resid).
compute mse = sse/(n-ncol(xy)).
compute
hcnor=coeff&/(sqrt(diag(mse*invXtX))).
do if (hc3 = 1).
compute h = xy(:,1).
loop i3=1 to n.
compute h(i3,1)= xy(i3,:)*invXtX*t(xy(i3,:)).
end loop.
loop i3=1 to k3.
compute xy(:,i3) = (resid(:,ncol(resid))&/(1-
h))*xy(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xy(:,i3) = sqrt(mse)*xy(:,i3).
end loop.
end if.
compute covmat = (invXtX*t(xy)*xy*invXtX).
end if.
end if.
do if (bt = 1).
do if (model = 2).
compute xy2={cons, datayed(:,3:ydatacol)}.
compute temp = ncol(xy2).
do if (temp > 6).
compute xy3=xy2(:,7:temp)}.
end if.
compute xy2={xy2(:,1:3), xy2(:,5)}.
do if (temp > 6).
compute xy2={xy2, xy3}.
release xy3.
end if.
compute invXtX = inv(t(xy2)*xy2).
compute coeff2 =invXtX*t(xy2)*datayed(:,2).
compute ssem2=cssq(datayed(:,2)-xy2*coeff2).
release xy2.
end if.

```

```

compute stderr = sqrt(diag(covmat)).
do if (totlp = 1).
compute
mcmats((nmeds+1):ncol(mcmats),(nmeds+1):n
col(mcmats))=covmat(2:(1+nmeds),2:(1+nmed
s)).
end if.
compute stderr=stderr(1:(nrow(stderr)-
clsdmy),1).
do if (ws=1 and totp=2).
compute stderr(1,1)=stddevy/sqrt(n).
end if.
compute coeffplt = coeff.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*coeff)))/dfnum).
compute coeff=coeff(1:(nrow(coeff)-clsdmy),1).
do if (totlp=1).
compute
mccoeff((nmeds+1):nrow(mccoeff))=coeff(2:(1+
nmeds),1).
end if.
compute bbbb=coeff(2,1).
do if totp = 1.
compute deco(1,1)=2+nmeds.
compute deco = deco(1:decoc,1).
compute covdirt = make((nrow(covmat)-
clsdmy),(ncol(covmat)-clsdmy),0).
compute covdirt = covmat(deco,:).
compute covdir =
make(nrow(covdirt),ncol(covdirt),0).
compute covdir = covdirt(:,t(deco)).
compute deco=coeff(deco,1).
do if (ttt > 0).
compute sedir =
sqrt(diag(directv*covdir*t(directv))).

```

```

compute directv = directv*deco.
end if.
compute sobel(:,3)=coeff(2:(1+nmeds),1).
compute sobel(:,4) = standerr(2:(1+nmeds),1).
compute sobel2 = sobel&*sobel.
do if (varorder <> 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)&*sobel2(:,4)+sobel2(:,3)&*sob
el2(:,2)).
end if.
do if (varorder = 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)&*sobel2(:,4)+sobel2(:,3)&*sob
el2(:,2)+sobel2(:,2)&*sobel2(:,4)).
end if.
compute sobel(:,1) = sobel(:,1)&*sobel(:,3).
compute sobel(:,3) = sobel(:,1)&/sobel(:,2).
compute sobel(:,4) = 2*(1-
cdfnorm(abs(sobel(:,3)))).
end if.
do if (dichy = 0).
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
do if (ws= 1 and totlp=2).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute dfden=n-1.
end if.
compute cnms = {'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute op = {coeff, standerr, tratio, p}.
end if.
do if (dichy = 1).
compute tratio = (coeff&/standerr).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute wald = tratio&*tratio.
compute cnms = {'coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute temp=coeff-abs(xp2)*standerr.
compute op = {coeff, standerr, tratio, p, temp}.
compute temp=coeff+abs(xp2)*standerr.

```

```

compute op = {op, temp}.
end if.
do if (detail = 1).
do if (totlp = 2).
print yname/title = '*****'
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome: '/format = A8.
end if.
do if (totlp <> 2).
print yname/title =
*****
*****'/rlabels = 'Outcome: '/format
= A8.
end if.
end if.
do if (dichy = 1 and bt = 1 and totp = 1).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis: '/cnames = nmsd/format = F9.2.
end if.
do if (dichy = 0).
compute r2full = 1-(sse/ssty).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute jndf=dfden.
compute wsdf=dfden.
compute xd = abs(xp2).
compute jncrit = (dfden* (exp((dfden-
(5/6))*((xd/(dfden-
(2/3)+(.11/dfden)))*(xd/(dfden-
(2/3)+(.11/dfden)))))-1)).
compute summ = {sqrt(r2full), r2full, mse,
fratio, dfnum, dfden, pfr}.
compute temp1=coeff-sqrt(jncrit)*standerr.
compute temp2=coeff+sqrt(jncrit)*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (detail = 1).
do if (ws <> 1).

```

```

print summ/title = 'Model Summary'/clabels =
'R', 'R-sq', 'MSE' 'F', 'df1', 'df2', 'p'/format =
F10.4.
end if.
end if.
end if.
do if (dichy = 1).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(coeff)-
1)).
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
do if (detail = 1).
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format =
F10.4.
end if.
end if.
do if (totlp = 2).
compute datanmy={'constant';
datanmy((nmeds+2):nrow(datanmy),1)}.
end if.
do if (detail = 1).
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
do if (ws=1).
compute op2=op(1:(1+nmeds),:).
compute datanmy2 = {'c" path'; bpathnam(
(1+(nmeds>1)): ((1+(nmeds>1))+(nmeds-
1)),1)}.
do if (totlp=2).
compute datanmy2={'c path'}.
compute op2=op(1,:).
compute wsdf=n-1.
end if.

```

```

print op2/title = 'Model'/rnames =
datanmy2/cnames = cnms/format = F10.4.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
do if (ws <> 1).
print op/title = 'Model'/rnames =
datanmy/cnames = cnms/format = F10.4.
do if (covcoeff=1).
compute
covmattp=covmat(1:nrow(op),1:nrow(op)).
compute cnamestp=t(datanmy).
print covmattp/title='Covariance matrix of
regression parameter
estimates'/rnames=datanmy/cnames=cnamest
p/format= F10.4.
end if.
end if.
end if.
do if (ttt = 0 and totlp = 1).
compute deco = op((nmeds+2),:).
do if (ws = 1).
compute deco=op(1,:).
end if.
end if.
do if (ttt = 0 and totlp = 2).
compute decotot = op(2,:).
do if (ws = 1).
compute decotot=op(1,:).
end if.
end if.
do if (nmods > 0 and model > 4 and detail = 1
and nrow(yintkey) > 1)).
print yintkey/title = 'Product terms key:'/format
= A8.
end if.
do if (nmods > 0 and model < 4 and detail = 1).
print yintkey2/title = 'Product terms key:'/format
= A8.
do if ((model = 1 or model = 2) and dichy = 0).

```

```

compute temp={((op(4,3)**2)*(1-r2full))/dfden,
op(4,3)**2, 1, dfden, op(4,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(4,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(2,1).
do if (model = 2).
compute temp={temp;((op(6,3)**2)*(1-
r2full))/dfden, op(6,3)**2, 1, dfden, op(6,4)}.
do if (hc3=1).
compute temp(2,1)=((hcnot(6,1)**2)*(1-
r2full))/dfden.
end if.
compute frat2=(dfden*(r2full-(1-
(ssem2/ssty))))/(2*(1-r2full)).
do if (hc3=1).
compute lmat=make(nrow(coeffplt),2,0).
compute lmat(4,1)=1.
compute lmat(6,2)=1.
compute frat2 =
(t(t(lmat)*coeffplt)*inv(t(lmat)*covmat*lmat)*((t(l
mat)*coeffplt)))/2).
end if.
compute temp={temp;(r2full-(1-
(ssem2/ssty))),frat2,2,dfden,1-
fcdf(frat2,2,dfden)}.
compute rnms={rnms; yintkey2(3,1),'Both'}.
end if.
print temp/title = 'R-square increase due to
interaction(s):'/rnames=rnms/clabels = 'R2-
chng', 'F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (model = 3 and dichy = 0).
compute temp={((op(8,3)**2)*(1-r2full))/dfden,
op(8,3)**2, dfden, op(8,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(8,1)**2)*(1-
r2full))/dfden.
end if.

```



```

compute rnms=yintkey2(5,1).
print temp/title = 'R-square increase due to
three-way interaction:.'/rnames=rnms/clabels =
'R2-chng', 'F(1,df2)', 'df2', 'p'/format = F10.4.
end if.
end if.
end if.
do if (model = 6 and totlp = 1).
compute
mmpaths(nrow(mmpaths),1)=coeff(nrow(mmpa
ths),1).
compute
mmpaths(nrow(mmpaths),(2:(nmeds+1)))=t(co
eff(2:(nmeds+1),1)).
end if.
do if (totlp = 1).
loop im = 1 to nmeds.
do if (model < 4).
compute ymat(1,im) = coeff(3,1)*(1-yes).
compute ymat(2,im) = coeff(4,1).
compute cmat(1,im) = covmat(3,3).
compute cmat(2,im) = covmat(4,4).
compute cmat(5,im) = covmat(3,4).
compute jnb1=coeff(3,1).
compute jnb3=coeff(4,1).
compute jnsb1=covmat(3,3).
compute jnsb3=covmat(4,4).
compute jnsb1b3=covmat(3,4).
do if (model = 2 or model = 3).
compute ymat(3,im) = coeff(6,1).
compute cmat(3,im) = covmat(6,6).
compute cmat(6,im) = covmat(3,6).
compute cmat(8,im) = covmat(4,6).
end if.
do if (model = 3).
compute ymat(4,im) = coeff(8,1).
compute cmat(4,im) = covmat(8,8).
compute cmat(7,im) = covmat(3,8).
compute cmat(9,im) = covmat(4,8).
compute cmat(10,im) = covmat(6,8).

```

```

compute jnb1=coeff(4,1).
compute jnb3=coeff(8,1).
compute jnsb1=covmat(4,4).
compute jnsb3=covmat(8,8).
compute jnsb1b3=covmat(4,8).
end if.
end if.
do if (model > 3).
compute ymat(5,im) = coeff((1+im),1).
end if.
do if (xmy = 1).
compute ymat(6,im) = coeff((2+nmeds+im),1).
end if.
do if (vy = 1).
compute ymat(6,im) = coeff((3+nmeds+im),1).
end if.
do if (qy = 1 and vy = 1).
compute ymat(6,im) = coeff((5+nmeds+((im-1)*2)),1).
compute ymat(7,im) = coeff((6+nmeds+((im-1)*2)),1).
end if.
do if (vqy = 1).
compute ymat(6,im) = coeff((6+nmeds+((im-1)*3)),1).
compute ymat(7,im) = coeff((7+nmeds+((im-1)*3)),1).
compute ymat(8,im) = coeff((8+nmeds+((im-1)*3)),1).
end if.
do if (wmy = 1).
compute ymat(7,im) = coeff((3+nmeds+im-wy),1).
end if.
do if (wmy = 1 and vy = 1).
compute ymat(7,im) =coeff((4+(nmeds*2)+im-wy),1).
end if.
do if (wmy = 1 and vy = 1 and wvmy = 1).

```

```

compute ymat(7,im) =
coeff((6+(nmeds*2)+((im-1)*2)-wy),1).
compute ymat(8,im) =
coeff((7+(nmeds*2)+((im-1)*2)-wy),1).
end if.
do if (wmy = 1 and zmy = 1).
compute ymat(6,im) = coeff((6-(wzy*3)+(wzm-
1)+(nmeds*2)+((im-1)*2)*wzm)+((im-1)*(1-
wzm))-((zy-wzy)*2),1).
compute ymat(7,im) = coeff((3-
zy+im+nmeds),1).
do if (wzmy = 1).
compute ymat(8,im) = coeff((7-
(wzy*3)+(nmeds*2)+((im-1)*2)),1).
end if.
end if.
do if (nmods > 0 and model <> 5).
loop indlp = 1 to nrow(modvals).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp)).
do if (model > 6).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp))*csum(ym
at(5:8,im)*vmat(5:8,indlp)).
end if.
end loop.
compute indboot((bt+(im-
1)*(boot+1)),:)=t(indeff).
do if (imm = 1 or imm = 3).
compute
indbootp(bt,im)=ymat(2,im)*ymat(5,im).
do if (imm=3).
compute
indbootp(bt,(im+nmeds))=ymat(3,im)*ymat(5,i
m).
do if (zdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxz-cminz).
end if.

```

```

end if.
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
end if.
end if.
do if ((imm= 2 or imm = 4) or model = 74).
compute
indbootp(bt,im)=ymat(1,im)*ymat(6,im).
do if (imm=4).
compute
indbootp(bt,(im+nmeds))=ymat(1,im)*ymat(7,i
m).
do if (qdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxq-cminq).
end if.
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
end if.
end if.
do if (imm=7).
compute
indbootp(bt,im)=ymat(2,im)*ymat(6,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(2,im)*y
mat(6,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(6,im)*onesmv1)+(ymat(2,im)*y
mat(6,im)*t(immmv1)).
do if (wdich=1).

```

```

compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxv-cminv).
end if.
end if.
do if (imm=5).
compute
indbootp(bt,im)=ymat(4,im)*ymat(5,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(4,im)*y
mat(5,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(3,im)*ymat(5,im)*onesmv1)+(ymat(4,im)*y
mat(5,im)*t(immmv1)).
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).

```

```

end if.
do if (zdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxz-
cminz).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxz-cminz).
end if.
end if.
do if (imm=6).
compute
indbootp(bt,im)=ymat(1,im)*ymat(8,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(1,im)*ymat(6,im)*onesmv2)+(ymat(1,im)*y
mat(8,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(7,im)*onesmv1)+(ymat(1,im)*y
mat(8,im)*t(immmv1)).
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxv-cminv).
end if.
do if (qdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxq-
cminq).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-

```

```

1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxq-cminq).
end if.
end if.
do if ((model = 58 or model = 59) and
wvdich=1).
compute
indbootp(bt,im)=(ymat(1,im)*ymat(7,im)*(cmax
w-cminw))+(ymat(2,im)*ymat(5,im)*(cmaxw-
cminw))+(ymat(2,im)*ymat(7,im)*((cmaxw*cma
xw)-(cminw*cminw))).
end if.
end if.
do if (model = 4 or model = 5).
compute
indboot(bt,im)=csum(ymat(1:4,im)*vmat(1:4,1
))*csum(ymat(5:8,im)*vmat(5:8,1)).
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.000000000000001.
end if.
compute sumind=rsum(indboot(bt,:)).
do if (im=nmeds).
compute
pmeff(bt,2:(im+1))=indboot(bt,:)/(sumind+coeff(
(2+nmeds),1)).
end if.
compute
rmeff(bt,(im+1))=indboot(bt,im)/coeff((2+nmeds
),1).
compute
abpseff(bt,(im+1))=indboot(bt,im)/stddevy.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*stddevx.
do if (ncovs > 0 or clsdmy > 0).
compute
abpseff(bt,(im+1))=indboot(bt,im)/sdy cov.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*sdx cov.
end if.

```

```

do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,1) = r2my-(r2full-r2xy).
compute abmmr = 1.
do if (indboot(bt,im) < 0).
compute abmmr = cmin(abmm).
end if.
do if (indboot(bt,im) > 0).
compute abmmr = cmax(abmm).
end if.
compute kappa2(bt,1)=indboot(bt,im)/abmmr.
compute tmp = indboot(bt,im)/abmmr.
end if.
end if.
end if.
do if (model = 6).
do if (nmeds = 2).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(4,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3).
compute
indboot(bt,3)=mmpaths(3,1)*mmpaths(4,3).
end if.
do if (nmeds = 3).
compute indboot(bt,1) =
mmpaths(2,1)*mmpaths(5,2).
compute indboot(bt,2) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3).
compute indboot(bt,3) =
mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4).
compute indboot(bt,4) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4).
compute indboot(bt,5) =
mmpaths(3,1)*mmpaths(5,3).
compute indboot(bt,6) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4).

```



```

compute indboot(bt,7) =
mmpaths(4,1)*mmpaths(5,4).
end if.
do if (nmeds = 4).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(6,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(6,3).
compute
indboot(bt,3)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(6,4).
compute
indboot(bt,4)=mmpaths(2,1)*mmpaths(5,2)*mmpaths(6,5).
compute
indboot(bt,5)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(6,4).
compute
indboot(bt,6)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3)*mmpaths(6,5).
compute
indboot(bt,7)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,8)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,9)=mmpaths(3,1)*mmpaths(6,3).
compute
indboot(bt,10)=mmpaths(3,1)*mmpaths(4,3)*mmpaths(6,4).
compute indboot(bt,11)
=mmpaths(3,1)*mmpaths(5,3)*mmpaths(6,5).
compute indboot(bt,12) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,13) =
mmpaths(4,1)*mmpaths(6,4).

```

```

compute indboot(bt,14) =
mmpaths(4,1)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,15) =
mmpaths(5,1)*mmpaths(6,5).
end if.
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.00000000000001.
end if.
compute
pmeff(bt,:)=indboot(bt,:)/(rsum(indboot(bt,:))+m
mpaths(nrow(mmpaths),1)).
compute rmeff(bt,:) =
indboot(bt,:)/mmpaths(nrow(mmpaths),1).
compute abpseff(bt,:)=indboot(bt,:)/stddevy.
compute
abcseff(bt,:)=(stddevx*indboot(bt,:))/stddevy.
do if (ncovs > 0 or clsdmy > 0).
compute abpseff(bt,:)=indboot(bt,:)/sdy cov.
compute
abcseff(bt,:)=(sdx cov*indboot(bt,:))/sdy cov.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,:) = r2my-(r2full-r2xy).
end if.
end if.
end if.
end loop.
end if.
end loop.
do if (savboot = 1 and bt > 1).
compute bootstrp((bt-1),:)=bootcoef.
end if.
end loop.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.

```

```

release datayed, dat, datamed, data, y, x,
datatm, dataty, mints2, cons, mnv, tmp.
do if (mc > 0).
compute x1 = sqrt(-
2*ln(uniform(mc,nrow(mcmats))))&*cos((2*3.14
159265358979)*uniform(mc,nrow(mcmats))).
compute x1=x1*chol(mcmats).
loop i=1 to nrow(x1).
compute x1(i,:)=x1(i,:)+t(mccoeff).
end loop.
loop i = 1 to nmeds.
compute
indboot(2:nrow(indboot),i)=x1(:,i)&*x1(:,(i+nme
ds)).
end loop.
end if.
do if (ttt = 0 and model > 3).
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
end if.
do if (model <> 74).
do if (dichy = 0).
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 't', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 't', 'p', 'LLCI', 'ULCI'/format =
F10.4.
else.
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 'Z', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.

```

```

print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 'Z', 'p', 'LLCI', 'ULCI' /format =
F10.4.
end if.
end if.
end if.
do if (ttt > 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
compute clbs = {modvnm2, 'Effect', 'SE', 't', 'p',
'LLCI', 'ULCI'}.
compute tratio = (directv&/sedir).
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
compute outp = {modvalsd, directv, sedir,
tratio, p}.
do if (dichy = 0).
compute temp1=directv-sqrt(jncrit)*sedir.
compute temp2=directv+sqrt(jncrit)*sedir.
compute outp = {outp, temp1, temp2}.
end if.
do if (dichy = 1).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute temp = directv-abs(xp2)*sedir.
compute outp = {outp, temp}.
compute temp = directv+abs(xp2)*sedir.
compute outp = {outp, temp}.
compute clbs = {modvnm2, 'Effect', 'SE', 'Z', 'p',
'LLCI', 'ULCI'}.
end if.
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
print outp/title = 'Conditional direct effect(s) of
X on Y at values of the moderator(s):'/format =
F10.4 /cnames = clbs.
end if.
do if (nmods > 0 and model <> 5).
do if (model < 4).

```

```

print/title =
*****
*****!

compute zmat(1,1) = 1*(1-yes).
compute cfse = make(nrow(modvals),1,0).
loop #m = 1 to nrow(modvals).
do if (model = 1).
compute zmat(2,1)=modvals(#m,1)**2.
compute zmat(5,1)=2*modvals(#m,1).
end if.
do if (model = 2 or model = 3).
compute zmat(2,1)=modvals(#m,2)**2.
compute zmat(3,1)=modvals(#m,1)**2.
compute
zmat(4,1)=(modvals(#m,1)**2)*(modvals(#m,2)
**2)*(1-yes2).
compute zmat(5,1)=2*modvals(#m,2)*(1-yes).
compute zmat(6,1)=2*modvals(#m,1)*(1-yes).
compute
zmat(7,1)=2*modvals(#m,1)*modvals(#m,2)*(1-
yes).
compute
zmat(8,1)=2*modvals(#m,1)*modvals(#m,2).
compute
zmat(9,1)=2*modvals(#m,1)*(modvals(#m,2)**
2)*(1-yes2).
compute
zmat(10,1)=2*(modvals(#m,1)**2)*modvals(#m
,2)*(1-yes2).
end if.
compute cfse(#m,1)=sqrt(csum(zmat&*cmat)).
end loop.
end if.
do if (nmods > 0).
compute clbs = {modvnm, 'Effect'}.
loop im = 1 to nmeds.
compute obs = t(indboot(1+(im-1)*(boot+1),:)).
compute outp = {modvals, obs}.
do if (model < 4).
compute tstat = obs&/cfse.

```

```

do if (dichy = 0).
compute pval = 2*(1-tcdf(abs(tstat), (n-
ncol(xy)))).
compute temp=obs-sqrt(jncrit)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp=obs+sqrt(jncrit)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 't', 'p', 'LLCI', 'ULCI'}.
compute jnclbs=clbs.
end if.
do if (dichy = 1).
compute pval = 2*(1-cdfnorm(abs(tstat))).
compute temp = obs-abs(xp2)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp = obs+abs(xp2)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute jnclbs=clbs.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indboot((1+(im-
1)*(boot+1)),:).
compute indboot2 = indboot((2+(im-
1)*(boot+1)):(1+(im-1)*(boot+1)+boot),:).
compute mnind = t(csum(indboot2)/boot).
compute stdind=t(sqrt((cssq(indboot2)-
((csum(indboot2)**2)/boot))/(boot-1))).
compute llci=make(1,ncol(indboot2),-999).
compute ulci=make(1,ncol(indboot2),-999).
loop #e = 1 to ncol(indboot2).
.
compute temp = indboot2(:,#e).
compute temp(GRADE( indboot2(:,#e) )) =
indboot2(:,#e).
compute badlo = 0.
compute badhi = 0.

```

```

do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.

```

```

do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {modvals, obs, stdind, t(llci),
t(ulci)}.
compute clbs = {modvnm, 'Effect', 'Boot SE',
'BootLLCI', 'BootULCI'}.
end if.
compute mtemp = mnames(1, im).
compute rlbs =
make(nrow(modvals),1,mnames(1,im)).
do if (model < 4).
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
do if (yes=0).
print outp/title = 'Conditional effect of X on Y at
values of the moderator(s):'/cnames =
clbs/format = F10.4.
else.
compute outp=outp(:,3:ncol(outp)).
compute clbs=clbs(:,3:ncol(clbs)).
compute clbs(1,1)='Contrast'.
print outp/title = 'Contrast of conditional effects
of X on Y'/cnames = clbs/format = F10.4.
end if.
end if.
do if (model > 5 and mod74dic <> 1).
do if (im = 1).
print/title = 'Conditional indirect effect(s) of X on
Y at values of the moderator(s):'.
end if.
print outp/title = 'Mediator'/rnames =
rlbs/cnames = clbs/format = F10.4.

```



```

else if (mod74dic = 1 and model = 74).
do if (im = 1).
print/title = 'Indirect effect(s) of X on Y:'.
compute clbs3=clbs(1,2:ncol(clbs)).
end if.
compute outp3=outp(1,2:ncol(outp)).
print outp3/title = 'Mediator'/rnames =
rlbs/cnames = clbs3/format = F10.4.
end if.
end loop.
loop i = notes to 1 by -1.
do if (note(i,1) = 4 and yes=0).
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 5 and yes=0).
print/title = 'Values for quantitative moderators
are the mean and plus/minus one SD from
mean.'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 14 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    below the mean was replaced
with the minimum because one SD below the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.
do if (note(i,1) = 15 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    above the mean was replaced
with the maximum because one SD above the
mean'/space=0.

```

```

print/title = '      is outside of the range of the
data.'/space=0.
end if.
end loop.
do if (model = 3 and yes=0).
compute jnvals=make(nrow(matw),7,0).
compute jnvals(:,1)=matw.
compute jnvals(:,2)=jnb1+jnb3*jnvals(:,1).
compute
jnvals(:,3)=sqrt(jnsb1+2*jnvals(:,1)*jnsb1b3+(jn
vals(:,1)*jnvals(:,1))*jnsb3).
compute jnvals(:,4)=jnvals(:,2)/jnvals(:,3).
do if (dichy = 0).
compute jnvals(:,5)=2*(1-tcdf(abs(jnvals(:,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(:,5)=2*(1-
cdfnorm(abs(jnvals(:,4)))).
end if.
compute jnvals(:,6)=jnvals(:,2)-
sqrt(jncrit)*jnvals(:,3).
compute
jnvals(:,7)=jnvals(:,2)+sqrt(jncrit)*jnvals(:,3).
compute clbs={clbs(:,1),clbs(:,3:ncol(clbs))}.
do if (coeffci = 0).
compute jnvals =jnvals(:,1:(ncol(jnvals)-2)).
end if.
print jnvals/title = 'Conditional effect of X*M
interaction at values of W:'/cnames
=clbs/format = F10.4.
end if.
do if (jn = 1 and (model = 1 or model = 3) and
jndich = 0 and yes=0).
print/title = '***** JOHNSON-
NEYMAN TECHNIQUE
*****'.
compute ajn =(jncrit*jnsb3)-(jnb3*jnb3).
compute bjn = 2*((jncrit*jnsb1b3)-(jnb1*jnb3)).
compute cjn = (jncrit*jnsb1)-(jnb1*jnb1).

```

```

compute radarg = (bjn*bjn)-(4*ajn*cjn).
compute den = 2*ajn.
compute nrts = 0.
do if (radarg >= 0 and den <> 0).
compute x21 = (-bjn+sqrt(radarg))/den.
compute x22 = (-bjn-sqrt(radarg))/den.
compute roots = 0.
do if (x21 >= jnmin and x21 <= jnmax).
compute nrts = 1.
compute roots = {roots; x21}.
end if.
do if (x22 >= jnmin and x22 <= jnmax).
compute nrts = nrts + 1.
compute roots = {roots; x22}.
end if.
compute roots={roots,make(nrow(roots),2,0)}.
compute modtemp=m.
do if (model=3).
compute modtemp=w.
end if.
do if (nrts > 0).
compute roots = roots(2:nrow(roots),1:3).
compute roots(1,2)=(csum(modtemp <
roots(1,1))/n)*100.
compute roots(1,3)=(csum(modtemp >
roots(1,1))/n)*100.
do if (nrow(roots)=2).
compute roots(2,2)=(csum(modtemp <
roots(2,1))/n)*100.
compute roots(2,3)=(csum(modtemp >
roots(2,1))/n)*100.
end if.
print roots/title = 'Moderator value(s) defining
Johnson-Neyman significance
region(s)'/clabels = 'Value', '% below', '%
above'/format = F10.4.
compute jnvals=make((21+nrts),7,0).
loop i= 0 to 20.
compute jnvals((i+1),1)=jnmin+(i*((jnmax-
jnmin)/20)).

```

```

end loop.
loop i = 1 to nrts.
loop j = 2 to nrow(jnvals).
do if ((roots(i,1) > jnvals((j-1),1)) and (roots(i,1)
< jnvals(j,1))).
compute
jnvals((j+1):(21+i),1)=jnvals(j:(20+i),1).
compute jnvals(j,1)=roots(i,1).
end if.
end loop.
end loop.
loop i = 1 to nrow(jnvals).
compute jnvals(i,2)=jnb1+jnb3*jnvals(i,1).
compute
jnvals(i,3)=sqrt(jnsb1+2*jnvals(i,1)*jnsb1b3+(jn
vals(i,1)*jnvals(i,1))*jnsb3).
compute jnvals(i,4)=jnvals(i,2)/jnvals(i,3).
do if (dichy = 0).
compute jnvals(i,5)=2*(1-tcdf(abs(jnvals(i,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(i,5)=2*(1-
cdfnorm(abs(jnvals(i,4)))).
end if.
compute jnvals(i,6)=jnvals(i,2)-
sqrt(jncrit)*jnvals(i,3).
compute
jnvals(i,7)=jnvals(i,2)+sqrt(jncrit)*jnvals(i,3).
end loop.
do if (model = 1).
print jnvals/title = 'Conditional effect of X on Y
at values of the moderator (M)'/cnames
=jnclbs/format = F10.4.
end if.
do if (model = 3).
compute
jnclbs={jnclbs(:,1),jnclbs(:,3:ncol(jnclbs))}.

```

```

print jnvals/title = 'Conditional effect of X*M on
Y at values of the moderator (W)'/cnames
=jnclbs/format = F10.4.
end if.
end if.
do if (nrts = 0).
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
else.
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
end if.
end if.
do if (model < 4 and (plot = 1)).
compute dataplot =
make((nrow(modvals)*nrow(matx)),(ncol(modv
als)+1),0).
compute tmp = 1.
loop i = 1 to nrow(modvals).
loop j = 1 to nrow(matx).
compute dataplot(tmp,.)={matx(j,1),
modvals(i,:)}.
compute tmp=tmp+1.
end loop.
end loop.
compute dataplot = {dataplot,
make(nrow(dataplot),(1+dichy),0)}.
compute dataplo2 = make(nrow(dataplot),1,1).
do if (model = 1).
compute dataplo2 = {dataplo2, dataplot(:,2),
dataplot(:,1), (dataplot(:,1)*dataplot(:,2))}.
end if.
do if (model = 2 or model = 3).
compute dataplo2 = {dataplo2, dataplot(:,3),
dataplot(:,1), (dataplot(:,1)*dataplot(:,3)),
dataplot(:,2), (dataplot(:,1)*dataplot(:,2))}.

```

```

do if (model = 3).
compute dataplo2 = {dataplo2,
(dataplot(:,2)*dataplot(:,3)),
(dataplot(:,1)*dataplot(:,2)*dataplot(:,3))}.
end if.
end if.
loop i = 1 to nrow(dataplot).
compute tmp=dataplo2(i,:).
do if (ncovs > 0).
compute tmp = {tmp, covmeans}.
end if.
do if (cluster > 0).
compute tmp = {tmp, cldmeans}.
end if.
compute dataplot(i,(ncol(dataplot)-
(dichy)))=tmp*coeffplt.
do if (dichy = 1).
compute
dataplot(i,(ncol(dataplot)))=exp(tmp*coeffplt)/(1
+exp(tmp*coeffplt)).
end if.
end loop.
compute clbs = {xname, modvnm, 'yhat'}.
do if (dichy = 1).
compute clbs = {xname, modvnm, 'ln(odds)',
'prob'}.
end if.
print/title =
*****
*****!

```

```

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (model = 1).
do if (dichy = 0).
print/title = 'DATA LIST FREE/Opra_Pract
Technological_Advan Financial_Performance.'
/space=1.
else if (dichy = 1).
print/title = 'DATA LIST FREE/Opra_Pract
Technological_Advan logodds prob.'
/space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
Financial_Performance BY
Technological_Advan.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH Financial_Performance BY
Opra_Pract.' /space=0.
end if.
end if.

```

```

do if (dichy = 1).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
logodds BY Technological_Advan.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
prob BY Technological_Advan.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Opra_Pract.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Opra_Pract.' /space=0.
end if.
end if.
end if.
do if (model = 2 or model = 3).
do if (dichy = 0).
print/title = 'DATA LIST FREE/Opra_Pract
Technological_Advan Financial_Performance.'
/space=1.
end if.
do if (dichy = 1).
print/title = 'DATA LIST FREE/Opra_Pract
Technological_Advan logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' /format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
Financial_Performance BY
Technological_Advan/PANEL ROWVAR='
/space=0.
end if.

```



```

do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH Financial_Performance BY
Opra_Pract/PANEL ROWVAR=' /space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
logodds BY Technological_Advan/PANEL
ROWVAR=' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
prob BY Technological_Advan/PANEL
ROWVAR=' /space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Opra_Pract/PANEL
ROWVAR=' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Opra_Pract/PANEL
ROWVAR=' /space=0.
end if.
end if.
end if.
do if (ncovs > 0).
print/title = '* Estimates are based on setting
covariates to their sample means.'.
end if.
end if.
end if.
do if ((imm > 0) or (model = 74 and
mod74dic=0) or ((model = 58 or model = 59)
and wvdich=1))).
do if (imm > 4 and imm < 8).

```

```

compute
indbootp={indbootp(:,1:nmeds),indboop1,indbo
op2}.
end if.
compute obsprod = t(indbootp(1,:)).
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indbootp(1,:).
compute indbootp = indbootp(2:(boot+1),:).
compute mnindp = t(csum(indbootp)/boot).
compute stdindp=t(sqrt((cssq(indbootp)-
((csum(indbootp)&*2)/boot))/(boot-1))).
compute llcip=make(1,ncol(indbootp),-999).
compute ulcip=make(1,ncol(indbootp),-999).
loop #e = 1 to ncol(indbootp).

.
compute temp = indbootp(:,#e).
compute temp(GRADE( indbootp(:,#e) )) =
indbootp(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).

```

```

compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcip(1,#e)=llcit.
compute ulcip(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {obsprod, stdindp, t(llcip),
t(ulcip)}.
compute clbs = {'Effect', 'SE(Boot)', 'BootLLCI',
'BootULCI'}.
do if (model = 8 or model = 12).
print/title='-----'.
compute outpimm=outp(1:nmeds,:).

```

```

print/title = 'Indirect effect of highest order
product:.'/space=0.
print outpimm/title = 'Mediator'/cnames =
clbs/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
compute clbs3 = {'Index', 'SE(Boot)',
'BootLLCI', 'BootULCI'}.
do if (imm < 5).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=outp((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
do if (imm > 4 and imm < 8).

```

```

print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'.

compute outpimm=outp(1:nmeds,:).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
compute
outpimm=outp((nmeds+1):nrow(outp),:).
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.

loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),clbs3}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3(1,1)=modvnm(1,1).
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.

```

```

print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
end if.
do if (boot = 0).
do if (model = 8 or model = 12).
print/title='-----'.
compute obsimm=obsprod(1:nmeds,:).
print/title = 'Indirect effect of highest order
product.'/space=0.
print obsimm/title = 'Mediator'/clabels =
'Effect'/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
do if ((imm > 0 and imm < 5) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=obsprod((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
end loop.
end if.

```

```

do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'.
compute outpimm=obsprod(1:nmeds,1).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
compute
outpimm=obsprod((nmeds+1):nrow(obsprod),:)
.
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),'Index'}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3={modvnm(1,1),'Index'}.
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator: '/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
end if.
end if.

```

```

end if.
compute conmake=0.
compute concols = 0.
do if ((model > 3 and model < 7) and (contrast
> 0) and nmods = 0 and nmeds > 1 ).
compute concols =
((ncol(indboot)*(ncol(indboot)-1))/2).
compute indcon=make(nrow(indboot),concols,-
999).
compute conkey = { ' ', ' ', ' ', ' ' }.
compute temp=1.
compute conmake=1.
loop i = 1 to (ncol(indboot)-1).
loop j = (i+1) to (ncol(indboot)).
compute indcon(:,temp)=indboot(:,i)-
indboot(:,j).
do if (contrast = 2).
compute indcon(:,temp)=abs(indboot(:,i))-
abs(indboot(:,j)).
end if.
do if (model <= 6).
compute conkey={conkey; mnames(1,i),
'minus', mnames(1,j)}.
end if.
do if (model = 6).
compute conkey={conkey; indlbl2(i,1), 'minus',
indlbl2(j,1)}.
end if.
compute temp=temp+1.
end loop.
end loop.
end if.
do if (model = 4 or model = 5).
compute clbs = {'Effect'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obs = {csum(obs); obs}.
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.

```



```

compute rlbs = {rlbs;
cntname(1:ncol(indcon),1)}.
end if.
compute outp = obs.
compute outp2=outp.
do if (effsize = 1 and dichy = 0).
compute
pmeff(:,1)=rsum(pmeff(:,2:ncol(pmeff))).
compute
rmeff(:,1)=rsum(rmeff(:,2:ncol(rmeff))).
compute
abpseff(:,1)=rsum(abpseff(:,2:ncol(abpseff))).
compute
abcseff(:,1)=rsum(abcseff(:,2:ncol(abcseff))).
compute eff = {pmeff, rmeff, abpseff, abcseff}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute eff = {eff, r245}.
compute r245obs = {r245(1,1);r245(1,1)}.
compute kappa2ob = kappa2(1,1).
end if.
compute pmobs = t(pmeff(1,1:(nmeds+1))).
compute rmobs = t(rmeff(1,1:(nmeds+1))).
compute psobs = t(abpseff(1,1:(nmeds+1))).
compute csobs = t(abcseff(1,1:(nmeds+1))).
do if (contrast = 0).
compute outp2 = {obs, psobs, csobs, pmobs,
rmobs}.
end if.
do if (contrast > 0).
compute obs2=obs(1:nrow(psobs),:).
compute outp2 = {obs2, psobs, csobs, pmobs,
rmobs}.
end if.
compute clbs = {'ab', 'ab_ps', 'ab_cs', 'ab/c',
'ab/c'}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute outp2 = {outp2, r245obs}.
compute clbs = {clbs, 'R-sq_med'}.

```

```

do if (boot = 0 and mc = 0).
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
do if (boot = 0 and mc = 0).
do if (nmeds = 1).
compute outp2 = outp2(2,:).
compute rlbs = rlbs(2,1).
end if.
do if (model=5).
compute outp2=outp2(:,1:3).
end if.
print outp2/title = 'Indirect effect(s) of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1 and nmeds >
1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0 or mc > 0).
compute indboot = {rsum(indboot), indboot}.
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
compute ones = make(bootsz,1,1).
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(bootsz+1),:).
compute mnind = t(csum(indboot)/bootsz).

```

```

compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/bootSZ))/(bootSZ-1))).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).

.

compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.

```

```

end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llici(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
do if (effsize = 1 and dichy = 0 and boot > 0).
compute estmte=eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).
.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.

```

```

compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).

```

```

compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
end if.
end if.
do if (boot > 0 or mc > 0).
compute outp = {obs, stdind, t(llci), t(ulci)}.
do if (nmeds = 1).
compute outp = outp(2,:).
compute rlbs = rlbs(2,1).
end if.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
do if (mc > 0).
compute clbs = {'Effect', 'MC SE', 'MC LLCI',
'MC ULCI'}.
end if.
print outp/title = 'Indirect effect of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (dichy = 0 and effsize = 1 and boot > 0).
compute outp = {psobs,
stdindf((2*(nmeds+1)+1):(3*(nmeds+1)),1),
t(llcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))),
t(ulcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Partially standardized indirect
effect of X on Y'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute outp = {csobs,
stdindf((3*(nmeds+1)+1):(4*(nmeds+1)),1),
t(llcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))),
t(ulcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.

```

```

print outp/title = 'Completely standardized
indirect effect of X on Y'/rnames = rlbs/cnames
= clbs/format = F10.4.
compute outp = {pmobs,
stdindf(1:(nmeds+1),1),t(llcif(1,1:(nmeds+1))),
t(ulcif(1,1:(nmeds+1))) }.
do if (model = 4).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
compute outp = {rmobs, stdindf(((nmeds+1)+1)
:(2*(nmeds+1)) ,1),
t(llcif(1,((nmeds+1)+1):(2*(nmeds+1)))),
t(ulcif(1,((nmeds+1)+1):(2*(nmeds+1))))}.
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
do if (nmeds = 1 and cluster = 0 and ncovs = 0
and model = 4).
compute r245obs = r245obs(1,1).
compute outp = {r245obs,
stdindf((4*(nmeds+1)+1):(4*(nmeds+1)+1),1),
t(llcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1))),
t(ulcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1)))}.
print outp/title = 'R-squared mediation effect
size (R-sq_med)'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
end if.
do if (normal = 1).

```

```

do if (nmeds = 1).
print sobel/title = 'Normal theory tests for
indirect effect'/clabels = 'Effect', 'se', 'Z',
'p'/format = F10.4.
end if.
do if (nmeds > 1).
compute rlbs2 = rlbs(2:nrow(rlbs),1).
print sobel/title = 'Normal theory tests for
specific indirect effects'/rnames = rlbs2/clabels
= 'Effect', 'se', 'Z', 'p'/format = F10.4.
end if.
end if.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
do if (model = 6).
compute clbs = {'eff'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obsfsum=csum(obs).
compute obs = {obsfsum; obs}.
compute indlbl = indlbl(1:nrow(obs),1).
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute indlbl = {indlbl;
cntname(1:ncol(indcon),1)}.
end if.
compute obs2=obs.
do if (boot = 0).
do if (dichy = 0 and effsize = 1).
compute clbs = {'eff', 'eff_ps', 'eff_cs', 'eff/c',
'eff/c'}.
do if (ncovs = 0).

```



```

compute obs = {obs, obs/stddevy,
(obs*stddevx/stddevy), obs/ctot,
obs/mmpaths(nrow(mmpaths),1)}.
end if.
do if (ncovs > 0 or clsdmy > 0).
compute obs = {obs, obs/sdycov,
(obs*sdxcov/sdycov),
obs/(obsfsum+mmpaths(nrow(mmpaths),1)),
obs/mmpaths(nrow(mmpaths),1)}.
end if.
compute obs2=obs.
do if (contrast > 0).
compute obs2 = obs(1:(nrow(obs)-concols),:).
end if.
end if.
print obs2/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute indboot = {rsum(indboot), indboot}.
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(boot+1),:).
compute mnind = t(csum(indboot)/boot).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/boot))/(boot-1))).
compute temp = nrow(indboot).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).

```

```

loop #e = 1 to ncol(indboot).

.

compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.

```

```

do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute obs = {obs, stdind, t(llci), t(ulci)}.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
print obs/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (effsize = 1 and dichy = 0).
compute indboot=indboot(:,1:(ncol(indboot)-
concols)).
compute eff = {rsum(abpseff), abpseff,
rsum(abcseff), abcseff, rsum(pmeff), pmeff,
rsum(rmeff), rmeff}.
compute effobs = eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).
.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).

```

```

compute badlo = 0.
compute badhi = 0.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(effobs(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.

```

```

compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute temp2 = stdindf(1:ncol(indboot),1).
compute temp3 = effobs(:,1:ncol(indboot)).
compute templo = llcif(1,1:ncol(indboot)).
compute temphi = ulcif(1,1:ncol(indboot)).
compute outp = {t(temp3), temp2,t(templo),
t(temphi)}.
print outp/title = 'Partially standardized indirect
effect of X on Y'/cnames = clbs/rnames =
indlbl/format = F10.4.
compute temp2 =
stdindf((ncol(indboot)+1):(2*ncol(indboot)),1).
compute temp3 =
effobs(:,(ncol(indboot)+1):(2*ncol(indboot))).
compute templo =
llcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute temphi =
ulcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templo),
t(temphi)}.
print outp/title = 'Completely standardized
indirect effect of X on Y'/cnames = clbs/rnames
= indlbl/format = F10.4.
compute temp2 =
stdindf((2*(ncol(indboot))+1):(3*ncol(indboot)),1
).
compute temp3 =
effobs(:,(2*(ncol(indboot))+1):(3*ncol(indboot)))
.

```

```

compute templow =
  llcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute temphi =
  ulcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
  t(temphi)}.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
compute temp =
  eff(:,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temp2 =
  stdindf((3*(ncol(indboot))+1):(4*ncol(indboot)),1
  ).
compute temp3 =
  effobs(:,(3*(ncol(indboot))+1):(4*ncol(indboot)))
  .
compute templow =
  llcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temphi =
  ulcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
  t(temphi)}.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
end if.
end if.
do if (nmeds = 2).
compute effkey = {xname, '->', mnames(1,1), '-
->', yname, '', ''}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, '', ''}.
compute effkey = {indlbl(2:4,1), effkey}.
end if.
do if (nmeds = 3).
compute effkey = {xname, '->' , mnames(1,1), '-
->', yname, '', '', ''}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, '',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, '',' ',' ',' '}.
compute effkey = {indbl(2:8,1), effkey}.
end if.
do if (nmeds = 4).
compute effkey = {xname, '->' , mnames(1,1), '-
>', yname, '',' ',' ',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,4), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname, '',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,4), '->', yname, '',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, '',' '}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', mnames(1,4), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,4), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', mnames(1,4), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,4), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {indl(2:16,1), effkey}.
end if.
print effkey/title = 'Indirect effect key'/format =
A8.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
end if.
DO IF (medte2=1).
compute
name=t(mdlnms2(2:nrow(mdlnms2),1)).
compute tmpdat=datayed(:,2:ncol(datayed)).
compute dd=tmpdat(:,1).

```



```

compute
dd={dd,tmpdat(:,(2+nmeds)),tmpdat(:,2:(1+nmeds))}.
compute stratify=0.
do if (ncovs > 0).
compute cname = cnames.
compute
dd={dd,tmpdat(:,(3+nmeds):(2+nmeds+ncovs))}.
end if.
compute method = 1.
compute samples = boot.
do if (mc > 0).
compute samples = mc.
compute method=2.
end if.
compute omnibus = 1.
compute booterr = 0.
do if (mcx < 0 or mcx > 4).
compute mcx = 1.
end if.
compute ovals = ncol(design(dd(:,1))).
compute nte = notes.
compute criterr = 0.
compute xskip = 0.
compute dumok = 0.
compute badend = 0.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute priorlo = -9999999.
compute priorhi = 9999999.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute reps = samples.
do if (contrast = 1).
compute note(nte,1) = 26.
compute nte=nte+1.

```

```

end if.
do if (effsize = 1).
compute note(nte,1) = 27.
compute nte=nte+1.
end if.
do if (normal = 1).
compute note(nte,1) = 28.
compute nte=nte+1.
end if.
do if (method = 2).
do if (omnibus = 1).
compute note(nte,1) = 25.
compute nte=nte+1.
end if.
compute samples = 0.
do if (reps < 1000).
compute reps = 1000.
end if.
end if.
do if (method = 1).
compute samples = reps.
end if.
compute temp = ncol(dd).
compute n = nrow(dd).
compute ones = make(n,1,1).
compute temp=dd(:,2).
compute nxd = ncol(temp).
compute nx = nxd.
do if (nx = 1 and mcx > 0).
compute temp = dd.
compute temp(GRADE(dd(:,2)),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,2)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute mnvls = cmin(t(nnvls)).
do if (mnvls < 2).
compute errs = errs + 1.
compute runerrs(errs,1) = 37.
compute criterr = 1.

```

```

end if.
do if (nvls > 9).
compute errs = errs+1.
compute runerrs(errs,1) = 36.
compute criterr = 1.
end if.
do if (criterr = 0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,2).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,2) <> nnvls((temp-1),1)).
compute nnvls(temp,1)=dd(i,2).
compute temp = temp+1.
end if.
end loop.
do if (mcx > 0).
compute x = dummy(:,2:ncol(dummy)).
compute nx = ncol(x).
compute minus1 = make(1,ncol(x),-1).
compute xdes=make((nx+1),3,0).
compute xdes(1,1)=dd(1,2).
compute xdes(1,2)=1.
compute temp = 2.
loop k = 2 to n.
do if (dd(k,2) <> dd((k-1),2)).
compute xdes(temp,2) = k.
compute xdes(temp,1) = dd(k,2).
compute xdes((temp-1),3) = k-1.
compute temp=temp+1.
end if.
end loop.
compute xdes((temp-1),3)=n.
compute xdes = {xdes, (xdes(:,3)-xdes(:,2)+1)}.
do if (stratify <> 0).
compute note(nte,1) = 7.
compute nte=nte+1.
end if.
do if (mcx = 4).

```

```

loop k = 1 to n.
do if (rsum(x(k,:)) = 0).
compute x(k,:) = minus1.
end if.
end loop.
end if.
do if (mcx = 2 or mcx = 3).
compute note(nte,1)=4.
compute nte = nte+1.
loop k = 1 to n.
do if (rsum(x(k,:)) > 0).
loop i = 1 to ncol(x).
do if (x(k,i) = 0).
compute x(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcx = 3).
compute conmat1={-8,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1; 0,0,0,0,0,-
3,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.
loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,:)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute x(k,:)=conmat1((rsum(x(k,:))+1),:).
end loop.
end if.
end if.
compute xdata = x.
compute xskip = 1.
compute xname = ddd(1,1:nx).
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).

```

```

compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcx = 4).
compute dummat(1,:) = minus1.
end if.
do if (mcx = 2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcx = 3).
compute dummat=conmat1.
end if.
compute dummat={nnvls, dummat}.
compute xname2={name(1,2), xname}.
end if.
end if.
compute nm=nmeds.
compute r2b = make(nm,1,-999).
compute indeff = make((nx*nm),1,-999).
compute cov =
make((nx*nm+nm),(nx*nm+nm),0).
compute mns=make(nrow(cov),1,0).
compute ny = ncol(dd)-nm-nxd-ncovs.
compute y = dd(:,1).
compute ydata = y.
compute yname = name(1,1).
do if (xskip = 0).
compute x = dd(:,2:(1+nxd)).
compute xdata = x.
compute xname = name(1,2:(1+nxd)).
compute indlbs = t(xname).
end if.
compute m = dd(:,(2+nxd):(1+nxd+nm)).
compute mdata = m.

```

```

compute mlab = { ' M1 = ';' ' M2 = ';' ' M3 = ';' '
M4 = ';' ' M5 = ';' ' M6 = ';' ' M7 = ';' ' M8 = ';' '
M9 = ';' ' M10 = '}.
compute mlab = {mlab; ' M11 = ';' ' M12 = ';' '
M13 = ';' ' M14 = ';' ' M15 = '}.
compute mname =
name(1,(2+nxd):(1+nxd+nm)).
do if (ncovs > 0).
compute c =
dd(:,(2+nxd+nm):(1+nxd+nm+ncovs)).
compute x = {x, c}.
compute xdata = x.
compute cdata = c.
compute xname = {xname, cname}.
end if.
compute nms={yname; t(mname);
t(xname(1,1:(ncol(xname)-ncovs)))}.
do if (dumok = 1).
compute nms={yname; t(mname); name(1,2)}.
end if.
do if (nm < 16).
do if (dumok = 1).
print dummat/title = 'Coding of categorical X
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
compute
boots=make((samples+1),((nx+1)*nm),-999).
do if (errs = 0 and criterr = 0).
loop k = 1 to (samples+1).
do if (k = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef=0.
do if (k > 1).
compute u=trunc(uniform(n,1)*n)+1.
compute xdata = x(u,:).
compute mdata = m(u,:).

```

```

compute ydata = y(u,:).
do if (ncovs > 0).
compute cdata = c(u,:).
end if.
end if.
loop i = 1 to nm.
compute start = 1+(i-1)*nx.
compute xmat = {ones, xdata}.
do if (covmy=2).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = mdata(:,i).
compute amat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*amat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-(((1-r2)*(n-1))/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute r2cha = r2.
compute adjr2cha = adjr2.
do if (ncovs > 0 and covmy <> 2).
compute xmat = {ones, cdata}.
compute atmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*atmat)**2).
compute r2step1 = (sstotal-ssresid2)/sstotal.
compute r2cha = r2-r2step1.
compute adjr2cha = adjr2- (1-(((1-r2step1)*(n-
1))/(n-ncol(xmat)))).
compute xmat = {ones, xdata}.
end if.
compute bootcoef={bootcoef,t(amat)}.
compute r2b(i,1)=adjr2cha.
do if (k = 1).

```

```

compute hcinvtX=inv(t( xmat )' xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,1)*hcinvtX*t( xmat
(i3,1)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvtX*t( xmat )' xmat
*hcinvtX
.
compute tratio = amat&/sqrt(diag(covmat)).
compute rnms = mname(1,i).
print rnms/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.
compute lmat = ident(nrow(amat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*amat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*amat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/'format = F10.4
/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'.

```



```

compute op = {amat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*(xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute cov(start:(start+(nx-
1)),start:(start+(nx-
1)))=covmat(2:(ncol(covmat)-(ncovs*(covmy <>
2))),2:(ncol(covmat)-(ncovs*(covmy <> 2)))).
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns(start:(start+(nx-
1)),1)=amat(2:(1+nx)).
end loop.
compute xmat = {ones,mdata,xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (k = 1).
print yname/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.

```

```

do if (ovals = 2).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:'/cnames = nmsd/format = F9.2.
end if.
end if.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(yamat)/n)).
compute LL3 = yamat*&ln(pt2)+(1-yamat)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= yamat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).

```

```

compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute bootcoef={bootcoef,t(bmat)}.
compute LL1 = LL2.
compute covmat=varb.
end if.
do if (ovals <> 2).
compute bmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*bmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-(((1-r2)*(n-1))/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute bootcoef={bootcoef,t(bmat)}.
end if.
do if (k = 1).
do if (ovals = 2).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.

```

```

compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).
compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms2/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute
opdirsum=outp((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
end if.
do if (ovals <> 2).
.
compute hcinvXtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtXt*( xmat
(i3,:)).

```

```

end loop.
loop i3=1 to k3.
compute xmat(:,i3) = ( resid(:,ncol( resid
))&/(1-h))&* xmat(:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat(:,i3) = sqrt( mse )&* xmat(:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(bmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/(ncol(xmat)-1).
compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat((2+nmeds):(2+nmeds+(nx-1)),:)=
lmat2.
compute fratiodr =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/nx.
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute df2=(n-ncol(xmat)).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), df2, pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = bmat&/sqrt(diag(covmat)).
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute op = {bmat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat)))) }.

```

```

compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp))))*(xp2/(temp-
(2/3)+(.11/temp)))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms2/format =
F10.4.
compute
opdirsum=op((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
compute xmat = {ones,mdata}.
do if (ncovs > 0).
compute xmat = {xmat, cdata}.
end if.
compute btmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*btmat)**2).
compute r2direct = r2-((sstotal-
ssresid2)/sstotal).
compute pfr = 1-fcdf(fratiodr,(nx),(n-1-nx-nm-
ncovs)).
compute r2direct = {r2direct, fratiodr, nx, (n-1-
nx-nm-ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms2/cnames=rnms2/form
at= F10.4.
end if.
end if.

```

```

do if (k = 1 and toteff = 1).
print yname/title = '*****'
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:.'/format = A8.
compute xmat = {ones, xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(ymat)/n)).
compute LL3 = ymat*&ln(pt2)+(1-ymat*)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= ymat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.

```

```

compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute LL1 = LL2.
compute covmat=varb.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(bmat)-
1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms = {'Constant'; t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).

```



```

compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute optotsum=outp(2:(1+nx),:).
end if.
do if (ovals <> 2).
compute cmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*cmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).

.

compute hcinvXtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).

```

```

loop i3=1 to k3.
compute xmat(:,i3) = sqrt( mse )&* xmat(:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(cmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = cmatrix/sqrt(diag(covmat)).
compute op = {cmatrix, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*(xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute optotsum=op(2:(1+nx),:).
compute ssresid2 = ssresid.
compute r2total = r2.

```

```

compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat(2:(1+nx),:) = lmat2.
compute fratio3 =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/nx.
do if (ncovs > 0).
compute xmat = {ones, cdata}.
compute ctmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*ctmat)**2).
compute r2total = r2-((sstotal-ssresid2)/sstotal).
end if.
compute pfr = 1-fcdf(fratio3,(nx),(n-1-nx-
ncovs)).
compute r2tot2 = {r2total, fratio3, nx, (n-1-nx-
ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns((nrow(mns)-
nm+1):nrow(mns),1)=bmat(2:(1+nm),1).
compute tmp = 1.
loop i = 1 to (nm*nx).
compute
boots(k,i)=mns(i,1)*mns(((nm*nx)+tmp),1).
do if ((i/nx)=trunc(i/nx)).
compute
boots(k,((nm*nx)+tmp))=r2b(tmp,1)*mns(((nm*
nx)+tmp),1).
compute tmp = tmp+1.
end if.
end loop.
do if (savboot = 1 and k > 1).

```

```

compute bootstrp((k-1),:)=bootcoef.
end if.
end loop.
do if (method = 1).
compute llci=make(1,ncol(boots),-999).
compute ulci=make(1,ncol(boots),-999).
compute indeff = t(boots(1,1:(nm*nx))).
compute indeff22=t(boots(1,:)).
compute omni = t(boots(1,((nm*nx)+1) :
ncol(boots))).
do if (nrow(boots) > 1).
compute boots = boots(2:nrow(boots),:).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
loop i = 1 to ncol(boots).
.
compute temp = boots(:,i).
compute temp(GRADE( boots(:,i) )) = boots(:,i).
compute badlo = 0.
compute badhi = 0.
do if ( indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(indeff22(i,1)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.

```

```

do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,i)=llcit.
compute ulci(1,i)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute llci = t(llci).
compute ulci= t(ulci).
compute llci={llci,ulci}.
end if.
end if.
do if (method = 2).
compute tmp=1.
loop i = 1 to (nm*nx).
compute
indeff(i,1)=mns(i,1)&*mns(((nm*nx)+tmp),1).

```

```

do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = sqrt(-
2*ln(uniform(reps,nrow(cov))))&*cos((2*3.1415
9265358979)*uniform(reps,nrow(cov))).
compute boots=boots*chol(cov).
loop i = 1 to ncol(boots).
compute boots(:,i)=boots(:,i)+mns(i,1).
end loop.
compute tmp=1.
loop i = 1 to (nm*nx).
compute
boots(:,i)=boots(:,i)&*boots(:,((nm*nx)+tmp)).
compute temp = boots(:,i).
compute temp(GRADE(boots(:,i))) = boots(:,i).
compute boots(:,i) = temp.
do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = boots(:,1:(nm*nx)).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
compute llci =
t(((boots(cilow,:));(boots(cihigh,:)))).
end if.
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
compute rnmssum=rnms(2:(1+nx),1).
do if (ovals <> 2).
print optotsum/title = 'Relative total effects of X
of Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.

```

```

print r2tot2/title = 'Omnibus test of total effect of
X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print optotsum/title = 'Relative total effects of X
on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
end if.
compute
rnmssum=rnms2((2+nmeds):(1+nmeds+nx),1).
do if (ovals <> 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.
print r2direct/title = 'Omnibus test of direct
effect of X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
do if (omnibus = 1 and (method = 1 or method
= 3)).
compute indlbs = {indlbs; 'Omnibus'}.
end if.
loop i = 1 to nm.
compute op = mname(1,i).
print op/title = 'Relative indirect effect(s) of X on
Y through:.'/format = A8/space=0.
do if (method = 2).
compute clbs = {'Effect', 'SE MC', 'MC LLCI',
'MC ULCI'}.
compute op = {indeff, bootse(1:(nm*nx)), lici}.
end if.

```

```

do if (method = 1).
compute clbs = {'Effect', 'SE(boot)', 'LLCI',
'ULCI'}.
compute op = indeff.
do if (samples > 0).
compute op = {indeff, bootse(1:(nm*nx),:),
llci(1:(nm*nx),:)}).
end if.
end if.
compute temp = op((((i-1)*nx)+1):(nx*i),:).
do if ((method = 1) and omnibus = 1).
do if (samples = 0).
compute temp={temp;omni(i,1)}.
end if.
do if (samples > 0).
compute temp = {temp; omni(i,1),
bootse(((nm*nx)+i),:), llci(((nm*nx)+i),:)}).
end if.
end if.
print temp/title = ' '/rnames = indlbs/cnames =
clbs/format = F10.4 /space=0.
print/title = '-----'.
end loop.
end if.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
END IF.
DO IF (mdpbe=1).
compute nm={yname,xname,mnames}.
do if (ncovs > 0).

```



```

compute
nm={yname,cnames,xname,mnames}.
end if.

compute tpx=xname.
compute x = dd(:,2:ncol(dd)).
compute tpy = yname.
compute tempy= dd(:,1).
compute n = nrow(dd).
compute nx=1.
compute criterr=0.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute ddd1 = {'int_1', 'int_2', 'int_3', 'int_4',
'int_5', 'int_6', 'int_7', 'int_8', 'int_9'}.
compute dumok = 0.
compute mcfoc=mcx.
compute mcmmod=mcm.
compute mcloc=(mcfoc>0).
compute con = make(n,1,1).
compute ncovs=ncol(x)-2.
do if (mcfoc > 0 or mcmmod > 0).
compute temp = dd.
compute temp(GRADE(dd(:,ncol(dd)-
mcloc))),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,ncol(dd)-
mcloc)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute toosmall=rsum(nnvls < 2).
compute mnvls = cmin(t(nnvls)).
do if ((rsum(nnvls < 2)) > 0).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=37.
end if.
do if (nvls > 10).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=36.

```

```

end if.
do if (criterr=0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,ncol(dd)-mcloc).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,ncol(dd)-mcloc) <> nnvls((temp-
1),1)).
compute nnvls(temp,1)=dd(i,ncol(dd)-mcloc).
compute temp = temp+1.
end if.
end loop.
compute dummy = dummy(:,2:ncol(dummy)).
do if (mcfoc=4 or mcmmod=4).
compute minus1=make(1,ncol(dummy),-1).
loop k = 1 to n.
do if (rsum(dummy(k,:)) = 0).
compute dummy(k,:)=minus1.
end if.
end loop.
end if.
do if (mcfoc = 2 or mcfoc = 3) or (mcmmod = 2 or
mcmmod = 3)).
loop k = 1 to n.
do if (rsum(dummy(k,:)) > 0).
loop i = 1 to ncol(dummy).
do if (dummy(k,i) = 0).
compute dummy(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcfoc = 3 or mcmmod=3).
compute conmat1={-8,1,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1,1; 0,0,0,0,0,-
3,1,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.

```

```

loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,:)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute
dummy(k,:)=conmat1((rsum(dummy(k,:))+1),:).
end loop.
end if.
end if.
compute nx = ncol(dummy).
compute xname = ddd(1,1:nx).
compute xdata = dummy.
compute xname = ddd(1,1:nx).
compute xname2={nm(1,(ncol(nm)-mcloc)),
xname}.
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcfoc = 2 or mcmmod=2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcfoc = 3 or mcmmod=3).
compute dummat=conmat1.
end if.
do if (mcfoc = 4 or mcmmod = 4).
compute dummat(1,:)=minus1.
end if.
compute dummat={nnvls, dummat}.
end if.
end if.
do if (criterr=0).
compute y = dd(:,1).
compute ovals = ncol(design(dd(:,1))).

```

```

compute itprob = 0.
do if (ovals = 2).
compute omx = cmax(y(:,1)).
compute omn = cmin(y(:,1)).
compute y(:,1) = (y(:,1) = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute sstotal = csum((y-(csum(y)/n))&**2).
compute outv = t(nm(1,1)).
compute mdtr = nm(1,ncol(dd)).
compute fciv = nm(1,(ncol(dd)-1)).
do if (mcfoc <> 0 or mcmod <> 0).
compute inter=make(n,nx,0).
loop i = 1 to nx.
compute inter(:,i)=dummy(:,i)&*dd(:,ncol(dd)-
(1-mcloc)).
end loop.
do if (ncovs = 0).
compute x={con,dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
do if (ncovs > 0).
compute
x={con,dd(:,2:(1+ncovs)),dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={t(nm(1,2:(1+ncovs)));
nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
end if.
compute xmns = csum(x)/n.
compute focvals = {1,0,0}.
compute highwarn = 0.
compute lowwarn = 0.

```

```

do if (ncol(x) > (2*nx+2)).
compute covmns = {1, xmns(1,2:(ncol(x)-1-
(2*nx)))}.
compute focvals = {covmns,0,0}.
end if.
compute dfres = n-ncol(x).
do if (ovals = 2).
compute pt2 = make(n,1,(csum(y)/n)).
compute LL3 = y*&ln(pt2)+(1-y)*&ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(x),1,0).
compute LL1 = 0.

.

compute pt1lp= pt1.
compute xlp= x.
compute ylp= y.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)*&ln(1-
pt1lp).

```

```

compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
else if (ovals <> 2).
compute invXtX=inv(t(x)*x).
compute b = invXtX*t(x)*y.
compute resid = y-(x*b).
compute ssresid=csum(resid&*resid).
compute msresid=ssresid/dfres.
compute varb = msresid*invXtX.
compute
seint=sqrt(varb(nrow(varb),nrow(varb))).
compute k3 = nrow(b).
do if (hc3 = 1).
compute xhc=x.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)= xhc(i3,:)*invXtX*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).
end loop.
compute varb=(invXtX*t(xhc)*xhc*invXtX).
end if.
compute seb = sqrt(diag(varb)).
compute r2 = 1-(ssresid/sstotal).

```

```

do if (mcfoc <> 0 or mcmod <> 0).
compute ytp=y.
compute xtp=x(:,1:(ncol(x)-nx)).
compute btp = inv(t(xtp)*xtp)*t(xtp)*ytp.
compute residt = ytp-(xtp*btp).
compute ssresidt=csum(residt*residt).
compute r2noint = 1-(ssresidt/sstotal).
end if.
compute pr = ncol(x)-1.
compute lmat = ident(nrow(b)).
compute lmat = lmat(:,2:ncol(lmat)).
compute f =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
pr).
compute pf = 1-fcdf(f,pr,dfres).
compute pf = {sqrt(r2),r2,msresid,f,pr,dfres,pf}.
compute xd=abs(xp2).
end if.
do if (itprob = 0).
compute tstat = b&/seb.
do if (ovals <> 2).
compute p = 2*(1-tcdf(abs(tstat), dfres)).
else if (ovals = 2).
compute p = 2*(1-cdfnorm(abs(tstat))).
end if.
compute outp = {b,seb,tstat,p}.
compute nms = {'constant'; nms; 'interact'}.
do if (ovals <> 2).
compute tval = (dfres* (exp((dfres-
(5/6))*((xd/(dfres-(2/3)+(.11/dfres)))*(xd/(dfres-
(2/3)+(.11/dfres)))))-1)).
compute outp={outp,(b-
sqrt(tval)&*seb),(b+sqrt(tval)&*seb)}.
end if.
compute bb = tval.
do if (dumok = 1).
do if (mcfoc > 0).
print dummat/title = 'Coding of categorical X
variable for analysis:' /cnames = xname2/format
= F5.2.

```

```

else if (mcmod > 0).
print dummat/title = 'Coding of categorical M
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
print yname/title =
*****
*****'/rlabels = 'Outcome: '/format
= A8.
do if (ovals = 2).
compute nmsd = {outv, 'Analysis'}.
print rcd/title = 'Coding of binary Y for
analysis: '/cnames = nmsd/format = F10.4.
compute outp = {outp,(outp(:,1)-
sqrt(tval)&*outp(:,2)),(outp(:,1)+sqrt(tval)&*outp
(:,2))}.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
print outp/title = 'Logistic Regression
Model'/rnames = nms/clabels 'Coeff' 'se' 'Z' 'p'
'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
compute varbtmp=varb.
compute btmp=b.

```



```

compute LL2f=LL2.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=x(:,1:(ncol(x)-nx)).
compute bt1 = make(ncol(xlp),1,0).

.

compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(y-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)*&ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.

```

```

compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL2f.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(LLdiff,nx).
compute rcha={LLdiff,nx,pchi}.
end if.
do if (ovals <> 2).
print pf/title = 'Model Summary'/clabels = 'R' 'R-
sq' 'MSE' 'F' 'df1' 'df2' 'p'/format F10.4.
compute lmat=make(nrow(b),1,0).
compute lmat(nrow(lmat),1)=1.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
1).
compute
rcha=((b(nrow(b),1)/seint)*(b(nrow(b),1)/seint))*
(1-r2)/dfres.
compute rcha = {rcha, fcha, 1, dfres,
outp((pr+1),4)}.
print outp/title = 'Model'/rnames = nms/clabels
'coeff' 'se' 't' 'p' 'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
do if (mcfoc > 0 or mcmod > 0).
compute rcha=r2-r2noint.
compute lmat=make((nrow(b)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
nx).
compute pvalr2c=1-fcdf(fcha,nx,dfres).

```

```

compute rcha = {rcha, fcha, nx, dfres,pvalr2c}.
end if.
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute intkey = {'a', 'b', 'c', 'd', 'e'}.
loop i = 1 to nx.
compute intkey={intkey; ddd1(1,i), ' : ', ddd(1,i),
' X ', nm(1,ncol(dd)-(1-mcloc))}.
end loop.
compute intkey=intkey(2:nrow(intkey),:).
print intkey/title='Product terms key:.'/format =
a8.
end if.
do if (ovals <> 2).
print rcha/title = 'R-square increase due to
interaction:.'/clabels 'R2-chng' 'F' 'df1' 'df2'
'p'/format F10.4.
else if (ovals = 2).
print rcha/title = 'Likelihood ratio test for
interaction:.'/clabels 'Chi-sq' 'df' 'p'/format F10.4.
end if.
print/title =
*****
*****'.
compute mdvar = x(:,(ncol(x)-1)).
do if (mcfoc = 0 and mcmmod = 0).
compute g1 = b((ncol(x)-2),1).
compute g3 = b(ncol(x),1).
compute vg1 = varb((ncol(x)-2),(ncol(x)-2)).
compute vg3 = varb(ncol(x),ncol(x)).
compute covg1g3 = varb((ncol(x)-2), ncol(x)).
end if.
do if (mcfoc > 0 or mcmmod > 0).
compute mdvar = x(:,(ncol(x)-(2*nx))).
end if.
compute mdmin = cmin(mdvar).
compute mdmax = cmax(mdvar).
compute fvar = x(:,(ncol(x)-2)).
compute nval = ncol(design(mdvar)).
compute fvmin = cmin(fvar).

```

```

compute fvmax = cmax(fvar).
do if (mmodval = 999).
compute mnmd = csum(mdvar)/n.
compute tmp = make(n,1,mnmd).
compute sdmd = sqrt(csum(((mdvar-
tmp)**2))/(n-1)).
compute probeval = {mnmd-sdmd; mnmd;
mnmd+sdmd}.
do if (probeval(1,1) < mdmin).
compute lowwarn = 1.
compute probeval(1,1)=mdmin.
end if.
do if (probeval(nrow(probeval),1) > mdmax).
compute highwarn = 1.
compute probeval(nrow(probeval),1)=mdmax.
end if.
do if (quantile = 1).
compute tmp = mdvar.
compute tmp(GRADE(mdvar(:,1)),:)= mdvar.
compute mdvar = tmp.
compute
probeval={mdvar(trunc(0.1*n),1);mdvar(trunc(0
.25*n),1);mdvar(trunc(0.5*n),1);mdvar(trunc(0.7
5*n),1);mdvar(trunc(0.9*n),1)}.
end if.
end if.
do if (nval = 2).
compute probeval = make(2,1,0).
compute probeval(1,1)=cmin(mdvar).
loop i = 1 to n.
do if (mdvar(i,1) <> probeval(1,1)).
compute probeval(2,1) = mdvar(i,1).
BREAK.
end if.
end loop.
end if.
do if (mmodval <> 999).
compute probeval = mmodval.
end if.
compute outp = make(nrow(probeval),7,0).

```

```

do if (mcfoc > 0 or mcmmod > 0).
compute focvals=make(1,ncol(x)+1,1).
do if (mcfoc > 0).
print/title = 'Conditional effect of X on Y at
values of the moderator:'.
print/title = ' '/space=0.
compute rnn2=mdtr.
compute matt=make(nx,6,0).
end if.
loop jj=1 to nrow(probeval).
do if (mcfoc > 0).
loop ii=1 to nx.
compute g1=b((ncol(x)-(2*nx)+ii),1).
compute g3=b((ncol(x)-nx+ii),1).
compute vg1=varb((ncol(x)-(2*nx)+ii),(ncol(x)-
(2*nx)+ii)).
compute vg3=varb((ncol(x)-nx+ii),(ncol(x)-
nx+ii)).
compute covg1g3=varb((ncol(x)-
(2*nx)+ii),(ncol(x)-nx+ii)).
compute x2 = probeval(jj,1).
compute w1 = g1+g3*x2.
compute varw1 =
vg1+(2*x2*covg1g3)+((x2*x2)*vg3).
compute sew1 = sqrt(varw1).
compute t1 = w1/sew1.
do if (ovals <> 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-tcdf(abs(t1), dfres)).
compute cnms = {'Coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)= {w1,sew1,t1,p,llci,ulci}.
else if (ovals = 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-cdfnorm(abs(t1))).
compute cnms = {'Coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)= {w1,sew1,t1,p,llci,ulci}.

```

```

end if.
end loop.
compute rnms=t(xname).
compute mdvalpr=probeval(jj,1).
print mdvalpr/title = 'Moderator
value: '/rnames=rnn2/format= F10.4 /space=0.
print matt/title='
'/cnames=cnms/rnames=rnms/format= F10.4
/space=0.
compute xprob=x(:,(ncol(x)-(2*nx)))-mdvalpr.
loop kk = 1 to nx.
compute xprob={xprob,
(xprob(:,1)&*x(:,(ncol(x)-(2*nx)+kk)))}.
end loop.
compute xprob={x(:,1:((ncol(x)-(2*nx))-
1)),xprob}.
do if (ovals <> 2).
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute residc = y-(xprob*bmultc).
compute ssresidc=csum(residc&*residc).
compute r2c = r2-(1-(ssresidc/sstotal)).
compute fcha2=(dfres*r2c)/(nx*(1-r2)).
do if (hc3 = 1).
loop kk = 1 to nx.
compute xprob={xprob, x(:,(ncol(x)-
(2*nx)+kk))}.
end loop.
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute k3 = nrow(bmultc).
compute xhc=xprob.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)=
xhc(i3,1)*inv(t(xprob)*xprob)*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).

```

```

end loop.
compute
varbc=(inv(t(xprob)*xprob)*t(xhc)*xhc*inv(t(xprob)*xprob)).
compute lmat=make((nrow(bmultc)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha2 =
(t(t(lmat)*bmultc)*inv(t(lmat)*varbc*lmat)*((t(lmat)*bmultc)))/nx).
end if.
compute pvalr2cc=1-fcdf(fcha2,nx,dfres).
compute rcha2 = {r2c, fcha2, nx,
dfres,pvalr2cc}.
print rcha2/title = 'Test of equality of conditional
means at this value of the moderator'/clabels
'R2-chng' 'F' 'df1' 'df2' 'p'/format F10.4.
else if (ovals = 2).
compute btmp=b.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=xprob.
compute bt1 = make(ncol(xlp),1,0).

.
compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.

```

```

end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL4.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(lldiff,nx).
compute rcha2 = {LLdiff, nx,pchi}.
print rcha2/title = 'Test of equality of log odds
conditioned on this moderator value'/clabels
'Chi-sq' 'df' 'p'/format F10.4.
end if.
end if.
compute
tttt=make(nrow(dummat),1,probeval(jj,1)).
compute tttt={tttt,dummat(:,2:ncol(dummat))}.
loop kkk=1 to nx.
compute tttt={tttt,tttt(:,1)*tttt(:,1+kkk)}.
end loop.
compute ones=make(nrow(dummat),1,1).
do if (ncol(x) > (2*nx+2)).

```



```

compute
covmnmat=make(nrow(tttt),ncol(covmns),0).
loop kkk=1 to nrow(tttt).
compute covmnmat(kkk,:)=covmns.
end loop.
compute tttt={covmnmat,tttt}.
else.
compute tttt={ones,tttt}.
end if.
compute focvals={focvals;dummat(:,1),tttt}.
do if (mcfoc > 0).
compute yhat={dummat(:,1),(tttt*b)}.
compute cmnms={fciv, 'yhat'}.
do if (ovals <> 2).
print yhat/title = 'Estimated conditional means
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (ovals = 2).
print yhat/title = 'Estimated conditional log odds
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (jj <> nrow(probeval)).
print/title='-----'/space=0.
end if.
end if.
end loop.
compute focvals=focvals(2:nrow(focvals),:).
compute yhat=focvals(:,2:ncol(focvals))*b.
compute
focvals={focvals(:,1),focvals(:,(ncol(focvals)-
(2*nx))),yhat}.
compute cnms={fciv,mdtr,'yhat'}.
do if (mcmod > 0).
compute cnms={mdtr,fciv,'yhat'}.
end if.
end if.
do if (mcmod > 0).
compute outp=make((nx+1),7,0).

```

```

compute bcatm={b((ncol(x)-
(2*nx)),1);b((ncol(x)-nx+1):ncol(x),1)}.
compute outp(:,1)=dummat(:,1).
compute bcatcov=varb((ncol(x)-
nx):ncol(x),(ncol(x)-nx):ncol(x)).
compute bcatcov(1,1)=varb((ncol(x)-
(2*nx)),(ncol(x)-(2*nx))).
compute
bcatcov(2:nrow(bcatcov),1)=varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx)).
compute
bcatcov(1,2:nrow(bcatcov))=t(varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx))).
loop i = 1 to nrow(dummat).
compute
catmval={1,dummat(i,2:ncol(dummat))}.
compute condeff=catmval*bcatm.
compute
condse=sqrt(catmval*bcatcov*t(catmval)).
compute outp(i,2:3)={condeff,condse}.
end loop.
compute outp(:,4)=outp(:,2)&/outp(:,3).
compute outp(:,5) = 2*(1-tcdf(abs(outp(:,4)),
dfres)).
compute outp(:,6) = (outp(:,2)-
sqrt(tval)&*outp(:,3)).
compute outp(:,7) =
(outp(:,2)+sqrt(tval)&*outp(:,3)).
do if (ovals <> 2).
compute cnmms = {xname2(1,1), 'coeff', 'se',
't', 'p', 'LLCI', 'ULCI'}.
end if.
do if (ovals = 2).
compute outp(:,5) = 2*(1-
cdfnorm(abs(outp(:,4)))).
compute cnmms = {xname2(1,1), 'coeff', 'se',
'Z', 'p', 'LLCI', 'ULCI'}.
end if.

```

```

print outp/title = 'Conditional Effect of Focal
Predictor in Groups Defined by the Moderator
Variable:.'/cnames=cnmms/format = F10.4.
end if.
do if (nval > 2 and (mmodval = 999) and
mcmmod = 0).
do if (quantile <> 1).
print/title = 'Moderator values are the sample
mean and plus/minus one SD from mean'.
else.
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
end if.
do if (highwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD above the mean was'.
print/title = 'replaced with the maximum
because one SD above the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
do if (lowwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD below the mean was'.
print/title = 'replaced with the minimum
because one SD below the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
end if.
do if (nval = 2 and (mmodval = 999)).
print/title = 'The moderator variable is
dichotomous'.
end if.
do if (mmodval <> 999 and mcmmod = 0 and
((mmodval < mdmin) or (mmodval > mdmax))).
print/title = 'Warning: Moderator value specified
is outside of the range of the data'.
end if.

```

```

do if (plot = 1).
do if (mcfoc = 0 and mcmmod=0).
compute cnms = {t(nms((ncol(x)-2):(ncol(x)-
1),1)), 'yhat'}.
end if.
do if (ovals = 2).
compute prob =
exp(focvals(:,3))/(1+exp(focvals(:,3))).
compute focvals = {focvals, prob}.
compute cnms = {cnms, 'prob'}.
end if.
print/title =
*****
*****!

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (ovals = 2).
do if (mcfoc=0).
print/title = 'DATA LIST
FREE/Technological_Advan Opra_Pract
logodds prob.' /space=1.
end if.
do if (mcmmod=0).

```

```

print/title = 'DATA LIST FREE/Opra_Pract
Technological_Advan logodds prob.'
/space=1.
end if.
else if (ovals <> 2).
do if (mcfoc=0).
print/title = 'DATA LIST
FREE/Technological_Advan Opr a_Pract
Financial_Performance.' /space=1.
end if.
do if (mcmmod=0).
print/title = 'DATA LIST FREE/Opra_Pract
Technological_Advan Financial_Performance.'
/space=1.
end if.
end if.
print/title = 'BEGIN DATA.' /space=0.
print focvals/title = ' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (ovals <> 2 and mcmmod=0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH Financial_Performance BY
Opra_Pract.' /space=0.
end if.
do if (ovals <> 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
Financial_Performance BY
Technological_Advan.' /space=0.
end if.
do if (ovals = 2 and mcmmod=0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Opr a_Pract.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Opr a_Pract.' /space=0.
end if.
do if (ovals = 2 and mcfoc=0).

```

```

print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
logodds BY Technological_Advan.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Opra_Pract WITH
prob BY Technological_Advan.' /space=0.
end if.
do if (ncovs > 0).
print/title = 'NOTE: For data above, covariates
are set to their sample means.'.
end if.
end if.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
end if.
END IF.
end if.
do if (bad > 0).
compute note(notes,1) = 9.
compute notes = notes + 1.
end if.
print/title = '***** ANALYSIS
NOTES AND WARNINGS
*****'.
loop i = 1 to errs.
do if (runerrs(i,1) = 1).
print/title = 'ERROR: One of your declared
mediators is dichotomous. This procedure
can"t be used.'.
end if.
do if (runerrs(i,1) = 2).

```

```
print/title = 'ERROR: For model 6, this
procedure limits the number of mediators to
four.'.
end if.
do if (runerrs(i,1) = 3).
print/title = 'ERROR: For models 1, 2, and 3,
only a single variable can be listed in the M
list.'.
end if.
do if (runerrs(i,1) = 4).
print/title = 'ERROR: You requested a model
involving W but did not provide a valid W
variable name.'.
end if.
do if (runerrs(i,1) = 5).
print/title = 'ERROR: You requested a model
involving Z but did not provide a valid Z
variable name.'.
end if.
do if (runerrs(i,1) = 6).
print/title = 'ERROR: You requested a model
involving Q but did not provide a valid Q
variable name.'.
end if.
do if (runerrs(i,1) = 7).
print/title = 'ERROR: You requested a model
involving V but did not provide a valid V
variable name.'.
end if.
do if (runerrs(i,1) = 8).
print/title = 'ERROR: You specified a W
variable for a model that does not need it'.
end if.
do if (runerrs(i,1) = 9).
print/title = 'ERROR: You specified a Z variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 10).
print/title = 'ERROR: You specified a Q variable
for a model that does not need it'.
```

```

end if.
do if (runerrs(i,1) = 11).
print/title = 'ERROR: You specified a V variable
for a model that does not need it.'.
end if.
do if (runerrs(i,1) = 12).
print/title = 'ERROR: The variable specified for
W has already been assigned.'.
end if.
do if (runerrs(i,1) = 13).
print/title = 'ERROR: The variable specified for
Z has already been assigned.'.
end if.
do if (runerrs(i,1) = 14).
print/title = 'ERROR: The variable specified for
Q has already been assigned.'.
end if.
do if (runerrs(i,1) = 15).
print/title = 'ERROR: The variable specified for
V has already been assigned.'.
end if.
do if (runerrs(i,1) = 16).
print/title = 'ERROR: You did not provide a
valid Y variable name.'.
end if.
do if (runerrs(i,1) = 17).
print/title = 'ERROR: The variable specified for
Y has already been assigned.'.
end if.
do if (runerrs(i,1) = 18).
print/title = 'ERROR: Model 6 requires more
than one mediator.'.
end if.
do if (runerrs(i,1) = 19).
print/title = 'ERROR: You have not specified a
valid model number.'.
end if.
do if (runerrs(i,1) = 20).
print/title = 'ERROR: At least one and only one
variable must be listed for X.'.

```



```
end if.  
do if (runerrs(i,1) = 21).  
print/title = 'ERROR: At least one and only one  
variable must be listed for Y.'.  
end if.  
do if (runerrs(i,1) = 22).  
print/title = 'ERROR: Iteration didn"t converge  
to a solution. Interpret results with caution.'.  
end if.  
do if (runerrs(i,1) = 23).  
print/title = 'ERROR: You specified a clustering  
variable that does not exist in your variable  
list.'.  
end if.  
do if (runerrs(i,1) = 24).  
print/title = 'ERROR: You specified a clustering  
variable that has already been assigned.'.  
end if.  
do if (runerrs(i,1) = 25).  
print/title = 'ERROR: One or more of your M  
variables is not listed in the variables list.'.  
end if.  
do if (runerrs(i,1) = 26).  
print/title = 'ERROR: A maximum of 20 cluster  
units is allowed. Use multilevel modeling  
instead.'.  
end if.  
do if (runerrs(i,1) = 27).  
print/title = 'ERROR: One of the variables in  
your model is a constant.'.  
end if.  
do if (runerrs(i,1) = 28).  
print/title = 'ERROR: Dichotomous Y is not  
permitted with WS option.'.  
end if.  
do if (runerrs(i,1) = 29).  
print/title = 'ERROR: Insufficient number of  
variables in vars= list when using WS option.'.  
end if.  
do if (runerrs(i,1) = 30).
```

```

print/title = 'ERROR: Too many variables in
vars= list when using WS option. Covariates
not allowed.'.
end if.
do if (runerrs(i,1) = 31).
print/title = 'ERROR: mmodval and wmodval
can"t both be set to zero with contrast option.'.
end if.
do if (runerrs(i,1) = 32).
print/title = 'ERROR: You did not provide a
valid X variable name.'.
end if.
do if (runerrs(i,1) = 33).
print/title = 'ERROR: PROCESS requires all
variable names to be eight characters or
fewer.'.
print/title = '      Please shorten variable names
and reexecute.'/space=0.
end if.
do if (runerrs(i,1)=34).
print/title = 'ERROR: X and M cannot both be
specified as multicategorical.'.
end if.
do if (runerrs(i,1)=35).
print/title = 'ERROR: Multicategorical variable
specification available only for models 1 and
4.'.
end if.
do if (runerrs(i,1)=36).
print/title = 'ERROR: Categorical variables
cannot have more than 10 categories.'.
end if.
do if (runerrs(i,1)=37).
print/title = 'ERROR: Each group must have at
least two cases.'.
end if.
do if (runerrs(i,1)=38).
print/title = 'ERROR: Cluster option not
available with multicategorical variables.'.
end if.

```

```

do if (runerrs(i,1)=39).
print/title = 'ERROR: A maximum of 10
variables is allowed in the M= list.'.
end if.
do if (runerrs(i,1) = 41).
print/title = 'ERROR: MCM option not permitted
in model 4.'.
end if.
end loop.
do if (errs = 0).
do if (boot > 1 or mc > 0).
do if (bconoff = 1 and boot > 1).
print boot/title = 'Number of bootstrap samples
for bias corrected bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (bconoff = 0 and boot > 1).
print boot/title = 'Number of bootstrap samples
for percentile bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (mc > 1).
print mc/title = 'Number of samples for Monte
Carlo confidence intervals:'/format = F8.0.
end if.
do if (booterr = 1).
compute badend = badend(1,2:ncol(badend)).
print badend/title = 'WARNING: Bootstrap CI
endpoints below not trustworthy. Decrease
confidence or increase bootstraps'/format =
F10.4.
end if.
end if.
print conf/title = 'Level of confidence for all
confidence intervals in output:'/format = F8.2.
do if ((center = 1 or ws=1) and (ncol(centvar) >
1)).
compute centvar = centvar(1,2:ncol(centvar)).

```

```

print centvar/title = 'NOTE: The following
variables were mean centered prior to
analysis:.'/format = a8.
end if.
loop i = 1 to notes.
do if (note(i,1) = 1).
print/title = 'NOTE: Confidence level restricted
to between 50 and 99.9999%. 95% confidence
is provided in output'.
end if.
do if (note(i,1) = 2).
print/title = 'NOTE: Effect size measures for
indirect effects are not available for models
with dichotomous outcomes'.
end if.
do if (note(i,1) = 3).
print/title = 'NOTE: All standard errors for
continuous outcome models are based on the
HC3 estimator'.
end if.
do if (note(i,1) = 6).
print/title = 'NOTE: The number of bootstrap
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 7).
print/title = 'NOTE: The Johnson-Neyman
method is available only for Models 1 and 3'.
end if.
do if (note(i,1) = 8).
print/title = 'NOTE: The Johnson-Neyman
method cannot be used with a dichotomous
moderator'.
end if.
do if (note(i,1)=9).
print bad/title = 'NOTE: Some bootstrap
samples had to be replaced. The number of
such replacements was:'.
end if.
do if (note(i,1) = 11).

```

```

print nmiss/title = 'NOTE: Some cases were
deleted due to missing data. The number of
such cases was:'.
end if.
do if (note(i,1) = 12).
print /title = 'NOTE: Monte Carlo method
available only for models 4 and 5.
Bootstrapping was used instead.'.
end if.
do if (note(i,1) = 13).
print/title = 'NOTE: The number of Monte Carlo
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 19).
print/title = 'NOTE: Effect sizes not available for
within-subject analyses.'.
end if.
do if (note(i,1) = 16).
print/title = 'NOTE: Normal theory tests not
available for within-subject analyses.'.
end if.
do if (note(i,1) = 17).
print/title = 'NOTE: Monte Carlo confidence
intervals not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 18 and warnrep=0).
print/title = 'WARNING: You have requested
OLS estimation with a dichotomous criterion.'.
print/title = 'Interpret model coefficients and
inferential statistics with caution.'/space=0.
compute warnrep=1.
end if.
do if (note(i,1) = 20).
print/title = 'NOTE: Saving of bootstrap
estimates not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 22).

```

```
print/title = 'NOTE: Effect size option with
covariates requires covariates in models of M
and Y.'.
end if.
do if (note(i,1) = 23).
print/title = 'NOTE: Johnson-Neyman method
not available with multicategorical X'.
end if.
do if (note(i,1) = 24).
print/title = 'NOTE: MMODVAL option not
available with a multicategorical moderator'.
end if.
do if (note(i,1) = 25).
print/title = 'NOTE: Omnibus test for relative
indirect effects is not available with the MC
option.'.
end if.
do if (note(i,1) = 26).
print/title = 'NOTE: CONTRAST option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 27).
print/title = 'NOTE: EFFSIZE option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 28).
print/title = 'NOTE: NORMAL option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 31).
print/title = 'NOTE: The TOTAL option is not
available when Y is dichotomous.'.
end if.
do if (note(i,1) = 32).
print/title = 'NOTE: Kappa-squared is disabled
from output as of version 2.16.'.
end if.
do if (note(i,1) = 33).
print/title = 'NOTE: COVMY option ignored in
models 1, 2, and 3.'.
```

| | | | |
|-----------|----------------|-------------|-------------|
| | | end if. | |
| | | end loop. | |
| | | end if. | |
| | | end matrix. | |
| Resources | Processor Time | | 00:00:01.13 |
| | Elapsed Time | | 00:00:01.04 |

Run MATRIX procedure:

***** PROCESS Procedure for SPSS Release 2.16.3 *****

Written by Andrew F. Hayes, Ph.D. www.afhayes.com
Documentation available in Hayes (2013). www.guilford.com/p/hayes3

Model = 1
Y = Financia
X = Opra_Pra
M = Technolo

Sample size
856

Outcome: Financia

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|-----------|--------|----------|-------|
| .9148 | .8369 | .1980 | 1457.6198 | 3.0000 | 852.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|--------|-------|---------|-------|--------|--------|
| constant | -.3493 | .1206 | -2.8973 | .0039 | -.5860 | -.1127 |
| Technolo | .8252 | .0482 | 17.1039 | .0000 | .7305 | .9199 |
| Opra_Pra | .4626 | .0468 | 9.8899 | .0000 | .3708 | .5544 |
| int_1 | -.0525 | .0116 | -4.5147 | .0000 | -.0753 | -.0297 |

Product terms key:

int_1 Opra_Pra X Technolo

R-square increase due to interaction(s):

| | R2-chng | F | df1 | df2 | p |
|-------|---------|---------|--------|----------|-------|
| int_1 | .0039 | 20.3823 | 1.0000 | 852.0000 | .0000 |

Conditional effect of X on Y at values of the moderator(s):

| Technolo | Effect | se | t | p | LLCI | ULCI |
|----------|--------|-------|---------|-------|-------|-------|
| 2.7372 | .3189 | .0286 | 11.1315 | .0000 | .2626 | .3751 |
| 3.8107 | .2625 | .0289 | 9.0702 | .0000 | .2057 | .3193 |
| 4.8843 | .2062 | .0342 | 6.0361 | .0000 | .1391 | .2732 |

Values for quantitative moderators are the mean and plus/minus one SD from mean.

Values for dichotomous moderators are the two values of the moderator.

***** JOHNSON-NEYMAN TECHNIQUE *****

There are no statistical significance transition points within the observed range of the moderator.

Data for visualizing conditional effect of X on Y

Paste text below into a SPSS syntax window and execute to produce plot.

DATA LIST FREE/Opra_Pract Technological_Advan Financial_Performance.
BEGIN DATA.

| | | |
|--------|--------|--------|
| 2.7750 | 2.7372 | 2.7943 |
| 3.9019 | 2.7372 | 3.1537 |
| 5.0288 | 2.7372 | 3.5130 |
| 2.7750 | 3.8107 | 3.5238 |
| 3.9019 | 3.8107 | 3.8197 |
| 5.0288 | 3.8107 | 4.1155 |
| 2.7750 | 4.8843 | 4.2533 |
| 3.9019 | 4.8843 | 4.4856 |
| 5.0288 | 4.8843 | 4.7179 |

END DATA.

GRAPH/SCATTERPLOT=Opra_Pract WITH Financial_Performance BY
Technological_Advan.

***** ANALYSIS NOTES AND WARNINGS *****

Level of confidence for all confidence intervals in output:
95.00

----- END MATRIX -----

restore.

Matrix

Notes

| | | | |
|----------------|--------------------------------|----------------------|--|
| Output Created | | 20-OCT-2020 16:07:32 | |
| Comments | | | |
| Input | Active Dataset | DataSet1 | |
| | Filter | <none> | |
| | Weight | <none> | |
| | Split File | <none> | |
| | N of Rows in Working Data File | 856 | |

Syntax

```
matrix.  
get dat/file = */variables =  
Financial_Performance Comm_Pract  
Technological_Advan /names =  
vnames/missing = 9999.  
compute ninit = nrow(dat).  
get dat/file = */variables =  
Financial_Performance Comm_Pract  
Technological_Advan /names =  
vnames/missing = omit.  
get tmp/file = */variables =  
Financial_Performance /names =  
ynames/missing = omit.  
get tmp2/file = */variables = Comm_Pract  
/names = xnames/missing = omit.  
get tmp3/file = */variables =  
Technological_Advan /names =  
mnames/missing = omit.  
get tmp4/file = */variables = w999999t z999999t  
v999999t q999999t.  
compute wname=tmp(1,1).  
do if (wname = ' ').  
compute wname = 'xxx'.  
end if.  
compute zname=tmp(1,2).  
do if (zname = ' ').  
compute zname = 'xxx'.  
end if.  
compute vname=tmp(1,3).  
do if (vname = ' ').  
compute vname = 'xxx'.  
end if.  
compute qname=tmp(1,4).  
do if (qname = ' ').  
compute qname = 'xxx'.  
end if.  
compute n = nrow(dat).  
compute p0=-.322232431088.  
compute p1 = -1.  
compute p2 = -.342242088547.
```

```

compute p3 = -.0204231210245.
compute p4 = -.0000453642210148.
compute q0 = .0993484626060.
compute q1 = .588581570495.
compute q2 = .531103462366.
compute q3 = .103537752850.
compute q4 = .0038560700634.
compute badend = 0.
compute priorlo = -9999999.
compute priorhi = 9999999.
compute criterr = 0.
compute cluster = 0.
compute clsdmy = 0.
compute jndich = 0.
compute wvdich=0.
compute zqdich=0.
compute mod74dic=0.
compute warnrep=0.
compute contrast = trunc( 0 ).
do if (contrast > 2 or contrast < 0).
compute contrast = 0.
end if.
compute booterr = 0.
compute effsize = ( 0 = 1).
compute note = make(10,1,0).
compute notes = 1.
compute iterr = 0.
compute clsmth = 0.
compute quantile = ( 0 = 1).
compute mmodval= 999.
compute jn = ( 1 = 1).
compute covcoeff=( 0 =1).
compute center = ( 0 = 1).
compute detail = ( 1 = 1).
compute coeffci = ( 1 = 1).
compute conf = 95.
compute longname = ( 1 =1).
compute bconoff=( 0 <> 1).
compute covmy = trunc( 0 ).
do if (covmy < 0 or covmy > 2).

```

```

compute covmy = 0.
end if.
do if (trunc(conf) >= 100 or (trunc(conf) <=
50)).
compute conf = 95.
compute note(notes,1) = 1.
compute notes = notes + 1.
end if.
do if (n < ninit).
compute nmiss = ninit-n.
compute note(notes,1) = 11.
compute notes = notes + 1.
end if.
compute errs = 0.
compute quantd = {0,0,0,0,0,0}.
compute quantc = {0,0,0,0,0,0}.
compute mcheck = 0.
compute ttt = 0.
compute plot = trunc( 1 ).
compute runerrs = make(50,1,0).
compute model = trunc( 1 ).
compute normal = 0.
compute ws= ( 0 =1).
compute olsdichm= ( 0 =1).
compute olsdichy= ( 0 =1).
compute mcx=trunc( 0 ).
compute mcm=trunc( 0 ).
do if (ws = 1).
do if (effsize = 1).
compute note(notes,1) = 19.
compute notes = notes + 1.
compute effsize=0.
end if.
do if (normal=1).
compute note(notes,1) = 16.
compute notes = notes + 1.
compute normal=0.
end if.
end if.
do if (model <> 4).

```

```

compute ws = 0.
end if.
do if (model > 0 and model < 4 and covmy <>
0).
compute covmy=0.
compute note(notes,1) = 33.
compute notes = notes + 1.
end if.
print/space=0.
do if (jn = 1 and model <> 1 and model <> 3).
compute note(notes,1) = 7.
compute notes = notes + 1.
end if.
do if ((model > 76) or (model < 1)).
compute model = 77.
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 19.
end if.
compute itprob=0.
do if (mcx < 0).
compute mcx=0.
end if.
do if (mcx > 4).
compute mcx=0.
end if.
do if (mcm < 0).
compute mcm=0.
end if.
do if (mcm > 4).
compute mcm=0.
end if.
do if (mcx > 0 and mcm > 0).
compute errs=errs+1.
compute runerrs(errs,1)=34.
compute criterr=1.
end if.
do if ((model <> 1 and model <> 4) and (mcx >
0 or mcm > 0)).
compute errs=errs+1.

```

```

compute runerrs(errs,1)=35.
compute criterr=1.
end if.
do if ((model = 4) and (mcm > 0)).
compute errs=errs+1.
compute runerrs(errs,1)=41.
compute criterr=1.
end if.
do if ((mcx > 0) and jn <> 0).
compute jn=0.
compute note(notes,1) = 23.
compute notes = notes + 1.
end if.
do if (mmodval <> 999 and mcm > 0).
compute note(notes,1) = 24.
compute notes = notes + 1.
end if.
compute toteff = 0.
compute toteff = (( 0 = 1)*( 1 = 4 or 1 = 6)).
compute varorder = 2.
compute hc3 = ( 0 <> 0).
compute cname = 'xxx'.
compute centvar = {'xxx'}.
compute nmeds = ncol(mnames).
do if (longname=0).

```

```
compute criterr = 1.  
compute errs = errs+1.  
compute runerrs(errs,1) = 33.
```

1

2

3

end if.

compute modelm =

```
{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,2;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,3;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,4;  
0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,5;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,6;  
1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,7;  
1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,8;  
1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,9;  
1,1,0,0,0,0,1,1,0,0,0,0,0,0,0,0,3,10;  
1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,5,11;  
1,1,1,0,0,0,1,1,1,0,0,0,0,0,0,0,5,12;  
1,1,1,0,0,0,1,0,0,0,0,0,0,0,0,0,5,13;  
0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,2,14;  
0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,2,15;  
0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,4,16;  
0,0,0,1,1,0,0,0,0,1,1,0,0,0,0,0,4,17;  
0,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,6,18;  
0,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,6,19;  
0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,6,20;
```

1,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,7,21;
1,0,0,1,0,0,1,0,0,0,0,0,0,0,0,0,7,22;
1,1,0,1,0,0,0,0,0,0,0,0,0,0,0,0,23;
1,1,0,1,0,0,1,1,0,0,0,0,0,0,0,0,24; 1,1,1,1

,0,0,0,0,0,0,0,0,0,0,0,0,0,0,25;
1,1,1,1,0,0,1,1,1,0,0,0,0,0,0,0,26;
1,1,1,1,0,0,1,0,0,0,0,0,0,0,0,0,27;
1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,7,28;
1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,7,29;
1,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,30;
1,1,0,1,0,0,1,1,0,1,0,0,0,0,0,0,31;
1,1,1,1,0,0,0,0,0,1,0,0,0,0,0,0,32;
1,1,1,1,0,0,1,1,1,1,0,0,0,0,0,0,33;
1,1,1,1,0,0,1,0,0,1,0,0,0,0,0,0,34;
1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,35;
1,0,0,1,1,0,0,0,0,0,1,1,0,0,0,0,0,36;
1,0,0,1,1,1,0,0,0,0,0,0,0,0,0,0,37;
1,0,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,38;
1,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,39;
1,0,0,1,1,0,1,0,0,0,0,0,0,0,0,0,40;
1,0,0,1,1,0,1,0,0,1,1,0,0,0,0,0,0,41;
1,0,0,1,1,1,1,0,0,0,0,0,0,0,0,0,42;
1,0,0,1,1,1,1,0,0,1,1,1,0,0,0,0,0,43;
1,0,0,1,1,1,1,0,0,1,0,0,0,0,0,0,0,44;
1,1,0,1,1,0,0,0,0,0,0,0,0,0,0,0,45;
1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,46;
1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,47;
1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,48;
1,1,0,1,1,0,1,1,0,0,0,

0,0,0,0,0,0,0,0,49;
1,1,0,1,1,0,0,0,0,1,1,0,0,0,0,0,0,50;
1,1,1,1,1,0,1,1,1,0,0,0,0,0,0,0,0,51;
1,1,1,1,1,0,0,0,0,1,1,0,0,0,0,0,0,52;
1,1,0,1,1,1,1,1,0,0,0,0,0,0,0,0,53;
1,1,0,1,1,1,0,0,0,1,1,1,0,0,0,0,0,54;
1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,55;
1,1,1,1,1,1,0,0,0,1,1,1,0,0,0,0,0,56;
1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,57;

```
1,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,58;  
1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,59;  
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,60;  
1,1,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,61;  
1,1,0,0,0,0,1,0,0,0,0,1,0,0,0,0,0,0,0,62;  
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,0,0,0,0,63;  
1,0,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,64;  
1,0,0,1,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,65;  
1,0,0,1,0,0,0,0,0,1,0,0,1,0,0,0,0,0,0,66;  
1,0,0,1,0,0,1,0,0,1,0,0,1,0,0,0,0,0,0,67;  
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,68;  
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,0,0,0,0,69;  
1,0,0,1,0,0,0,0,0,0,0,0,1,1,0,0,0,0,0,70;  
1,0,0,1,0,0,1,0,0,1,0,0,1,1,1,0,0,0,0,71;  
1,1,1,0,0,0,0,0,0,0,0,0,1,0,0,1,1,0,0,72;  
1,1,1,0,0,0,1,1,1,0,0,0,1,0,0,1,1,0,0
```

```
,73; 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,74;  
1,1,0,0,0,0,0,0,0,0,0,0,1,0,0,1,0,0,0,75;  
1,1,0,0,0,0,1,1,0,0,0,0,1,0,0,1,0,0,0,76;  
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,77}.
```

```
compute wm = modelm(model, 1).
```

```
compute zm = modelm(model, 2).
```

```
compute wzm = modelm(model, 3).
```

```
compute vy = modelm(model, 4).
```

```
compute qy = modelm(model, 5).
```

```
compute vqy = modelm(model, 6).
```

```
compute wy = modelm(model, 7).
```

```
compute zy = modelm(model, 8).
```

```
compute wzy = modelm(model, 9).
```

```
compute vxy = modelm(model, 10).
```

```
compute qxy = modelm(model, 11).
```

```
compute vqxy = modelm(model, 12).
```

```
compute wmy = modelm(model, 13).
```

```
compute wvmy = modelm(model, 14).
```

```
compute wvxy = modelm(model, 15).
```

```
compute zmy = modelm(model,16).
```

```
compute wzmy = modelm(model,17).
```

```
compute xmy = modelm(model,18).
```

```
compute imm=modelm(model,19).
```

```

do if (model = 74).
compute vdich=0.
end if.
do if (ncol(xname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 20.
compute criterr = 1.
end if.
do if (ncol(mnames) > 10).
compute errs = errs+1.
compute runerrs(errs,1) = 39.
compute criterr = 1.
end if.
do if (ncol(yname) <> 1).
compute errs = errs+1.
compute runerrs(errs,1) = 21.
compute criterr = 1.
end if.
do if (( 999 =0) and ( 999 =0) and (model=3 or
model=2) and (contrast=1)).
compute errs = errs+1.
compute runerrs(errs,1) = 31.
compute criterr = 1.
end if.
compute xlist = (wm or zm or wzm or wy or zy
or wzy or vxy or qxy or vqxy or wvxy or xmy).
compute mlist = (vy or qy or vqy or zmy or wmy
or wzmy or xmy or (model < 4)).
compute bad=0.
do if (criterr = 0).
compute werr = 0.
compute verr = 0.
compute qerr = 0.
compute zerr = 0.
compute yerr = 1.
compute xerr = 1.
compute wlist = (wm or wzm or wy or wzy or
wm or wvmy or wvmy or wvxy or wzmy).
do if (wlist = 1 and wname = 'xxx').
compute werr = 1.

```

```

compute wlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (wlist = 1 and ((wname = qname) or
(wname = vname) or (wname = zname) or
(wname = xname) or (wname = yname))).
compute werr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 12).
end if.
compute zlist = (zm or wzm or zy or wzy or
zmy or wzmy).
do if (zlist = 1 and zname = 'xxx').
compute zerr = 1.
compute zlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (zlist = 1 and ((zname = qname) or
(zname = vname) or (zname = wname) or
(zname = xname) or (zname = yname))).
compute zerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 13.
end if.
compute qlist = (qy or vqy or qxy or vqxy).
do if (qlist = 1 and qname = 'xxx').
compute qerr = 1.
compute qlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (qlist = 1 and ((qname = zname) or
(qname = vname) or (qname = wname) or
(qname = xname) or (qname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 14.
end if.

```

```

compute vlist = (vy or vqy or vxy or vqxy or
wvmy or wvxy).
do if (vlist = 1 and vname = 'xxx').
compute verr = 1.
compute vlist = 0.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (vlist = 1 and ((vname = zname) or
(vname = qname) or (vname = wname) or
(vname = xname) or (vname = yname))).
compute qerr = 4.
compute errs = errs+1.
compute runerrs(errs,1) = 15.
end if.
do if (wlist = 0 and wname <> 'xxx').
compute werr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 8.
end if.
do if (zlist = 0 and zname <> 'xxx').
compute zerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 9.
end if.
do if (qlist = 0 and qname <> 'xxx').
compute qerr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 10.
end if.
do if (vlist = 0 and vname <> 'xxx').
compute verr = 2.
compute errs = errs+1.
compute runerrs(errs,1) = 11.
end if.
do if (hc3 = 1).
compute note(notes,1) = 3.
compute notes = notes+1.
end if.
compute alpha2 = (1-(conf/100))/2.

```

```

compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute cons = make(n,1,1).
compute temp = (n*sscp(dat))-
(t(csum(dat))*(csum(dat))).
compute temp = temp/(n*(n-1)).
compute temp = csum((((diag(temp))=0)).
compute temp2=1.
do if (ws = 1).
compute temp2 = (n*sscp(tmp2))-
(t(csum(tmp2))*(csum(tmp2))).
compute temp2 = temp2/(n*(n-1)).
end if.
do if ((temp > 0 and ws = 0) or (temp > 0 and
ws = 1 and temp2 > 0)).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 27.
end if.
compute nmeds = ncol(mnames).
compute mcmats=ident(nmeds*2).
compute mccoeff=make(nmeds*2,1,0).
compute sobel = make(nmeds,4,-999).
do if (model = 6 and nmeds > 4).
compute errs = errs+1.
compute runerrs(errs,1)=2.
end if.
do if (model < 4 and nmeds > 1).
compute errs = errs+1.
compute runerrs(errs,1) = 3.
end if.
compute nmods = (model = 74).
compute bad = 0.
compute intcnt = 1.
compute modvals = 0.
compute modvalsd = 0.
compute yintemp = {'int_1', 'int_2', 'int_3',
'int_4', 'int_5', 'int_6', 'int_7', 'int_8', 'int_9',

```

```

'int_10', 'int_11', 'int_12', 'int_13', 'int_14',
'int_15'}.
compute yintemp = {yintemp, 'int_16', 'int_17',
'int_18', 'int_19', 'int_20', 'int_21', 'int_22',
'int_23', 'int_24', 'int_25', 'int_26', 'int_27',
'int_28'}.
compute yintemp = {yintemp, 'int_29', 'int_30',
'int_31', 'int_32', 'int_33', 'int_34', 'int_35',
'int_36', 'int_37', 'int_38', 'int_39', 'int_40'}.
compute yintemp = {yintemp, 'int_41', 'int_42',
'int_43', 'int_44', 'int_45', 'int_46', 'int_47',
'int_48', 'int_49', 'int_50', 'int_51', 'int_52'}.
compute yintemp = {yintemp, 'int_53', 'int_54',
'int_55', 'int_56', 'int_57', 'int_58', 'int_59',
'int_60', 'int_61', 'int_62', 'int_63', 'int_64'}.
compute
cntname={'(C1)';'(C2)';'(C3)';'(C4)';'(C5)';'(C6)';'(
C7)';'(C8)';'(C9)';'(C10)';'(C11)';'(C12)';'(C13)';(
C14)';'(C15)';'(C16)';'(C17)'}.
compute
cntname={cntname;'(C18)';'(C19)';'(C20)';'(C21
)';'(C22)';'(C23)';'(C24)';'(C25)';'(C26)';'(C27)';(
C28)';'(C29)';'(C30)';'(C31)'}.
compute
cntname={cntname;'(C32)';'(C33)';'(C34)';'(C35
)';'(C36)';'(C37)';'(C38)';'(C39)';'(C40)';'(C41)';(
C42)';'(C43)';'(C44)';'(C45)'}.
compute
cntname={cntname;'(C46)';'(C47)';'(C48)';'(C49
)';'(C50)';'(C51)';'(C52)';'(C53)';'(C54)';'(C55)';(
C56)';'(C57)';'(C58)';'(C59)'}.
compute
cntname={cntname;'(C60)';'(C61)';'(C62)';'(C63
)';'(C64)';'(C65)';'(C66)';'(C67)';'(C68)';'(C69)';(
C70)';'(C71)';'(C72)';'(C73)'}.
compute
cntname={cntname;'(C74)';'(C75)';'(C76)';'(C77
)';'(C78)';'(C79)';'(C80)';'(C81)';'(C82)';'(C83)';(
C84)';'(C85)';'(C86)';'(C87)'}.

```



```

compute
cntname={cntname;'(C88)';'(C89)';'(C90)';'(C91
)';'(C92)';'(C93)';'(C94)';'(C95)';'(C96)';'(C97)';'(
C98)';'(C99)';'(C100)';'(C101)'}.
compute
cntname={cntname;'(C102)';'(C103)';'(C104)';'(
C105)'}.
compute apathnam={'a path'; 'a1 path'; 'a2
path'; 'a3 path'; 'a4 path'; 'a5 path'; 'a6 path';
'a7 path'; 'a8 path'; 'a9 path'; 'a10 path'}.
compute bpathnam={'b path'; 'b1 path'; 'b2
path'; 'b3 path'; 'b4path'; 'b5path'; 'b6 path'; 'b7
path'; 'b8 path'; 'b9 path'; 'b10 path'}.
compute modvnm = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute modvnm2 = {'xxx', 'xxx', 'xxx', 'xxx',
'xxx'}.
compute mlab = { ' M1 ='; ' M2 ='; ' M3 ='; '
M4 ='; ' M5 ='; ' M6 ='; ' M7 ='; ' M8 ='; '
M9 ='; ' M10 ='}.
compute m = make(n,nmeds,1).
compute mmat = make(16,nmeds,0).
compute ymat = make(8,nmeds,0).
compute deco = make(10,1,0).
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
compute iterate = abs(trunc( 10000 )).
compute converge = abs( 0.00000001 ).
compute boot = abs(trunc( 5000 )).
do if (ws=1 and 0 > 0).
compute note(notes,1)=17.
compute notes=notes+1.
end if.
compute mc=(abs(trunc( 0 ))*(1-ws)).
do if (mc > 0 and model > 5).
do if (boot = 0).
compute boot=mc.
end if.

```

```

compute mc=0.
compute note(notes,1)=12.
compute notes = notes+1.
end if.
compute adjust = 0.
do if (mc > 0 and (model > 3 and model < 6)).
compute boot=0.
compute bconoff=0.
end if.
do if (boot > 0 and mc > 0).
compute mc = 0.
end if.
compute savboot = 0.
compute saveboot = ( 0 = 1).
do if (ws=1 and saveboot=1).
compute saveboot=0.
compute note(notes,1) = 20.
compute notes = notes + 1.
compute normal=0.
end if.
do if (saveboot =1 and boot > 0 and model >
3).
compute savboot = 1.
end if.
do if (boot <> 0 or mc <> 0).
compute bootsz=boot.
do if (mc > 0).
compute bootsz=mc.
end if.
loop.
compute cilow = rnd(bootsz*(1-(conf/100))/2).
compute cihigh =
trunc((bootsz*(conf/100)+(bootsz*(1-
(conf/100))/2)))+1.
do if (cilow < 1 or cihigh > bootsz).
compute
bootsz=trunc((bootsz+1000)/1000)*1000.
compute adjust = 1.
end if.
end loop if (cilow gt 0 and cihigh le bootsz).

```

```

compute boot=bootsz.
do if (mc > 0).
compute mc = bootsz.
end if.
do if (boot > 0 and mc > 0 and (model > 3 and
model < 6)).
compute boot=0.
end if.
do if (adjust = 1 and boot > 0).
compute note(notes,1)=6.
compute notes = notes+1.
end if.
do if (adjust = 1 and mc > 0).
compute note(notes,1)=13.
compute notes = notes+1.
end if.
end if.
do if (model = 6 and nmeds > 1).
compute mmpaths =
make((nmeds+2),(nmeds+2),0).
do if (nmeds = 2).
compute indboot = make(boot+1, 3, 999).
else if (nmeds = 3).
compute indboot = make(boot+1, 7, 999).
else if (nmeds = 4).
compute indboot = make(boot+1, 15, 999).
end if.
compute indlbl = {'Total:','Ind1 :','Ind2 :','Ind3
:','Ind4 :','Ind5 :','Ind6 :','Ind7 :','Ind8 :','Ind9
:','Ind10:','Ind11:','Ind12:','Ind13:','Ind14:','
Ind15:'}.
compute indlbl2 = {'Ind1','Ind2','Ind3','Ind4';
'Ind5','Ind6','Ind7','Ind8','Ind9','Ind10';
'Ind11','Ind12','Ind13','Ind14','Ind15'}.
compute indces = make(boot+1, 4, 999).
end if.
do if (model < 4).
compute boot = 0.
compute cmat = make(10,1,0).
compute zmat = make(10,1,0).

```

```

end if.
compute nvarch = make(1,ncol(dat),0).
compute wmatch = 0.
compute zmatch = 0.
compute vmatch = 0.
compute qmatch = 0.
compute mmatch=0.
compute minprobe=0.
compute maxprobe=0.
loop i = 1 to ncol(vnames).
do if (vnames(:,i)=yname).
compute y = dat(:,i).
compute nvarch(1,i)=1.
compute yerr = 0.
do if ((yname = xname) or (yname = wname) or
(yname = zname) or (yname = vname) or
(yname = qname)).
compute errs = errs+1.
compute runerrs(errs,1)=17.
end if.
end if.
do if (vnames(:,i)=xname).
compute x = dat(:,i).
compute nvarch(1,i)=1.
compute xdich = 1.
compute xerr=0.
loop jj = 1 to n.
do if ((x(jj,1) <> cmin(x)) and (x(jj,1) <>
cmax(x))).
compute xdich = 0.
break.
end if.
end loop.
compute xmean = csum(x)/n.
do if (center = 1 and (model < 4 or xlist > 0)
and mcs = 0).
compute meanvec = make(n,1, xmean).
compute x = x-meanvec.
compute centvar = {centvar, xname}.
end if.

```

```

compute xmean = csum(x)/n.
compute tmp = x-(cons*xmean).
compute xsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
do if (xdich = 0).
compute quantc(1,6) = 1.
compute matx = {xmean-xsd; xmean;
xmean+xsd}.
do if ( 999 = 999 and quantile = 0).
do if ((matx(1,1) < cmin(x)) and model = 74).
compute matx(1,1)=cmin(x).
compute minprobe=1.
end if.
do if ((matx(3,1) > cmax(x)) and model = 74).
compute matx(3,1)=cmax(x).
compute maxprobe=1.
end if.
end if.
do if (quantile = 1).
compute quantd(1,6) = 1.
compute quantc(1,6) = 0.
compute tmp = x.
compute tmp(GRADE(x)) = x.
compute matx =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (xdich = 1).
compute matx = {cmin(x); cmax(x)}.
do if (model=74).
compute matx=cmin(x).
compute mod74dic=1.
end if.
end if.
do if ( 999 <> 999).
compute matx = 999.
compute quantd(1,6) = 0.
compute quantc(1,6) = 0.
end if.

```

```

end if.
do if (werr = 0 and wlist = 1).
do if (vnames(:,i)=wname).
compute werr = 0.
compute wmatch = 1.
compute w = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(w)/n).
compute w = w-meanvec.
compute centvar = {centvar, wname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute wmean = csum(w)/n.
compute tmp = w-(cons*wmean).
compute wsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute wdich = 1.
loop jj = 1 to n.
do if ((w(jj,1) <> cmin(w)) and (w(jj,1) <>
cmax(w))).
compute wdich = 0.
break.
end if.
end loop.
do if (model = 3).
compute jndich=wdich.
compute jnmin=cmin(w).
compute jnmax=cmax(w).
end if.
do if (wdich = 0).
compute matw = {wmean-wsd; wmean;
wmean+wsd}.
do if ( 999 = 999 and quantile = 0).
do if (matw(1,1) < cmin(w)).
compute matw(1,1)=cmin(w).
compute minprobe=1.
end if.
do if (matw(3,1) > cmax(w)).
compute matw(3,1)=cmax(w).
compute maxprobe=1.

```

```

end if.
end if.
compute quantc(1,1) = 1.
do if (quantile = 1).
compute quantd(1,1) = 1.
compute quantc(1,1) = 0.
compute tmp = w.
compute tmp(GRADE(w)) = w.
compute matw =
{tmp(trunc(n*.10),1);tmp(trunc(n*.25),1);
tmp(trunc(n*.50),1); tmp(trunc(n*.75),1);
tmp(trunc(n*.90),1)}.
end if.
end if.
do if (wdich = 1).
compute matw = {cmin(w); cmax(w)}.
compute wvdich=1.
compute cmaw=cmax(w).
compute cminw=cmin(w).
end if.
do if ( 999 <> 999).
compute matw = 999.
compute quantd(1,1) = 0.
compute quantc(1,1) = 0.
end if.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm(1,1)=wname.
compute modmatp(1,1) = 1.
end if.
end if.
do if (zerr = 0 and zlist = 1).
do if (vnames(:,i)=zname).
compute zerr = 0.
compute zmatch = 1.
compute z = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(z)/n).
compute z = z-meanvec.
compute centvar = {centvar, zname}.

```

```

end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute zmean = csum(z)/n.
compute tmp = z-(cons*zmean).
compute zsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute zdich = 1.
loop jj = 1 to n.
do if ((z(jj,1) <> cmin(z)) and (z(jj,1) <>
cmax(z))).
compute zdich = 0.
break.
end if.
end loop.
do if (zdich = 0).
compute matz = {zmean-zsd; zmean;
zmean+zsd}.
do if ( 999 = 999 and quantile = 0).
do if (matz(1,1) < cmin(z)).
compute matz(1,1)=cmin(z).
compute minprobe=1.
end if.
do if (matz(3,1) > cmax(z)).
compute matz(3,1)=cmax(z).
compute maxprobe=1.
end if.
end if.
compute quantc(1,2) = 1.
do if (quantile = 1).
compute quantd(1,2) = 1.
compute quantc(1,2) = 0.
compute tmp = z.
compute tmp(GRADE(z)) = z.
compute matz = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (zdich = 1).
compute matz = {cmin(z); cmax(z)}.

```



```

compute cmaxz=cmax(z).
compute cminz=cmin(z).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matz = 999.
compute quantd(1,2) = 0.
compute quantc(1,2) = 0.
end if.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm(1,2)=zname.
compute modmatp(1,2) = 1.
end if.
end if.
do if (verr = 0 and vlist = 1).
do if (vnames(:,i)=vname).
compute verr = 0.
compute vmatch = 1.
compute v = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(v)/n).
compute v = v-meanvec.
compute centvar = {centvar, vname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute vmean = csum(v)/n.
compute tmp = v-(cons*vmean).
compute vsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute vdich = 1.
loop jj = 1 to n.
do if ((v(jj,1) <> cmin(v)) and (v(jj,1) <>
cmax(v))).
compute vdich = 0.
break.
end if.
end loop.
do if (vdich = 0).

```

```

compute matv = {vmean-vsd; vmean;
vmean+vsd}.
do if ( 999 = 999 and quantile = 0).
do if (matv(1,1) < cmin(v)).
compute matv(1,1)=cmin(v).
compute minprobe=1.
end if.
do if (matv(3,1) > cmax(v)).
compute matv(3,1)=cmax(v).
compute maxprobe=1.
end if.
end if.
compute quantc(1,3) = 1.
do if (quantile = 1).
compute quantd(1,3) = 1.
compute quantc(1,3) = 0.
compute tmp = v.
compute tmp(GRADE(v)) = v.
compute matv = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (vdich = 1).
compute matv = {cmin(v); cmax(v)}.
compute wvdich=1.
compute cmaxv=cmax(v).
compute cminv=cmin(v).
end if.
do if ( 999 <> 999).
compute matv = 999.
compute quantd(1,3) = 0.
compute quantc(1,3) = 0.
end if.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm(1,3)=vname.
compute modmatp(1,3) = 1.
end if.
end if.

```

```

do if (qerr = 0 and qlist = 1).
do if (vnames(:,i)=qname).
compute qerr = 0.
compute qmatch = 1.
compute q = dat(:,i).
do if (center = 1).
compute meanvec = make(n,1,csum(q)/n).
compute q = q-meanvec.
compute centvar = {centvar, qname}.
end if.
compute nvarch(1,i)=1.
compute nmods = nmods + 1.
compute qmean = csum(q)/n.
compute tmp = q-(cons*qmean).
compute qsd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute qdich = 1.
loop jj = 1 to n.
do if ((q(jj,1) <> cmin(q)) and (q(jj,1) <>
cmax(q))).
compute qdich = 0.
break.
end if.
end loop.
do if (qdich = 0).
compute matq = {qmean-qsd; qmean;
qmean+qsd}.
do if ( 999 = 999 and quantile = 0).
do if (matq(1,1) < cmin(q)).
compute matq(1,1)=cmin(q).
compute minprobe=1.
end if.
do if (matq(3,1) > cmax(q)).
compute matq(3,1)=cmax(q).
compute maxprobe=1.
end if.
end if.
compute quantc(1,4) = 1.
do if (quantile = 1).
compute quantd(1,4) = 1.
compute quantc(1,4) = 0.

```

```

compute tmp = q.
compute tmp(GRADE(q)) = q.
compute matq = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (qdich = 1).
compute matq = {cmin(q); cmax(q)}.
compute cmaxq=cmax(q).
compute cminq=cmin(q).
compute zqdich=1.
end if.
do if ( 999 <> 999).
compute matq = 999.
compute quantd(1,4) = 0.
compute quantc(1,4) = 0.
end if.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm(1,4)=qname.
compute modmatp(1,4) = 1.
end if.
end if.
do if (vnames(:,i)=cname).
compute cld = dat(:,i).
compute cvname = vnames(:,i).
compute nvarch(1,i)=1.
compute clsmtch = 1.
end if.
loop j = 1 to ncol(mnames).
do if (vnames(:,i)=mnames(1,j)).
compute mmatch = mmatch + 1.
compute m(:,j)=dat(:,i).
do if (center = 1 and nvarch(1,i) = 0 and mlist >
0 and mcm = 0).
compute meanvec = make(n,1,csum(m(:,j))/n).
compute m(:,j) = m(:,j)-meanvec.
compute centvar = {centvar, mnames(1,j)}.
end if.

```

```

compute nvarch(1,i)=1.
compute dichm = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute dichm = 0.
break.
end if.
end loop.
do if ((dichm = 1) and (olsdichm = 1)).
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.
do if (dichm = 1 and model > 3 and mcheck = 0
and olsdichm = 0).
compute errs= errs+1.
compute runerrs(errs,1) = 1.
compute mcheck = 1.
end if.
do if ((model <= 3) and (ncol(mnames)=1)).
compute nmods = nmods + 1.
compute mmean = csum(m(:,j))/n.
compute tmp = m(:,j)-(cons*mmean).
compute msd = sqrt((1/(n-1))*(t(tmp)*tmp)).
compute mdich = 1.
loop jj = 1 to n.
do if ((m(jj,j) <> cmin(m(:,j))) and (m(jj,j) <>
cmax(m(:,j)))).
compute mdich = 0.
break.
end if.
end loop.
do if (model = 1).
compute jndich=mdich.
compute jnmin=cmin(m(:,j)).
compute jnmax=cmax(m(:,j)).
end if.
do if (mdich = 0).
compute matm = {mmean-msd; mmean;
mmean+msd}.

```

```

do if ( 999 = 999 and quantile = 0).
do if (matm(1,1) < cmin(m(:,j))).
compute matm(1,1)=cmin(m(:,j)).
compute minprobe=1.
end if.
do if (matm(3,1) > cmax(m(:,j))).
compute matm(3,1)=cmax(m(:,j)).
compute maxprobe=1.
end if.
end if.
compute quantc(1,5) = 1.
do if (quantile = 1).
compute quantd(1,5) = 1.
compute quantc(1,5) = 0.
compute tmp = m(:,j).
compute tmp(GRADE(m(:,j))) = m(:,j).
compute matm = {tmp(trunc(n*.10),1);
tmp(trunc(n*.25),1); tmp(trunc(n*.50),1);
tmp(trunc(n*.75),1); tmp(trunc(n*.90),1)}.
end if.
end if.
do if (mdich = 1).
compute matm = {cmin(m); cmax(m)}.
end if.
do if ( 999 <> 999).
compute matm = 999.
compute quantd(1,5) = 0.
compute quantc(1,5) = 0.
end if.
compute modmatv(1,5)=nrow(matm).
compute modmat((1:nrow(matm)), 5) = matm.
compute modvnm(1,5)=mnames(1,j).
compute modmatp(1,5) = 1.
end if.
end if.
end loop.
end loop.
do if (minprobe=1).
compute note(notes,1) = 14.
compute notes = notes + 1.

```

```

end if.
do if (maxprobe=1).
compute note(notes,1) = 15.
compute notes = notes + 1.
end if.
do if (cname <> 'xxx' and clsmth = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 23.
end if.
do if (cname <> 'xxx').
do if ((cname = zname) or (cname = vname)
or (cname = wname) or (cname = xname) or
(cname = yname) or (cname = qname)).
compute errs = errs+1.
compute runerrs(errs,1) = 24.
end if.
end if.
do if (wlist = 1 and werr = 0 and wmatch = 0).
compute werr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 4.
end if.
do if (zlist = 1 and zerr = 0 and zmatch = 0).
compute zerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 5.
end if.
do if (qlist = 1 and qerr = 0 and qmatch = 0).
compute qerr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 6.
end if.
do if (vlist = 1 and verr = 0 and vmatch = 0).
compute verr = 3.
compute errs = errs+1.
compute runerrs(errs,1) = 7.
end if.
do if (yerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 16.

```

```

end if.
do if (xerr = 1).
compute errs = errs+1.
compute runerrs(errs,1) = 32.
end if.
do if (model = 6 and nmeds < 2).
compute errs = errs+1.
compute runerrs(errs,1) = 18.
end if.
do if (mmatch < ncol(mnames)).
compute errs=errs+1.
compute runerrs(errs,1) = 25.
end if.
do if (cname <> 'xxx').
compute cld = design(cld).
compute cluster = ncol(cld).
compute cld = cld(:,2:ncol(cld)).
compute clsdmy = ncol(cld).
do if (clsdmy > 19).
compute errs = errs+1.
compute runerrs(errs,1) = 26.
end if.
do if (mcx > 0 or mcm > 0).
compute errs = errs+1.
compute runerrs(errs,1) = 38.
end if.
end if.
compute dichy = 1.
loop jj = 1 to n.
do if ((y(jj,1) <> cmin(y)) and (y(jj,1) <>
cmax(y))).
compute dichy = 0.
break.
end if.
end loop.
do if ((dichy = 1) and (olsdichy = 1)).
compute dichy=0.
compute note(notes,1) = 18.
compute notes = notes + 1.
end if.

```



```

do if (dichy = 1).
do if (toteff=1).
compute toteff=0.
compute note(notes,1) = 31.
compute notes = notes + 1.
end if.
compute jncrit=xp2*xp2.
do if (ws = 1).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 28.
end if.
end if.
compute ncovs = ncol(dat)-rsum(nvarch).
do if (effsize = 1 and covmy <> 0 and ncovs >
0 and model > 3 and model < 7).
compute note(notes,1) = 22.
compute notes = notes + 1.
compute effsize = 0.
end if.
do if (ws = 1 and ncovs < nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 29.
end if.
do if (ws = 1 and ncovs > nmeds).
compute criterr = 1.
compute errs = errs+1.
compute runerrs(errs,1) = 30.
end if.
end if.
do if (errs = 0).
do if (rsum(quantd) > 0).
compute note(notes,1) = 4.
compute notes = notes+1.
end if.
do if (rsum(quantc) > 0).
compute note(notes,1) = 5.
compute notes = notes+1.
end if.

```

```

do if (ncovs > 0).
compute c = make(n,ncovs,0).
compute cnames = {'x'}.
compute j = 1.
loop i = 1 to ncol(vnames).
do if (nvarch(1,i)) = 0.
compute c(:,j) = dat(:,i).
compute nvarch(1,i)=1.
compute j=j+1.
compute cnames = {cnames, vnames(:,i)}.
end if.
end loop.
compute cnames = cnames(1,2:ncol(cnames)).
do if (ws = 1).
compute covmean=csum(c)/n.
loop i = 1 to ncovs.
compute meanvec = make(n,1, covmean(1,i)).
compute c(:,i) = c(:,i)-meanvec.
end loop.
compute centvar = {centvar, cnames}.
compute x=c(:,1:ncovs).
do if (ncovs=nmeds).
compute ncovs=0.
end if.
end if.
end if.
compute names = {yname, xname, mnames,
wname, zname, vname, qname}.
do if (ncovs > 0).
compute names = {names, cnames}.
end if.
do if (dichy = 1 and effsize = 1).
compute note(notes,1) = 2.
compute notes = notes+1.
end if.
do if (model > 3 and model < 6).
compute indeff=make(nmeds,1,0).
compute indboot=make(boot+1,nmeds,999).
do if (mc > 0).
compute indboot=make(mc+1,nmeds,999).

```

```

end if.
do if (effsize = 1 and dichy = 0).
compute rmeff=make(boot+1,nmeds+1,999).
compute abpseff=make(boot+1,nmeds+1,999).
compute abcseff=make(boot+1,nmeds+1,999).
compute pmeff=make(boot+1,nmeds+1,999).
compute r245 = make(boot+1,1,999).
compute kappa2 = make(boot+1,1,999).
end if.
end if.
do if (model = 6 and effsize = 1 and dichy = 0).
compute
rmeff=make(boot+1,ncol(indboot),999).
compute
abpseff=make(boot+1,ncol(indboot),999).
compute
abcseff=make(boot+1,ncol(indboot),999).
compute
pmeff=make(boot+1,ncol(indboot),999).
end if.
do if (nmods > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm(1,tmp) = modvnm(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:nmods).
compute modvnm=modvnm(:,1:nmods).
compute modmatv=modmatv(:,1:nmods).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.
loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.

```

```

compute modvals =
make((modmatv(1,1)*modprod(1,1)),
nmods,0).
loop i = 1 to nmods.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvals)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvals(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
do if (model = 74).
compute modvals=matx.
compute modvnm=xname.
end if.
compute vmat = make(8,nrow(modvals),0).
compute vmat(1,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute vmat(5,1:nrow(modvals)) =
make(1,nrow(modvals),1).
compute indeff=make(nrow(modvals),1,0).
do if (model <> 5).
compute indboot = make(((boot+1)*nmeds),
nrow(modvals), -99999999).
compute indbootp =
make(boot+1,(nmeds*nmods), -99999999).
do if (imm > 4 and imm < 8).
do if (imm = 7).
compute immmv1=matw.
compute immmv2=matv.
end if.
do if (imm = 5).
compute immmv1=matw.
compute immmv2=matz.
end if.

```

```

do if (imm = 6).
compute immmv1=matv.
compute immmv2=matq.
end if.
compute
indboop1=make((boot+1),(nmeds*(nrow(immv2))),-9999999).
compute
indboop2=make((boot+1),(nmeds*(nrow(immv1))),-9999999).
compute onesmv1=make(1,nrow(immmv1),1).
compute onesmv2=make(1,nrow(immmv2),1).
compute
nimvs=nrow(immmv1)+nrow(immmv2).
compute cmmval=0.
compute cmmlbs1={'xx'}.
compute cmmlbs2={'xx'}.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv2}.
compute
cmmlbt=make(nrow(immmv2),1,mnames(1,i)).
compute cmmlbs1={cmmlbs1;cmmlbt}.
end loop.
loop i = 1 to nmeds.
compute cmmval={cmmval;immmv1}.
compute
cmmlbt=make(nrow(immmv1),1,mnames(1,i)).
compute cmmlbs2={cmmlbs2;cmmlbt}.
end loop.
compute cmmval=cmmval(2:nrow(cmmval),:).
end if.
end if.
end if.
do if (nmods> 0).
loop i = 1 to ncol(modvals).
do if (modvnm(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm(1,i)=zname).
compute zcol = i.

```

```

end if.
do if (modvnm(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm(1,i)=qname).
compute qcol = i.
end if.
end loop.
end if.
do if (dichy = 1).
compute omx = cmax(y).
compute omn = cmin(y).
compute y = (y = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute data = {cons,y,m,x}.
compute datamed = data.
compute datayed = data.
compute datanm = {'constant'; yname;
t(mnames); xname}.
compute datanmm = {'constant'; yname;
t(mnames); xname}.
compute datanmy = {'constant'; yname;
t(mnames); xname}.
compute yintkey = {' ',' ',' ',' ',' ',' '}.
do if (model < 4 and errs = 0).
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' ', ' '}.
compute datayed = {datayed, x&*m}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(1,i) = 1.
compute vmat(2,i) = modvals(i,1).
end loop.
compute mmat = make(16,nmeds,1).
end if.
do if (model = 2 or model = 3).
compute int1 = x&*w.

```

```

compute datayed = {datayed, w, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(2,i) = modvals(i,2).
compute vmat(3,i) = modvals(i,1).
compute vmat(4,i) = modvals(i,1)*modvals(i,2).
end loop.
end if.
do if (model = 3).
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w&*m.
compute int2 = x&*w&*m.
compute datayed = {datayed, int1, int2}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', mnames, ' X', wname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (model = 4 or model = 5).
compute vmat = make(8,1,1).
end if.
compute yintkey2 = yintkey.
do if (wm = 1).
compute int1 = x&*w.
compute datamed = {datamed, w,int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmm = {datanmm; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).

```

```

compute vmat(2,i) = modvals(i,wcol).
end loop.
do if (zm = 1).
compute int1 = x*z.
compute datamed = {datamed, z, int1}.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
loop i = 1 to nrow(modvals).
compute vmat(3,i) = modvals(i,zcol).
end loop.
end if.
do if (wzm = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmm = {datanmm;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute int1 = w*z.
compute int2 = x*w*z.
compute datamed = {datamed, int1, int2}.
loop i = 1 to nrow(modvals).
compute vmat(4,i) =
modvals(i,wcol)*modvals(i,zcol).
end loop.
end if.
end if.
compute mdatacol=ncol(datamed).
compute mintkey = yintkey.
compute yintkey = {' ', ' ', ' ', ' ', ' ', ' '}.
compute medints = intcnt-1.
do if (vy = 1 or xmy = 1).
compute mp = 1.

```



```

do if (xmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,1).
end loop.
end if.
do if (vy = 1).
compute datayed = {datayed, v}.
compute datanmy = {datanmy; vname}.
compute mmods = 1.
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,vcol).
end loop.
do if (qy = 1).
compute mp = 2.
compute datayed = {datayed,q}.
compute datanmy = {datanmy; qname}.
compute mmods = 2.
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,qcol).
end loop.
end if.
do if (vqy = 1).
compute mp = 3.
compute datayed = {datayed, v*&q}.
compute mmods = 3.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,vcol)*modvals(i,qcol).
end loop.
end if.
end if.
compute mints = make(n,(nmeds*mp),0).
loop i = 0 to (nmeds-1).
do if (i = 0 and vqy = 1).
compute yintkey = {yintkey; yintemp(1,intcnt),
vname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.

```

```

do if (vy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (xmy = 1).
compute mints(:,((i*mp)+1))= m(:,(i+1))*x.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', xname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (qy = 1).
compute mints(:,((i*mp)+2))=m(:,(i+1))*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (vqy = 1).
compute mints(:,((i*mp)+3))=m(:,(i+1))*v*q.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', vname, ' X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end loop.
compute datayed = {datayed, mints}.
end if.
compute mp = 1.
do if (wvmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).

```

```

compute vmat(8,i) =
modvals(i,wcol)*modvals(i,vcol).
end loop.
end if.
compute mints2 = make(n,(nmeds*mp),0).
do if (wmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(7,i) = modvals(i,wcol).
end loop.
do if (wy = 0 and model > 3).
compute datayed = {datayed, w}.
compute datanmy = {datanmy; wname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wvmy = 1).
compute datayed = {datayed,w&*v}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints2(:,((i*mp)+1))= m(:,(i+1))&*w.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wvmy = 1).
compute mints2(:,((i*mp)+2))=
m(:,(i+1))&*w&*v.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints2}.
end if.

```

```

compute mp = 1.
do if (zmy = 1).
loop i = 1 to nrow(modvals).
compute vmat(6,i) = modvals(i,zcol).
end loop.
do if (wzmy = 1).
compute mp = 2.
loop i = 1 to nrow(modvals).
compute vmat(8,i) =
modvals(i,zcol)&*modvals(i,wcol).
end loop.
end if.
end if.
do if (zmy = 1).
compute mints3 = make(n,(nmeds*mp),0).
do if (zy = 0).
compute datayed = {datayed, z}.
compute datanmy = {datanmy; zname}.
end if.
loop i = 0 to (nmeds-1).
do if (i = 0 and wzmy = 1 and wzy = 0).
compute datayed = {datayed,w&*z}.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute mints3(:,((i*mp)+1))= m(:,(i+1))&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', zname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (wzmy = 1).
compute mints3(:,((i*mp)+2))=
m(:,(i+1))&*w&*z.
compute yintkey = {yintkey; yintemp(1,intcnt),
mnames(1,(i+1)), ' X', wname, ' X', zname}.

```

```

compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end loop.
compute datayed = {datayed, mints3}.
end if.
compute decoc = 1.
compute modmat = make(5,5,999).
compute modmatv = make(1,5,1).
compute modmatp = make(1,5,0).
compute modprod = modmatv.
do if (wy = 1 and model > 3).
compute datayed = {datayed, w, x&*w}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,1)=nrow(matw).
compute modmat((1:nrow(matw)), 1) = matw.
compute modvnm2(1,1)=wname.
compute modmatp(1,1) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' ', ' '}.
compute datanmy = {datanmy; wname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (zy = 1).
compute datayed = {datayed,z,x&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,2)=nrow(matz).
compute modmat((1:nrow(matz)), 2) = matz.
compute modvnm2(1,2)=zname.
compute modmatp(1,2) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', zname, ' ', ' '}.
compute datanmy = {datanmy; zname;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.

```

```

do if (wzy = 1).
compute datayed = {datayed,w&*z,x&*w&*z}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
wname, ' X', zname, ' ', ' '}.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, ' X', zname }.
compute datanmy =
{datanmy;yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
do if (vxy = 1).
compute datayed = {datayed, x&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,3)=nrow(matv).
compute modmat((1:nrow(matv)), 3) = matv.
compute modvnm2(1,3)=vname.
compute modmatp(1,3) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
do if (qxy = 1).
compute datayed = {datayed, x&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute modmatv(1,4)=nrow(matq).
compute modmat((1:nrow(matq)), 4) = matq.
compute modvnm2(1,4)=qname.
compute modmatp(1,4) = 1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', qname, ' ', ' '}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.

```

```

compute intcnt = intcnt+1.
do if (vqxy = 1).
compute datayed = {datayed, x&*v&*q}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', vname, 'X', qname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
end if.
end if.
do if (wvxy = 1).
compute datayed = {datayed, x&*w&*v}.
compute decoc = decoc+1.
compute deco(decoc,1) = ncol(datayed)-1.
compute yintkey = {yintkey; yintemp(1,intcnt),
xname, ' X', wname, 'X', vname}.
compute datanmy = {datanmy;
yintemp(1,intcnt)}.
compute intcnt = intcnt+1.
end if.
compute modvalsd = 0.
compute ttt = rsum(modmatp).
do if (rsum(modmatp) > 0).
compute tmp = 1.
loop i = 1 to 5.
do if (modmatp(1,i) = 1).
compute modmat(:,tmp) = modmat(:,i).
compute modvnm2(1,tmp) = modvnm2(1,i).
compute modmatv(1,tmp) = modmatv(1,i).
compute tmp=tmp+1.
end if.
end loop.
compute modmat=modmat(:,1:ttt).
compute modvnm2=modvnm2(:,1:ttt).
compute modmatv=modmatv(:,1:ttt).
loop i = 1 to (ncol(modmatv)-1).
compute tmp = 1.

```

```

loop j = (i+1) to ncol(modmatv).
compute tmp = tmp*modmatv(1,j).
end loop.
compute modprod(1,i)=tmp.
end loop.
compute modvalsd =
make((modmatv(1,1)*modprod(1,1)), ttt,0).
loop i = 1 to ttt.
compute strt = 1.
compute fnsh=0.
loop if (fnsh < nrow(modvalsd)).
loop j = 1 to modmatv(1,i).
compute
tmp=make(modprod(1,i),1,modmat(j,i)).
compute fnsh = fnsh+nrow(tmp).
compute modvalsd(strt:fnsh, i) = tmp.
compute strt = fnsh+1.
end loop.
end loop.
end loop.
end if.
do if (ttt > 0).
loop i = 1 to ncol(modvalsd).
do if (modvnm2(1,i)=wname).
compute wcol = i.
end if.
do if (modvnm2(1,i)=zname).
compute zcol = i.
end if.
do if (modvnm2(1,i)=vname).
compute vcol = i.
end if.
do if (modvnm2(1,i)=qname).
compute qcol = i.
end if.
end loop.
compute directv = make(nrow(modvalsd),1,1).
do if (wy = 1).
compute directv ={directv, modvalsd(:,wcol)}.
end if.

```



```

do if (zy = 1).
compute directv = {directv, modvalsd(:,zcol)}.
end if.
do if (wzy = 1).
compute directv = {directv,
(modvalsd(:,wcol)*modvalsd(:,zcol))}.
end if.
do if (vxy = 1).
compute directv = {directv, modvalsd(:,vcol)}.
end if.
do if (qxy = 1).
compute directv = {directv, modvalsd(:,qcol)}.
end if.
do if (vqxy = 1).
compute directv = {directv,
(modvalsd(:,vcol)*modvalsd(:,qcol))}.
end if.
do if (wvxy = 1).
compute directv = {directv,
(modvalsd(:,vcol)*modvalsd(:,wcol))}.
end if.
end if.
compute ydatacol = ncol(datayed).
do if (ncovs > 0).
do if (covmy <> 2).
compute datamed = {datamed,c}.
end if.
do if (covmy <> 1).
compute datayed = {datayed,c}.
end if.
do if (covmy = 1 and model = 4 and mcx > 0).
compute datayed = {datayed,c}.
end if.
compute covmeans = csum(c)/n.
end if.
do if (cluster > 0).
compute datamed = {datamed,cld}.
compute datayed = {datayed, cld}.
compute cldmeans = csum(cld)/n.
end if.

```

```

compute mst = 3.
compute mnd = mst+nmeds-1.
compute ydatacol=ncol(datayed).
compute mdatacol=ncol(datamed).
do if (ncovs > 0).
compute datanmy = {datanmy; t(cnames)}.
do if (model > 3).
compute datanmm = {datanmm; t(cnames)}.
end if.
end if.
compute datanmy = {'constant';
datanmy(3:nrow(datanmy),1)}.
do if (model > 3).
compute datanmm = {'constant';
datanmm(3:nrow(datanmm),1)}.
end if.
compute amm = make(2,1,0).
compute abmm = make(2,1,0).
compute mnv = csum(datayed(:,2)/n).
compute mnv = make(n,1,mnv).
compute ssty = csum((datayed(:,2)-mnv)**2).
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
do if (ws=1).
compute stddevm=diag(sigma).
compute
stddevm=sqrt(stddevm(3:(nmeds+2),1)).
end if.
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute r2xy =
(sigma(2,(3+nmeds))/(stddevy*stddevx))**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))**2.
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nmeds)).

```

```

do if (model = 4 and nmeds = 1 and cluster = 0
and ncovs = 0).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
compute datatm = datamed.
compute dataty = datayed.
compute mdlrms2 = { '1' ; yname; xname}.
compute mdlrms = {'Model ='; ' Y ='; ' X ='
}.
do if (ws = 1).
compute mdlrms2 = { '1' ; yname}.
compute mdlrms = {'Model ='; ' Y ='}.
end if.
loop i = 1 to ncol(mnames).
compute mdlrms2 = {mdlrms2; mnames(1,i)}.
do if (i = 1 and ncol(mnames) = 1).
compute mdlrms = {mdlrms; ' M ='}.
else.

```

```

compute mdlInms = {mdlInms; mlab(i,1)}.
end if.
end loop.
do if (wname <> 'xxx').
compute mdlInms2 = {mdlInms2; wname}.
compute mdlInms = {mdlInms; '  W = '}.
end if.
do if (zname <> 'xxx').
compute mdlInms2 = {mdlInms2; zname}.
compute mdlInms = {mdlInms; '  Z = '}.
end if.
do if (vname <> 'xxx').
compute mdlInms2 = {mdlInms2; vname}.
compute mdlInms = {mdlInms; '  V = '}.
end if.
do if (qname <> 'xxx').
compute mdlInms2 = {mdlInms2; qname}.
compute mdlInms = {mdlInms; '  Q = '}.
end if.
do if (jn = 1 and model = 1 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
do if (jn = 1 and model = 3 and jndich = 1).
compute note(notes,1) = 8.
compute notes = notes + 1.
end if.
compute mdpbe=0.
compute medte2=0.
do if ( 1 =4 and mcx > 0).
compute medte2=1.
end if.
do if (((mcx > 0) or (mcm > 0)) AND model =
1).
compute mdpbe=1.
compute dd={datayed(:,4),datayed(:,3)}.
do if (ncovs > 0).
compute dd={datayed(:,6:(5+ncovs)),dd}.
end if.
compute dd={datayed(:,2),dd}.

```

```

end if.
compute yes = ((nrow(modvals)=1) and
(contrast = 1) and (model = 3 or model = 2)).
compute yes2 = ((nrow(modvals)=1) and
(contrast = 1) and (model = 2)).
print/title = '***** PROCESS Procedure
for SPSS Release 2.16.3 *****'.
print/title = '      Written by Andrew F. Hayes,
Ph.D.    www.afhayes.com'.
print/title = '  Documentation available in
Hayes (2013).
www.guilford.com/p/hayes3'/space=0.
print mdlnms2/title =
*****
*****'/rnames = mdlnms/format =
a8.
do if (ncovs > 0).
print cnames/title = 'Statistical Controls: '/rlabels
= 'CONTROL='/format a8.
end if.
print n/title = 'Sample size'/format F10.0.
do if (cluster > 0).
print cluster/rnames = cvname/title =
'Clustering Variable and Number of Clusters'.
end if.
do if (model > 3 and ( 'random' <> 'random')).
compute seedt= 'random'.
print seedt/title='Custom'/format=A12/rlabel =
'Seed:'.
end if.
do if (mdpbe=0 and medte2 = 0).
loop bt = 1 to (boot+1).
do if (bt = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef = 0.
do if (bt > 1).
loop.
compute v=trunc(uniform(n,1)*n)+1.

```

```

compute datayed = dataty(v,:).
do if (ws=1).
compute
bmeans=csum(datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed)))/n.
compute cntmat=make(n,nmeds,1).
compute bmeans2=bmeans.
do if (nmeds > 1).
compute bmeans2=mdia(bmeans).
end if.
compute cntmat=cntmat*bmeans2.
compute datayed(:,(ncol(datayed)-
nmeds+1):ncol(datayed))=datayed(:,(ncol(data
yed)-nmeds+1):ncol(datayed))-cntmat.
end if.
compute detcheck=det(t(datayed)*datayed).
compute rk=(detcheck <> 0).
do if (model > 3).
compute datamed = datatm(v,:).
compute detcheck=det(t(datamed)*datamed).
do if (rk=1).
compute rk=(detcheck <> 0).
end if.
end if.
compute sigma = (n*sscp(datayed))-
(t(csum(datayed))*(csum(datayed))).
compute sigma = sigma/(n*(n-1)).
compute temp=diag(sigma).
compute bad = bad+(1-rk).
compute false = 1.
end loop if (rk = 1).
compute stddevy = sqrt(sigma(2,2)).
compute stddevx =
sqrt(sigma((3+nmeds),(3+nmeds))).
compute ctot =
sigma(2,(3+nmeds))/sigma((3+nmeds),(3+nme
ds)).
do if (model = 4 and nmeds = 1 and ncovs = 0
and cluster = 0).

```

```

compute r2xy =
(sigma(2,4)/(stddevy*stddevx))&**2.
compute r2my =
(sigma(2,3)/(stddevy*sqrt(sigma(3,3))))&**2.
compute sstot = sigma(2,2)*(n-1).
compute kappaa = sigma(2,3)*sigma(2,4).
compute kappab =
sqrt((sigma(3,3)*sigma(2,2))-
(sigma(2,3)*sigma(2,3))).
compute kappac =
sqrt((sigma(4,4)*sigma(2,2))-
(sigma(2,4)*sigma(2,4))).
compute kappad = sigma(4,4)*sigma(2,2).
compute kappae =
sqrt((sigma(4,4)*sigma(3,3))-
(sigma(3,4)*sigma(3,4))).
compute amm(1,1) =
(kappaa+(kappab*kappac))/kappad.
compute amm(2,1) = (kappaa-
(kappab*kappac))/kappad.
do if (sigma(3,4) < 0).
compute amma =cmin(amm).
end if.
do if (sigma(3,4) > 0).
compute amma = cmax(amm).
end if.
compute abmm(1,1)=-amma*(kappac/kappae).
compute abmm(2,1)=amma*(kappac/kappae).
end if.
end if.
do if (model > 3).
loop im = 1 to nmeds.
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant';
datanmm((2+nmeds):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
do if (model = 6).

```

```

do if (im = 1).
compute xm={cons,
datamed(:,(mnd+1):mdatacol)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
end if.
do if (im > 1).
compute xm={cons, datamed(:,3:(im+1)),
datamed(:,(mnd+1):mdatacol)}.
compute xnm = {'constant'; datanmm(2:im,1);
datanmm((mnd):nrow(datanmm),1)}.
compute invXtX = inv(t(xm)*xm).
compute coeff
=invXtX*t(xm)*datamed(:,(2+im)).
compute
mmpaths((im+1),(2:im))=t(coeff(2:im,1)).
end if.
end if.
do if (ws = 1).
compute
coeff(1,1)=csum(datamed(:,(2+im)))/n.
end if.
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
do if (bt = 1).
compute resid=datamed(:,(2+im))-xm*coeff.
compute sse =cssq(resid).
compute mse = sse/(n-ncol(xm)).
compute mnv = csum(data(:,(2+im)))/n.
compute mnv = make(n,1,mnv).
compute sstm = csum((data(:,(2+im))-
mnv)**2).
compute k3 = nrow(coeff).
do if (hc3 = 1).
compute h = xm(:,1).
loop i3=1 to n.
compute h(i3,1)= xm(i3,:)*invXtX*t(xm(i3,:)).
end loop.

```



```

loop i3=1 to k3.
compute xm(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xm(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xm(:,i3) = sqrt(mse)&*xm(:,i3).
end loop.
end if.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute hccov=invXtX*t(xm)*xm*invXtX.
compute mcmats(im,im)=hccov(2,2).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*hccov*lmat)*((t(lmat
)*coeff)))/dfnum.
compute coeff = coeff(1:(nrow(coeff)-
clsdmy),1).
compute mccoeff(im,1)=coeff(2,1).
compute standerr =
sqrt(diag(invXtX*t(xm)*xm*invXtX)).
compute standerr = standerr(1:(nrow(standerr)-
clsdmy),1).
do if (ws=1).
compute standerr(1,1)=stddevm(im,1)/sqrt(n).
end if.
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-
ncol(xm)))).
compute temp=(n-ncol(xm)).
do if (ws=1).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute temp=(n-1).
end if.
compute xd = abs(xp2).

```

```

compute temp = (temp* (exp((temp-
(5/6))*((xd/(temp-(2/3)+(.11/temp))))*(xd/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = coeff-
sqrt(abs(temp))*standerr.
compute temp2 =
coeff+sqrt(abs(temp))*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (ws=1).
compute op=op(1,:).
compute xnmn =
apathnam(((nmeds>1)+1+(im-1)),1).
end if.
compute sobel(im,1) = coeff(2,1).
compute sobel(im,2)=standerr(2,1).
compute temp = mnames(1,im).
compute r2full = 1-(sse/sstm).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute summ = {sqrt(r2full), r2full, mse,
Fratio, dfnum, dfden, pfr}.
do if (detail = 1).
print temp/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.
do if (ws <> 1).
print summ/title = 'Model Summary'/clabels 'R',
'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'/'format = F10.4.
end if.
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
print op/title = 'Model'/rnames = xnmn/clabels
'coeff', 'se', 't', 'p', 'LLCI', 'ULCI'/'format = F10.4.
do if (covcoeff=1 and ws <> 1).
compute
hccovtmp=hccov(1:nrow(op),1:nrow(op)).
compute cnamestp=t(xnmn).

```

```

print hccovtmp/title='Covariance matrix of
regression parameter
estimates'/rnames=xmnm/cnames=cnamestp/f
ormat= F10.4.
end if.
do if (nmods > 0 and nrow(mintkey) > 1).
print mintkey/title = 'Product terms key:.'/format
= A8.
end if.
do if (ws = 1).
compute wsdf=n-1.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
end if.
end if.
compute ymat(1,im) = coeff(2,1).
do if (ws = 1).
compute ymat(1,im)=coeff(1,1).
end if.
do if (wm = 1).
compute ymat(2,im) = coeff(4,1).
do if (zm = 1).
compute ymat(3,im) = coeff(6,1).
do if (wzm = 1).
compute ymat(4, im) = coeff(8,1).
end if.
end if.
end if.
do if (model = 6).
compute mmpaths((im+1),1)=coeff((im+1),1).
end if.
end loop.
end if.
loop totlp = 1 to (1+(toteff*(bt = 1))).
do if (toteff = 1 and totlp =2).
compute xy =
{cons,datayed(:,(3+nmeds):ydatacol)}.
else.
compute xy={cons, datayed(:,3:ydatacol)}.
end if.

```

```

do if (dichy = 1).
compute pt2 =
make(nrow(datayed(:,2)),1,(csum(datayed(:,2))
/n)).
compute LL3 = datayed(:,2)*ln(pt2)+(1-
datayed(:,2))*ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xy),1,0).
compute LL1 = 0.
compute xy22=xy.
loop jjj = 1 to iterate.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.
compute coeff =
bt1+inv(t(xy22)*xy)*t(xy)*(datayed(:,2)-pt1).
compute pt1 = 1/(1+exp(-(xy*coeff))).
compute itprob = csum((pt1 <
.00000000000001) or (pt1 >
.99999999999999)).
do if (itprob = 0).
compute LL = datayed(:,2)*ln(pt1)+(1-
datayed(:,2))*ln(1-pt1).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
break.
end if.
compute bt1 = coeff.
compute LL1 = LL2.
end loop.
do if (jjj >= iterate and iterr = 0).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
compute iterr = 1.
end if.
loop ijk=1 to ncol(xy).
compute xy22(:,ijk)=xy(:,ijk)*pt1*(1-pt1).
end loop.

```

```

compute covmat = inv(t(xy22)*xy).
release xy22.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
end if.
do if (dichy = 0).
compute invXtX = inv(t(xy)*xy).
compute coeff =invXtX*t(xy)*datayed(:,2).
do if (effsize=1 and model > 3 and model < 7
and (ncovs > 0 or clsdmy > 0)).
compute xysdy =
{cons,datayed(:,(4+nmeds):ydatacol)}.
compute coeffsd
=inv(t(xysdy)*xysdy)*t(xysdy)*datayed(:,2).
compute resid=datayed(:,2)-(xysdy*coeffsd).
compute sdycov =sqrt(csum(resid&*resid)/(n-
ncol(xysdy))).
compute xvaron=datayed(:,(3+nmeds)).
compute
coeffx2=inv(t(xysdy)*xysdy)*t(xysdy)*xvaron.
compute residx2=xvaron-(xysdy*coeffx2).
compute sdxcov
=sqrt(csum(residx2&*residx2)/(n-ncol(xysdy))).
end if.
do if (totlp <> 2).
compute
bootcoef={bootcoef,t(coeff(1:(nrow(coeff)-
clsdmy),1))}.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4 and bt > 1).
compute resid=datayed(:,2)-xy*coeff.
compute sse = cssq(resid).
compute r2full = 1-(sse/sstot).
end if.
do if (bt =1).
compute resid=data(:,2)-xy*coeff.

```

```

compute k3 = nrow(coeff).
compute sse = cssq(resid).
compute mse = sse/(n-ncol(xy)).
compute
hcnor=coeff&/(sqrt(diag(mse*invXtX))).
do if (hc3 = 1).
compute h = xy(:,1).
loop i3=1 to n.
compute h(i3,1)= xy(i3,:)*invXtX*t(xy(i3,:)).
end loop.
loop i3=1 to k3.
compute xy(:,i3) = (resid(:,ncol(resid))&/(1-
h))*xy(:,i3).
end loop.
end if.
do if (hc3 <> 1).
loop i3=1 to k3.
compute xy(:,i3) = sqrt(mse)*xy(:,i3).
end loop.
end if.
compute covmat = (invXtX*t(xy)*xy*invXtX).
end if.
end if.
do if (bt = 1).
do if (model = 2).
compute xy2={cons, datayed(:,3:ydatacol)}.
compute temp = ncol(xy2).
do if (temp > 6).
compute xy3=xy2(:,7:temp)}.
end if.
compute xy2={xy2(:,1:3), xy2(:,5)}.
do if (temp > 6).
compute xy2={xy2, xy3}.
release xy3.
end if.
compute invXtX = inv(t(xy2)*xy2).
compute coeff2 =invXtX*t(xy2)*datayed(:,2).
compute ssem2=cssq(datayed(:,2)-xy2*coeff2).
release xy2.
end if.

```

```

compute stderr = sqrt(diag(covmat)).
do if (totlp = 1).
compute
mcmats((nmeds+1):ncol(mcmats),(nmeds+1):n
col(mcmats))=covmat(2:(1+nmeds),2:(1+nmed
s)).
end if.
compute stderr=stderr(1:(nrow(stderr)-
clsdmy),1).
do if (ws=1 and totp=2).
compute stderr(1,1)=stddevy/sqrt(n).
end if.
compute coeffplt = coeff.
compute lmat = ident(nrow(coeff)).
compute lmat = lmat(:,2:ncol(lmat)).
compute dfnum = nrow(coeff)-1.
compute dfden = n-dfnum-1.
compute fratio =
(t(t(lmat)*coeff)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*coeff)))/dfnum).
compute coeff=coeff(1:(nrow(coeff)-clsdmy),1).
do if (totlp=1).
compute
mccoeff((nmeds+1):nrow(mccoeff))=coeff(2:(1+
nmeds),1).
end if.
compute bbbb=coeff(2,1).
do if totp = 1.
compute deco(1,1)=2+nmeds.
compute deco = deco(1:decoc,1).
compute covdirt = make((nrow(covmat)-
clsdmy),(ncol(covmat)-clsdmy),0).
compute covdirt = covmat(deco,:).
compute covdir =
make(nrow(covdirt),ncol(covdirt),0).
compute covdir = covdirt(:,t(deco)).
compute deco=coeff(deco,1).
do if (ttt > 0).
compute sedir =
sqrt(diag(directv*covdir*t(directv))).

```

```

compute directv = directv*deco.
end if.
compute sobel(:,3)=coeff(2:(1+nmeds),1).
compute sobel(:,4) = standerr(2:(1+nmeds),1).
compute sobel2 = sobel&*sobel.
do if (varorder <> 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)&*sobel2(:,4)+sobel2(:,3)&*sob
el2(:,2)).
end if.
do if (varorder = 2).
compute sobel(:,2) =
sqrt(sobel2(:,1)&*sobel2(:,4)+sobel2(:,3)&*sob
el2(:,2)+sobel2(:,2)&*sobel2(:,4)).
end if.
compute sobel(:,1) = sobel(:,1)&*sobel(:,3).
compute sobel(:,3) = sobel(:,1)&/sobel(:,2).
compute sobel(:,4) = 2*(1-
cdfnorm(abs(sobel(:,3)))).
end if.
do if (dichy = 0).
compute tratio = coeff&/standerr.
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy)))).
do if (ws= 1 and totlp=2).
compute p = 2*(1-tcdf(abs(tratio), (n-1))).
compute dfden=n-1.
end if.
compute cnms = {'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute op = {coeff, standerr, tratio, p}.
end if.
do if (dichy = 1).
compute tratio = (coeff&/standerr).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute wald = tratio&*tratio.
compute cnms = {'coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute temp=coeff-abs(xp2)*standerr.
compute op = {coeff, standerr, tratio, p, temp}.
compute temp=coeff+abs(xp2)*standerr.

```



```

compute op = {op, temp}.
end if.
do if (detail = 1).
do if (totlp = 2).
print yname/title = '*****'
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:.'/format = A8.
end if.
do if (totlp <> 2).
print yname/title =
*****
*****'/rlabels = 'Outcome:.'/format
= A8.
end if.
end if.
do if (dichy = 1 and bt = 1 and totp = 1).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:.'/cnames = nmsd/format = F9.2.
end if.
do if (dichy = 0).
compute r2full = 1-(sse/ssty).
compute pfr = 1-fcdf(fratio,dfnum,dfden).
compute jndf=dfden.
compute wsdf=dfden.
compute xd = abs(xp2).
compute jncrit = (dfden* (exp((dfden-
(5/6))*((xd/(dfden-
(2/3)+(.11/dfden)))*(xd/(dfden-
(2/3)+(.11/dfden)))))-1)).
compute summ = {sqrt(r2full), r2full, mse,
fratio, dfnum, dfden, pfr}.
compute temp1=coeff-sqrt(jncrit)*standerr.
compute temp2=coeff+sqrt(jncrit)*standerr.
compute op = {coeff, standerr, tratio, p, temp1,
temp2}.
do if (detail = 1).
do if (ws <> 1).

```

```

print summ/title = 'Model Summary'/clabels =
'R', 'R-sq', 'MSE' 'F', 'df1', 'df2', 'p'/format =
F10.4.
end if.
end if.
end if.
do if (dichy = 1).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(coeff)-
1)).
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
do if (detail = 1).
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format =
F10.4.
end if.
end if.
do if (totlp = 2).
compute datanmy={'constant';
datanmy((nmeds+2):nrow(datanmy),1)}.
end if.
do if (detail = 1).
do if (coeffci = 0).
compute op = op(:,1:(ncol(op)-2)).
end if.
do if (ws=1).
compute op2=op(1:(1+nmeds),:).
compute datanmy2 = {'c" path'; bpathnam(
(1+(nmeds>1)): ((1+(nmeds>1))+(nmeds-
1)),1)}.
do if (totlp=2).
compute datanmy2={'c path'}.
compute op2=op(1,:).
compute wsdf=n-1.
end if.

```

```

print op2/title = 'Model'/rnames =
datanmy2/cnames = cnms/format = F10.4.
print wsdf/title=' '/rlabels = 'df = '/space=0.
end if.
do if (ws <> 1).
print op/title = 'Model'/rnames =
datanmy/cnames = cnms/format = F10.4.
do if (covcoeff=1).
compute
covmattp=covmat(1:nrow(op),1:nrow(op)).
compute cnamestp=t(datanmy).
print covmattp/title='Covariance matrix of
regression parameter
estimates'/rnames=datanmy/cnames=cnamest
p/format= F10.4.
end if.
end if.
end if.
do if (ttt = 0 and totlp = 1).
compute deco = op((nmeds+2),:).
do if (ws = 1).
compute deco=op(1,:).
end if.
end if.
do if (ttt = 0 and totlp = 2).
compute decotot = op(2,:).
do if (ws = 1).
compute decotot=op(1,:).
end if.
end if.
do if (nmods > 0 and model > 4 and detail = 1
and nrow(yintkey) > 1)).
print yintkey/title = 'Product terms key:'/format
= A8.
end if.
do if (nmods > 0 and model < 4 and detail = 1).
print yintkey2/title = 'Product terms key:'/format
= A8.
do if ((model = 1 or model = 2) and dichy = 0).

```

```

compute temp={((op(4,3)**2)*(1-r2full))/dfden,
op(4,3)**2, 1, dfden, op(4,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(4,1)**2)*(1-
r2full))/dfden.
end if.
compute rnms=yintkey2(2,1).
do if (model = 2).
compute temp={temp;((op(6,3)**2)*(1-
r2full))/dfden, op(6,3)**2, 1, dfden, op(6,4)}.
do if (hc3=1).
compute temp(2,1)=((hcnot(6,1)**2)*(1-
r2full))/dfden.
end if.
compute frat2=(dfden*(r2full-(1-
(ssem2/ssty))))/(2*(1-r2full)).
do if (hc3=1).
compute lmat=make(nrow(coeffplt),2,0).
compute lmat(4,1)=1.
compute lmat(6,2)=1.
compute frat2 =
(t(t(lmat)*coeffplt)*inv(t(lmat)*covmat*lmat)*((t(l
mat)*coeffplt)))/2).
end if.
compute temp={temp;(r2full-(1-
(ssem2/ssty))),frat2,2,dfden,1-
fcdf(frat2,2,dfden)}.
compute rnms={rnms; yintkey2(3,1),'Both'}.
end if.
print temp/title = 'R-square increase due to
interaction(s):'/rnames=rnms/clabels = 'R2-
chng', 'F', 'df1', 'df2', 'p'/format = F10.4.
end if.
do if (model = 3 and dichy = 0).
compute temp={((op(8,3)**2)*(1-r2full))/dfden,
op(8,3)**2, dfden, op(8,4)}.
do if (hc3=1).
compute temp(1,1)=((hcnot(8,1)**2)*(1-
r2full))/dfden.
end if.

```

```

compute rnms=yintkey2(5,1).
print temp/title = 'R-square increase due to
three-way interaction:.'/rnames=rnms/clabels =
'R2-chng', 'F(1,df2)', 'df2', 'p'/format = F10.4.
end if.
end if.
end if.
do if (model = 6 and totlp = 1).
compute
mmpaths(nrow(mmpaths),1)=coeff(nrow(mmpa
ths),1).
compute
mmpaths(nrow(mmpaths),(2:(nmeds+1)))=t(co
eff(2:(nmeds+1),1)).
end if.
do if (totlp = 1).
loop im = 1 to nmeds.
do if (model < 4).
compute ymat(1,im) = coeff(3,1)*(1-yes).
compute ymat(2,im) = coeff(4,1).
compute cmat(1,im) = covmat(3,3).
compute cmat(2,im) = covmat(4,4).
compute cmat(5,im) = covmat(3,4).
compute jnb1=coeff(3,1).
compute jnb3=coeff(4,1).
compute jnsb1=covmat(3,3).
compute jnsb3=covmat(4,4).
compute jnsb1b3=covmat(3,4).
do if (model = 2 or model = 3).
compute ymat(3,im) = coeff(6,1).
compute cmat(3,im) = covmat(6,6).
compute cmat(6,im) = covmat(3,6).
compute cmat(8,im) = covmat(4,6).
end if.
do if (model = 3).
compute ymat(4,im) = coeff(8,1).
compute cmat(4,im) = covmat(8,8).
compute cmat(7,im) = covmat(3,8).
compute cmat(9,im) = covmat(4,8).
compute cmat(10,im) = covmat(6,8).

```

```

compute jnb1=coeff(4,1).
compute jnb3=coeff(8,1).
compute jnsb1=covmat(4,4).
compute jnsb3=covmat(8,8).
compute jnsb1b3=covmat(4,8).
end if.
end if.
do if (model > 3).
compute ymat(5,im) = coeff((1+im),1).
end if.
do if (xmy = 1).
compute ymat(6,im) = coeff((2+nmeds+im),1).
end if.
do if (vy = 1).
compute ymat(6,im) = coeff((3+nmeds+im),1).
end if.
do if (qy = 1 and vy = 1).
compute ymat(6,im) = coeff((5+nmeds+((im-1)*2)),1).
compute ymat(7,im) = coeff((6+nmeds+((im-1)*2)),1).
end if.
do if (vqy = 1).
compute ymat(6,im) = coeff((6+nmeds+((im-1)*3)),1).
compute ymat(7,im) = coeff((7+nmeds+((im-1)*3)),1).
compute ymat(8,im) = coeff((8+nmeds+((im-1)*3)),1).
end if.
do if (wmy = 1).
compute ymat(7,im) = coeff((3+nmeds+im-wy),1).
end if.
do if (wmy = 1 and vy = 1).
compute ymat(7,im) =coeff((4+(nmeds*2)+im-wy),1).
end if.
do if (wmy = 1 and vy = 1 and wvmy = 1).

```

```

compute ymat(7,im) =
coeff((6+(nmeds*2)+((im-1)*2)-wy),1).
compute ymat(8,im) =
coeff((7+(nmeds*2)+((im-1)*2)-wy),1).
end if.
do if (wmy = 1 and zmy = 1).
compute ymat(6,im) = coeff((6-(wzy*3)+(wzm-
1)+(nmeds*2)+((im-1)*2)*wzm)+((im-1)*(1-
wzm))-((zy-wzy)*2),1).
compute ymat(7,im) = coeff((3-
zy+im+nmeds),1).
do if (wzmy = 1).
compute ymat(8,im) = coeff((7-
(wzy*3)+(nmeds*2)+((im-1)*2)),1).
end if.
end if.
do if (nmods > 0 and model <> 5).
loop indlp = 1 to nrow(modvals).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp)).
do if (model > 6).
compute indeff(indlp,1) =
csum(ymat(1:4,im)*vmat(1:4,indlp))*csum(ym
at(5:8,im)*vmat(5:8,indlp)).
end if.
end loop.
compute indboot((bt+(im-
1)*(boot+1)),:)=t(indeff).
do if (imm = 1 or imm = 3).
compute
indbootp(bt,im)=ymat(2,im)*ymat(5,im).
do if (imm=3).
compute
indbootp(bt,(im+nmeds))=ymat(3,im)*ymat(5,i
m).
do if (zdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxz-cminz).
end if.

```

```

end if.
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
end if.
end if.
do if ((imm= 2 or imm = 4) or model = 74).
compute
indbootp(bt,im)=ymat(1,im)*ymat(6,im).
do if (imm=4).
compute
indbootp(bt,(im+nmeds))=ymat(1,im)*ymat(7,i
m).
do if (qdich=1).
compute
indbootp(bt,(im+nmeds))=indbootp(bt,(im+nme
ds))*(cmaxq-cminq).
end if.
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
end if.
end if.
do if (imm=7).
compute
indbootp(bt,im)=ymat(2,im)*ymat(6,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(2,im)*y
mat(6,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(6,im)*onesmv1)+(ymat(2,im)*y
mat(6,im)*t(immmv1)).
do if (wdich=1).

```



```

compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).
end if.
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxv-cminv).
end if.
end if.
do if (imm=5).
compute
indbootp(bt,im)=ymat(4,im)*ymat(5,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(2,im)*ymat(5,im)*onesmv2)+(ymat(4,im)*y
mat(5,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(3,im)*ymat(5,im)*onesmv1)+(ymat(4,im)*y
mat(5,im)*t(immmv1)).
do if (wdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxw-
cminw).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxw-cminw).

```

```

end if.
do if (zdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxz-
cminz).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxz-cminz).
end if.
end if.
do if (imm=6).
compute
indbootp(bt,im)=ymat(1,im)*ymat(8,im).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=(y
mat(1,im)*ymat(6,im)*onesmv2)+(ymat(1,im)*y
mat(8,im)*t(immmv2)).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=(y
mat(1,im)*ymat(7,im)*onesmv1)+(ymat(1,im)*y
mat(8,im)*t(immmv1)).
do if (vdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxv-
cminv).
compute indboop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))=ind
boop1(bt,(((im-
1)*nrow(immmv2))+1):(im*nrow(immmv2)))*(c
maxv-cminv).
end if.
do if (qdich=1).
compute
indbootp(bt,im)=indbootp(bt,im)*(cmaxq-
cminq).
compute indboop2(bt,(((im-
1)*nrow(immmv1))+1):(im*nrow(immmv1)))=ind
boop2(bt,(((im-

```

```

1)*nrow(immmv1))+1):(im*nrow(immmv1)))*(c
maxq-cminq).
end if.
end if.
do if ((model = 58 or model = 59) and
wvdich=1).
compute
indbootp(bt,im)=(ymat(1,im)*ymat(7,im)*(cmax
w-cminw))+(ymat(2,im)*ymat(5,im)*(cmaxw-
cminw))+(ymat(2,im)*ymat(7,im)*((cmaxw*cma
xw)-(cminw*cminw))).
end if.
end if.
do if (model = 4 or model = 5).
compute
indboot(bt,im)=csum(ymat(1:4,im)*vmat(1:4,1
))*csum(ymat(5:8,im)*vmat(5:8,1)).
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.000000000000001.
end if.
compute sumind=rsum(indboot(bt,:)).
do if (im=nmeds).
compute
pmeff(bt,2:(im+1))=indboot(bt,:)/(sumind+coeff(
(2+nmeds),1)).
end if.
compute
rmeff(bt,(im+1))=indboot(bt,im)/coeff((2+nmeds
),1).
compute
abpseff(bt,(im+1))=indboot(bt,im)/stddevy.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*stddevx.
do if (ncovs > 0 or clsdmy > 0).
compute
abpseff(bt,(im+1))=indboot(bt,im)/sdy cov.
compute
abcseff(bt,(im+1))=abpseff(bt,(im+1))*sdx cov.
end if.

```

```

do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,1) = r2my-(r2full-r2xy).
compute abmmr = 1.
do if (indboot(bt,im) < 0).
compute abmmr = cmin(abmm).
end if.
do if (indboot(bt,im) > 0).
compute abmmr = cmax(abmm).
end if.
compute kappa2(bt,1)=indboot(bt,im)/abmmr.
compute tmp = indboot(bt,im)/abmmr.
end if.
end if.
end if.
do if (model = 6).
do if (nmeds = 2).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(4,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3).
compute
indboot(bt,3)=mmpaths(3,1)*mmpaths(4,3).
end if.
do if (nmeds = 3).
compute indboot(bt,1) =
mmpaths(2,1)*mmpaths(5,2).
compute indboot(bt,2) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3).
compute indboot(bt,3) =
mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4).
compute indboot(bt,4) =
mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4).
compute indboot(bt,5) =
mmpaths(3,1)*mmpaths(5,3).
compute indboot(bt,6) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4).

```

```

compute indboot(bt,7) =
mmpaths(4,1)*mmpaths(5,4).
end if.
do if (nmeds = 4).
compute
indboot(bt,1)=mmpaths(2,1)*mmpaths(6,2).
compute
indboot(bt,2)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(6,3).
compute
indboot(bt,3)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(6,4).
compute
indboot(bt,4)=mmpaths(2,1)*mmpaths(5,2)*mmpaths(6,5).
compute
indboot(bt,5)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(6,4).
compute
indboot(bt,6)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(5,3)*mmpaths(6,5).
compute
indboot(bt,7)=mmpaths(2,1)*mmpaths(4,2)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,8)=mmpaths(2,1)*mmpaths(3,2)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute
indboot(bt,9)=mmpaths(3,1)*mmpaths(6,3).
compute
indboot(bt,10)=mmpaths(3,1)*mmpaths(4,3)*mmpaths(6,4).
compute indboot(bt,11)
=mmpaths(3,1)*mmpaths(5,3)*mmpaths(6,5).
compute indboot(bt,12) =
mmpaths(3,1)*mmpaths(4,3)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,13) =
mmpaths(4,1)*mmpaths(6,4).

```

```

compute indboot(bt,14) =
mmpaths(4,1)*mmpaths(5,4)*mmpaths(6,5).
compute indboot(bt,15) =
mmpaths(5,1)*mmpaths(6,5).
end if.
do if (effsize = 1 and dichy = 0).
do if (ctot = 0).
compute ctot=.00000000000001.
end if.
compute
pmeff(bt,:)=indboot(bt,:)/(rsum(indboot(bt,:))+m
mpaths(nrow(mmpaths),1)).
compute rmeff(bt,:) =
indboot(bt,:)/mmpaths(nrow(mmpaths),1).
compute abpseff(bt,:)=indboot(bt,:)/stddevy.
compute
abcseff(bt,:)=(stddevx*indboot(bt,:))/stddevy.
do if (ncovs > 0 or clsdmy > 0).
compute abpseff(bt,:)=indboot(bt,:)/sdy cov.
compute
abcseff(bt,:)=(sdx cov*indboot(bt,:))/sdy cov.
end if.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute r245(bt,:) = r2my-(r2full-r2xy).
end if.
end if.
end if.
end loop.
end if.
end loop.
do if (savboot = 1 and bt > 1).
compute bootstrp((bt-1),:)=bootcoef.
end if.
end loop.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.

```

```

release datayed, dat, datamed, data, y, x,
datatm, dataty, mints2, cons, mnv, tmp.
do if (mc > 0).
compute x1 = sqrt(-
2*ln(uniform(mc,nrow(mcmats))))&*cos((2*3.14
159265358979)*uniform(mc,nrow(mcmats))).
compute x1=x1*chol(mcmats).
loop i=1 to nrow(x1).
compute x1(i,:)=x1(i,:)+t(mccoeff).
end loop.
loop i = 1 to nmeds.
compute
indboot(2:nrow(indboot),i)=x1(:,i)&*x1(:,(i+nme
ds)).
end loop.
end if.
do if (ttt = 0 and model > 3).
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
end if.
do if (model <> 74).
do if (dichy = 0).
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 't', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.
print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 't', 'p', 'LLCI', 'ULCI'/format =
F10.4.
else.
do if (toteff = 1).
print decotot/title = 'Total effect of X on
Y'/clabels = 'Effect', 'SE', 'Z', 'p', 'LLCI',
'ULCI'/format = F10.4.
end if.

```

```

print deco/title = 'Direct effect of X on Y'/clabels
= 'Effect', 'SE', 'Z', 'p', 'LLCI', 'ULCI' /format =
F10.4.
end if.
end if.
end if.
do if (ttt > 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
compute clbs = {modvnm2, 'Effect', 'SE', 't', 'p',
'LLCI', 'ULCI'}.
compute tratio = (directv&/sedir).
compute p = 2*(1-tcdf(abs(tratio), (n-ncol(xy))))).
compute outp = {modvalsd, directv, sedir,
tratio, p}.
do if (dichy = 0).
compute temp1=directv-sqrt(jncrit)*sedir.
compute temp2=directv+sqrt(jncrit)*sedir.
compute outp = {outp, temp1, temp2}.
end if.
do if (dichy = 1).
compute p = 2*(1-cdfnorm(abs(tratio))).
compute temp = directv-abs(xp2)*sedir.
compute outp = {outp, temp}.
compute temp = directv+abs(xp2)*sedir.
compute outp = {outp, temp}.
compute clbs = {modvnm2, 'Effect', 'SE', 'Z', 'p',
'LLCI', 'ULCI'}.
end if.
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
print outp/title = 'Conditional direct effect(s) of
X on Y at values of the moderator(s):'/format =
F10.4 /cnames = clbs.
end if.
do if (nmods > 0 and model <> 5).
do if (model < 4).

```



```

print/title =
*****
*****!

compute zmat(1,1) = 1*(1-yes).
compute cfse = make(nrow(modvals),1,0).
loop #m = 1 to nrow(modvals).
do if (model = 1).
compute zmat(2,1)=modvals(#m,1)**2.
compute zmat(5,1)=2*modvals(#m,1).
end if.
do if (model = 2 or model = 3).
compute zmat(2,1)=modvals(#m,2)**2.
compute zmat(3,1)=modvals(#m,1)**2.
compute
zmat(4,1)=(modvals(#m,1)**2)*(modvals(#m,2)
**2)*(1-yes2).
compute zmat(5,1)=2*modvals(#m,2)*(1-yes).
compute zmat(6,1)=2*modvals(#m,1)*(1-yes).
compute
zmat(7,1)=2*modvals(#m,1)*modvals(#m,2)*(1-
yes).
compute
zmat(8,1)=2*modvals(#m,1)*modvals(#m,2).
compute
zmat(9,1)=2*modvals(#m,1)*(modvals(#m,2)**
2)*(1-yes2).
compute
zmat(10,1)=2*(modvals(#m,1)**2)*modvals(#m
,2)*(1-yes2).
end if.
compute cfse(#m,1)=sqrt(csum(zmat&*cmat)).
end loop.
end if.
do if (nmods > 0).
compute clbs = {modvnm, 'Effect'}.
loop im = 1 to nmeds.
compute obs = t(indboot(1+(im-1)*(boot+1),:)).
compute outp = {modvals, obs}.
do if (model < 4).
compute tstat = obs&/cfse.

```

```

do if (dichy = 0).
compute pval = 2*(1-tcdf(abs(tstat), (n-
ncol(xy)))).
compute temp=obs-sqrt(jncrit)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp=obs+sqrt(jncrit)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 't', 'p', 'LLCI', 'ULCI'}.
compute jnclbs=clbs.
end if.
do if (dichy = 1).
compute pval = 2*(1-cdfnorm(abs(tstat))).
compute temp = obs-abs(xp2)*cfse.
compute outp = {outp, cfse, tstat, pval, temp}.
compute temp = obs+abs(xp2)*cfse.
compute outp = {outp, temp}.
compute clbs = {clbs, 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute jnclbs=clbs.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indboot((1+(im-
1)*(boot+1)),:).
compute indboot2 = indboot((2+(im-
1)*(boot+1)):(1+(im-1)*(boot+1)+boot),:).
compute mnind = t(csum(indboot2)/boot).
compute stdind=t(sqrt((cssq(indboot2)-
((csum(indboot2)**2)/boot))/(boot-1))).
compute llci=make(1,ncol(indboot2),-999).
compute ulci=make(1,ncol(indboot2),-999).
loop #e = 1 to ncol(indboot2).
.
compute temp = indboot2(:,#e).
compute temp(GRADE( indboot2(:,#e) )) =
indboot2(:,#e).
compute badlo = 0.
compute badhi = 0.

```

```

do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.

```

```

do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {modvals, obs, stdind, t(llci),
t(ulci)}.
compute clbs = {modvnm, 'Effect', 'Boot SE',
'BootLLCI', 'BootULCI'}.
end if.
compute mtemp = mnames(1, im).
compute rlbs =
make(nrow(modvals),1,mnames(1,im)).
do if (model < 4).
do if (coeffci = 0).
compute outp = outp(:,1:(ncol(outp)-2)).
end if.
do if (yes=0).
print outp/title = 'Conditional effect of X on Y at
values of the moderator(s):'/cnames =
clbs/format = F10.4.
else.
compute outp=outp(:,3:ncol(outp)).
compute clbs=clbs(:,3:ncol(clbs)).
compute clbs(1,1)='Contrast'.
print outp/title = 'Contrast of conditional effects
of X on Y'/cnames = clbs/format = F10.4.
end if.
end if.
do if (model > 5 and mod74dic <> 1).
do if (im = 1).
print/title = 'Conditional indirect effect(s) of X on
Y at values of the moderator(s):'.
end if.
print outp/title = 'Mediator'/rnames =
rlbs/cnames = clbs/format = F10.4.

```

```

else if (mod74dic = 1 and model = 74).
do if (im = 1).
print/title = 'Indirect effect(s) of X on Y:'.
compute clbs3=clbs(1,2:ncol(clbs)).
end if.
compute outp3=outp(1,2:ncol(outp)).
print outp3/title = 'Mediator'/rnames =
rlbs/cnames = clbs3/format = F10.4.
end if.
end loop.
loop i = notes to 1 by -1.
do if (note(i,1) = 4 and yes=0).
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 5 and yes=0).
print/title = 'Values for quantitative moderators
are the mean and plus/minus one SD from
mean.'.
print/title = 'Values for dichotomous moderators
are the two values of the moderator.'/space=0.
end if.
do if (note(i,1) = 14 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    below the mean was replaced
with the minimum because one SD below the
mean'/space=0.
print/title = '    is outside of the range of the
data.'/space=0.
end if.
do if (note(i,1) = 15 and yes=0).
print/title = 'NOTE: For at least one moderator
in the conditional effects table above, one SD '.
print/title = '    above the mean was replaced
with the maximum because one SD above the
mean'/space=0.

```

```

print/title = '      is outside of the range of the
data.'/space=0.
end if.
end loop.
do if (model = 3 and yes=0).
compute jnvals=make(nrow(matw),7,0).
compute jnvals(:,1)=matw.
compute jnvals(:,2)=jnb1+jnb3*jnvals(:,1).
compute
jnvals(:,3)=sqrt(jnsb1+2*jnvals(:,1)*jnsb1b3+(jn
vals(:,1)&*jnvals(:,1))*jnsb3).
compute jnvals(:,4)=jnvals(:,2)&/jnvals(:,3).
do if (dichy = 0).
compute jnvals(:,5)=2*(1-tcdf(abs(jnvals(:,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(:,5)=2*(1-
cdfnorm(abs(jnvals(:,4)))).
end if.
compute jnvals(:,6)=jnvals(:,2)-
sqrt(jncrit)&*jnvals(:,3).
compute
jnvals(:,7)=jnvals(:,2)+sqrt(jncrit)&*jnvals(:,3).
compute clbs={clbs(:,1),clbs(:,3:ncol(clbs))}.
do if (coeffci = 0).
compute jnvals =jnvals(:,1:(ncol(jnvals)-2)).
end if.
print jnvals/title = 'Conditional effect of X*M
interaction at values of W:'/cnames
=clbs/format = F10.4.
end if.
do if (jn = 1 and (model = 1 or model = 3) and
jndich = 0 and yes=0).
print/title = '***** JOHNSON-
NEYMAN TECHNIQUE
*****'.
compute ajn =(jncrit*jnsb3)-(jnb3*jnb3).
compute bjn = 2*((jncrit*jnsb1b3)-(jnb1*jnb3)).
compute cjn = (jncrit*jnsb1)-(jnb1*jnb1).

```

```

compute radarg = (bjn*bjn)-(4*ajn*cjn).
compute den = 2*ajn.
compute nrts = 0.
do if (radarg >= 0 and den <> 0).
compute x21 = (-bjn+sqrt(radarg))/den.
compute x22 = (-bjn-sqrt(radarg))/den.
compute roots = 0.
do if (x21 >= jnmin and x21 <= jnmax).
compute nrts = 1.
compute roots = {roots; x21}.
end if.
do if (x22 >= jnmin and x22 <= jnmax).
compute nrts = nrts + 1.
compute roots = {roots; x22}.
end if.
compute roots={roots,make(nrow(roots),2,0)}.
compute modtemp=m.
do if (model=3).
compute modtemp=w.
end if.
do if (nrts > 0).
compute roots = roots(2:nrow(roots),1:3).
compute roots(1,2)=(csum(modtemp <
roots(1,1))/n)*100.
compute roots(1,3)=(csum(modtemp >
roots(1,1))/n)*100.
do if (nrow(roots)=2).
compute roots(2,2)=(csum(modtemp <
roots(2,1))/n)*100.
compute roots(2,3)=(csum(modtemp >
roots(2,1))/n)*100.
end if.
print roots/title = 'Moderator value(s) defining
Johnson-Neyman significance
region(s)'/clabels = 'Value', '% below', '%
above'/format = F10.4.
compute jnvals=make((21+nrts),7,0).
loop i= 0 to 20.
compute jnvals((i+1),1)=jnmin+(i*((jnmax-
jnmin)/20)).

```

```

end loop.
loop i = 1 to nrts.
loop j = 2 to nrow(jnvals).
do if ((roots(i,1) > jnvals((j-1),1)) and (roots(i,1)
< jnvals(j,1))).
compute
jnvals((j+1):(21+i),1)=jnvals(j:(20+i),1).
compute jnvals(j,1)=roots(i,1).
end if.
end loop.
end loop.
loop i = 1 to nrow(jnvals).
compute jnvals(i,2)=jnb1+jnb3*jnvals(i,1).
compute
jnvals(i,3)=sqrt(jnsb1+2*jnvals(i,1)*jnsb1b3+(jn
vals(i,1)*jnvals(i,1))*jnsb3).
compute jnvals(i,4)=jnvals(i,2)/jnvals(i,3).
do if (dichy = 0).
compute jnvals(i,5)=2*(1-tcdf(abs(jnvals(i,4)),
jndf)).
end if.
do if (dichy = 1).
compute jnvals(i,5)=2*(1-
cdfnorm(abs(jnvals(i,4)))).
end if.
compute jnvals(i,6)=jnvals(i,2)-
sqrt(jncrit)*jnvals(i,3).
compute
jnvals(i,7)=jnvals(i,2)+sqrt(jncrit)*jnvals(i,3).
end loop.
do if (model = 1).
print jnvals/title = 'Conditional effect of X on Y
at values of the moderator (M)'/cnames
=jnclbs/format = F10.4.
end if.
do if (model = 3).
compute
jnclbs={jnclbs(:,1),jnclbs(:,3:ncol(jnclbs))}.

```



```

print jnvals/title = 'Conditional effect of X*M on
Y at values of the moderator (W)'/cnames
=jnclbs/format = F10.4.
end if.
end if.
do if (nrts = 0).
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
else.
print/title = 'There are no statistical significance
transition points within the observed'.
print/title = 'range of the moderator.'/space=0.
end if.
end if.
end if.
do if (model < 4 and (plot = 1)).
compute dataplot =
make((nrow(modvals)*nrow(matx)),(ncol(modv
als)+1),0).
compute tmp = 1.
loop i = 1 to nrow(modvals).
loop j = 1 to nrow(matx).
compute dataplot(tmp,:)= {matx(j,1),
modvals(i,:)}.
compute tmp=tmp+1.
end loop.
end loop.
compute dataplot = {dataplot,
make(nrow(dataplot),(1+dichy),0)}.
compute dataplo2 = make(nrow(dataplot),1,1).
do if (model = 1).
compute dataplo2 = {dataplo2, dataplot(:,2),
dataplot(:,1), (dataplot(:,1)*dataplot(:,2))}.
end if.
do if (model = 2 or model = 3).
compute dataplo2 = {dataplo2, dataplot(:,3),
dataplot(:,1), (dataplot(:,1)*dataplot(:,3)),
dataplot(:,2), (dataplot(:,1)*dataplot(:,2))}.

```

```

do if (model = 3).
compute dataplo2 = {dataplo2,
(dataplot(:,2)*dataplot(:,3)),
(dataplot(:,1)*dataplot(:,2)*dataplot(:,3))}.
end if.
end if.
loop i = 1 to nrow(dataplot).
compute tmp=dataplo2(i,:).
do if (ncovs > 0).
compute tmp = {tmp, covmeans}.
end if.
do if (cluster > 0).
compute tmp = {tmp, cldmeans}.
end if.
compute dataplot(i,(ncol(dataplot)-
(dichy)))=tmp*coeffplt.
do if (dichy = 1).
compute
dataplot(i,(ncol(dataplot)))=exp(tmp*coeffplt)/(1
+exp(tmp*coeffplt)).
end if.
end loop.
compute clbs = {xname, modvnm, 'yhat'}.
do if (dichy = 1).
compute clbs = {xname, modvnm, 'ln(odds)',
'prob'}.
end if.
print/title =
*****
*****!

```

```

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (model = 1).
do if (dichy = 0).
print/title = 'DATA LIST FREE/Comm_Pract
Technological_Advan Financial_Performance.'
/space=1.
else if (dichy = 1).
print/title = 'DATA LIST FREE/Comm_Pract
Technological_Advan logodds prob.'
/space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
Financial_Performance BY
Technological_Advan.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH Financial_Performance BY
Comm_Pract.' /space=0.
end if.
end if.

```

```

do if (dichy = 1).
do if (xdich = 0).
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
logodds BY Technological_Advan.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
prob BY Technological_Advan.' /space=0.
end if.
do if (xdich <> 0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Comm_Pract.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Comm_Pract.' /space=0.
end if.
end if.
end if.
do if (model = 2 or model = 3).
do if (dichy = 0).
print/title = 'DATA LIST FREE/Comm_Pract
Technological_Advan Financial_Performance.'
/space=1.
end if.
do if (dichy = 1).
print/title = 'DATA LIST FREE/Comm_Pract
Technological_Advan logodds prob.' /space=1.
end if.
print/title = 'BEGIN DATA.' /space=0.
print dataplot/title=' /format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (dichy = 0).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
Financial_Performance BY
Technological_Advan/PANEL ROWVAR='
/space=0.
end if.

```

```

do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH Financial_Performance BY
Comm_Pract/PANEL ROWVAR=' /space=0.
end if.
end if.
do if (dichy = 1).
do if (xdich=0).
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
logodds BY Technological_Advan/PANEL
ROWVAR=' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
prob BY Technological_Advan/PANEL
ROWVAR=' /space=0.
end if.
do if (xdich<>0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Comm_Pract/PANEL
ROWVAR=' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Comm_Pract/PANEL
ROWVAR=' /space=0.
end if.
end if.
end if.
do if (ncovs > 0).
print/title = '* Estimates are based on setting
covariates to their sample means.'.
end if.
end if.
end if.
do if ((imm > 0) or (model = 74 and
mod74dic=0) or ((model = 58 or model = 59)
and wvdich=1))).
do if (imm > 4 and imm < 8).

```

```

compute
indbootp={indbootp(:,1:nmeds),indboop1,indbo
op2}.
end if.
compute obsprod = t(indbootp(1,:)).
do if (boot > 0).
compute ones = make(boot,1,1).
compute estmte=indbootp(1,:).
compute indbootp = indbootp(2:(boot+1),:).
compute mnindp = t(csum(indbootp)/boot).
compute stdindp=t(sqrt((cssq(indbootp)-
((csum(indbootp)&*2)/boot))/(boot-1))).
compute llcip=make(1,ncol(indbootp),-999).
compute ulcip=make(1,ncol(indbootp),-999).
loop #e = 1 to ncol(indbootp).

.
compute temp = indbootp(:,#e).
compute temp(GRADE( indbootp(:,#e) )) =
indbootp(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).

```

```

compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcip(1,#e)=llcit.
compute ulcip(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute outp = {obsprod, stdindp, t(llcip),
t(ulcip)}.
compute clbs = {'Effect', 'SE(Boot)', 'BootLLCI',
'BootULCI'}.
do if (model = 8 or model = 12).
print/title='-----'.
compute outpimm=outp(1:nmeds,:).

```

```

print/title = 'Indirect effect of highest order
product:.'/space=0.
print outpimm/title = 'Mediator'/cnames =
clbs/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
compute clbs3 = {'Index', 'SE(Boot)',
'BootLLCI', 'BootULCI'}.
do if (imm < 5).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=outp((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.
print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
do if (imm > 4 and imm < 8).

```



```

print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'.

compute outpimm=outp(1:nmeds,:).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/cnames =
clbs3/rnames = mnames/format = F10.4.
compute
outpimm=outp((nmeds+1):nrow(outp),:).
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.

loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),clbs3}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3(1,1)=modvnm(1,1).
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
do if (wvdich=1 or zqdich=1).
print/title = 'When the moderator is
dichotomous, this is a test of equality of the'.

```

```

print/title = 'conditional indirect effects in the
two groups.'/space=0.
end if.
end if.
end if.
do if (boot = 0).
do if (model = 8 or model = 12).
print/title='-----'.
compute obsimm=obsprod(1:nmeds,:).
print/title = 'Indirect effect of highest order
product.'/space=0.
print obsimm/title = 'Mediator'/clabels =
'Effect'/rnames = mnames/format = F10.4.
end if.
do if ((imm = 1 or imm = 2) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
print/title = '***** INDEX OF
MODERATED MEDIATION
*****'.
end if.
do if (imm=3 or imm=4).
print/title = '***** INDEX OF
PARTIAL MODERATED MEDIATION
*****'.
end if.
do if ((imm > 0 and imm < 5) or ((model = 58 or
model = 59) and wvdich = 1) or (model = 74
and mod74dic=0)).
loop k4= 1 to nmods.
do if (nmods > 1).
print {modvnm(1,k4)}/title =
'Moderator:.'/format=A8.
end if.
compute outpimm=obsprod((((k4-
1)*nmeds)+1):(nmeds*k4),:).
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
end loop.
end if.

```

```

do if (imm > 4 and imm < 8).
print/title = '***** MODERATED AND
CONDITIONAL MODERATED MEDIATION
*****'
compute outpimm=obsprod(1:nmeds,1).
print/title='INDEX OF MODERATED
MODERATED MEDIATION:'.
print outpimm/title = 'Mediator'/clabels =
'Index'/rnames = mnames/format = F10.4.
compute
outpimm=obsprod((nmeds+1):nrow(obsprod),:)
.
compute outpimm={cmmval,outpimm}.
print/title='INDEX OF CONDITIONAL
MODERATED MEDIATION:'.
loop k4= 1 to 2.
do if (k4 = 1).
compute
outpimm1=outpimm(1:(nmeds*nrow(immmv2)),
:).
compute clbs3={modvnm(1,2),'Index'}.
compute
cmmlbs=cmmlbs1(2:nrow(cmmlbs1),1).
end if.
do if (k4 = 2).
compute
outpimm1=outpimm(((nmeds*nrow(immmv2))+
1):nrow(outpimm),:).
compute clbs3={modvnm(1,1),'Index'}.
compute
cmmlbs=cmmlbs2(2:nrow(cmmlbs2),1).
end if.
print {modvnm(1,k4)}/title =
'Moderator: '/format=A8.
print outpimm1/title =
'Mediator'/cnames=clbs3/rnames=cmmlbs/form
at= F10.4.
end loop.
end if.
end if.

```

```

end if.
compute conmake=0.
compute concols = 0.
do if ((model > 3 and model < 7) and (contrast
> 0) and nmods = 0 and nmeds > 1 ).
compute concols =
((ncol(indboot)*(ncol(indboot)-1))/2).
compute indcon=make(nrow(indboot),concols,-
999).
compute conkey = { ' ', ' ', ' ', ' ' }.
compute temp=1.
compute conmake=1.
loop i = 1 to (ncol(indboot)-1).
loop j = (i+1) to (ncol(indboot)).
compute indcon(:,temp)=indboot(:,i)-
indboot(:,j).
do if (contrast = 2).
compute indcon(:,temp)=abs(indboot(:,i))-
abs(indboot(:,j)).
end if.
do if (model <= 6).
compute conkey={conkey; mnames(1,i),
'minus', mnames(1,j)}.
end if.
do if (model = 6).
compute conkey={conkey; indlbl2(i,1), 'minus',
indlbl2(j,1)}.
end if.
compute temp=temp+1.
end loop.
end loop.
end if.
do if (model = 4 or model = 5).
compute clbs = {'Effect'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obs = {csum(obs); obs}.
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.

```

```

compute rlbs = {rlbs;
cntname(1:ncol(indcon),1)}.
end if.
compute outp = obs.
compute outp2=outp.
do if (effsize = 1 and dichy = 0).
compute
pmeff(:,1)=rsum(pmeff(:,2:ncol(pmeff))).
compute
rmeff(:,1)=rsum(rmeff(:,2:ncol(rmeff))).
compute
abpseff(:,1)=rsum(abpseff(:,2:ncol(abpseff))).
compute
abcseff(:,1)=rsum(abcseff(:,2:ncol(abcseff))).
compute eff = {pmeff, rmeff, abpseff, abcseff}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute eff = {eff, r245}.
compute r245obs = {r245(1,1);r245(1,1)}.
compute kappa2ob = kappa2(1,1).
end if.
compute pmobs = t(pmeff(1,1:(nmeds+1))).
compute rmobs = t(rmeff(1,1:(nmeds+1))).
compute psobs = t(abpseff(1,1:(nmeds+1))).
compute csobs = t(abcseff(1,1:(nmeds+1))).
do if (contrast = 0).
compute outp2 = {obs, psobs, csobs, pmobs,
rmobs}.
end if.
do if (contrast > 0).
compute obs2=obs(1:nrow(psobs),:).
compute outp2 = {obs2, psobs, csobs, pmobs,
rmobs}.
end if.
compute clbs = {'ab', 'ab_ps', 'ab_cs', 'ab/c',
'ab/c'}.
do if (nmeds = 1 and ncovs = 0 and cluster = 0
and model = 4).
compute outp2 = {outp2, r245obs}.
compute clbs = {clbs, 'R-sq_med'}.

```

```

do if (boot = 0 and mc = 0).
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
do if (boot = 0 and mc = 0).
do if (nmeds = 1).
compute outp2 = outp2(2,:).
compute rlbs = rlbs(2,1).
end if.
do if (model=5).
compute outp2=outp2(:,1:3).
end if.
print outp2/title = 'Indirect effect(s) of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1 and nmeds >
1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0 or mc > 0).
compute indboot = {rsum(indboot), indboot}.
compute bootSZ=boot.
do if (mc > 0).
compute bootSZ=mc.
end if.
compute ones = make(bootSZ,1,1).
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(bootSZ+1),:).
compute mnind = t(csum(indboot)/bootSZ).

```

```

compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/bootSZ))/(bootSZ-1))).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).
loop #e = 1 to ncol(indboot).

.

compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.

```

```

end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llici(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
do if (effsize = 1 and dichy = 0 and boot > 0).
compute estmte=eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).
.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.

```



```

compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).

```

```

compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
end if.
end if.
do if (boot > 0 or mc > 0).
compute outp = {obs, stdind, t(llci), t(ulci)}.
do if (nmeds = 1).
compute outp = outp(2,:).
compute rlbs = rlbs(2,1).
end if.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
do if (mc > 0).
compute clbs = {'Effect', 'MC SE', 'MC LLCI',
'MC ULCI'}.
end if.
print outp/title = 'Indirect effect of X on
Y'/rnames = rlbs/cnames = clbs/format =
F10.4.
do if (dichy = 0 and effsize = 1 and boot > 0).
compute outp = {psobs,
stdindf((2*(nmeds+1)+1):(3*(nmeds+1)),1),
t(llcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))),
t(ulcif(1,(2*(nmeds+1)+1):(3*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Partially standardized indirect
effect of X on Y'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute outp = {csobs,
stdindf((3*(nmeds+1)+1):(4*(nmeds+1)),1),
t(llcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))),
t(ulcif(1,(3*(nmeds+1)+1):(4*(nmeds+1))))).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.

```

```

print outp/title = 'Completely standardized
indirect effect of X on Y'/rnames = rlbs/cnames
= clbs/format = F10.4.
compute outp = {pmobs,
stdindf(1:(nmeds+1),1),t(llcif(1,1:(nmeds+1))),
t(ulcif(1,1:(nmeds+1))) }.
do if (model = 4).
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
compute outp = {rmobs, stdindf(((nmeds+1)+1)
:(2*(nmeds+1)) ,1),
t(llcif(1,((nmeds+1)+1):(2*(nmeds+1)))),
t(ulcif(1,((nmeds+1)+1):(2*(nmeds+1))))}.
do if (nmeds = 1).
compute outp = outp(2,:).
end if.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = rlbs/cnames = clbs/format
= F10.4.
do if (nmeds = 1 and cluster = 0 and ncovs = 0
and model = 4).
compute r245obs = r245obs(1,1).
compute outp = {r245obs,
stdindf((4*(nmeds+1)+1):(4*(nmeds+1)+1),1),
t(llcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1))),
t(ulcif(1,(4*(nmeds+1)+1):(4*(nmeds+1)+1)))}.
print outp/title = 'R-squared mediation effect
size (R-sq_med)'/rnames = rlbs/cnames =
clbs/format = F10.4.
compute note(notes,1) = 32.
compute notes = notes + 1.
end if.
end if.
end if.
end if.
do if (normal = 1).

```

```

do if (nmeds = 1).
print sobel/title = 'Normal theory tests for
indirect effect'/clabels = 'Effect', 'se', 'Z',
'p'/format = F10.4.
end if.
do if (nmeds > 1).
compute rlbs2 = rlbs(2:nrow(rlbs),1).
print sobel/title = 'Normal theory tests for
specific indirect effects'/rnames = rlbs2/clabels
= 'Effect', 'se', 'Z', 'p'/format = F10.4.
end if.
end if.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
do if (model = 6).
compute clbs = {'eff'}.
compute rlbs = {'TOTAL'; t(mnames)}.
compute obs = t(indboot(1,:)).
compute obsfsum=csum(obs).
compute obs = {obsfsum; obs}.
compute indlbl = indlbl(1:nrow(obs),1).
do if (conmake = 1).
compute obs={obs; t(indcon(1,:))}.
compute indlbl = {indlbl;
cntname(1:ncol(indcon),1)}.
end if.
compute obs2=obs.
do if (boot = 0).
do if (dichy = 0 and effsize = 1).
compute clbs = {'eff', 'eff_ps', 'eff_cs', 'eff/c',
'eff/c'}.
do if (ncovs = 0).

```

```

compute obs = {obs, obs/stddevy,
(obs*stddevx/stddevy), obs/ctot,
obs/mmpaths(nrow(mmpaths),1)}.
end if.
do if (ncovs > 0 or clsdmy > 0).
compute obs = {obs, obs/sdycov,
(obs*sdxcov/sdycov),
obs/(obsfsum+mmpaths(nrow(mmpaths),1)),
obs/mmpaths(nrow(mmpaths),1)}.
end if.
compute obs2=obs.
do if (contrast > 0).
compute obs2 = obs(1:(nrow(obs)-concols),:).
end if.
end if.
print obs2/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (contrast > 0 and effsize = 1).
compute outp2=t(indcon(1,:)).
compute rlbs2 = cntname(1:ncol(indcon),1).
print outp2/title = 'Contrast(s) between indirect
effects'/rnames = rlbs2/cnames=clbs/format =
F10.4.
end if.
end if.
do if (boot > 0).
compute ones = make(boot,1,1).
compute indboot = {rsum(indboot), indboot}.
do if (conmake = 1).
compute indboot={indboot, indcon}.
end if.
compute estmte=indboot(1,:).
compute indboot = indboot(2:(boot+1),:).
compute mnind = t(csum(indboot)/boot).
compute stdind=t(sqrt((cssq(indboot)-
((csum(indboot)**2)/boot))/(boot-1))).
compute temp = nrow(indboot).
compute llci=make(1,ncol(indboot),-999).
compute ulci=make(1,ncol(indboot),-999).

```

```

loop #e = 1 to ncol(indboot).

.

compute temp = indboot(:,#e).
compute temp(GRADE( indboot(:,#e) )) =
indboot(:,#e).
compute badlo = 0.
compute badhi = 0.
do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(estmte(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.

```

```

do if ( (estmte(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,#e)=llcit.
compute ulci(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute obs = {obs, stdind, t(llci), t(ulci)}.
compute clbs = {'Effect', 'Boot SE', 'BootLLCI',
'BootULCI'}.
print obs/title = 'Indirect effect(s) of X on
Y'/rnames = indlbl/cnames = clbs/format =
F10.4.
do if (effsize = 1 and dichy = 0).
compute indboot=indboot(:,1:(ncol(indboot)-
concols)).
compute eff = {rsum(abpseff), abpseff,
rsum(abcseff), abcseff, rsum(pmeff), pmeff,
rsum(rmeff), rmeff}.
compute effobs = eff(1,:).
compute eff = eff(2:nrow(eff),:).
compute stdindf=t(sqrt((cssq(eff)-
((csum(eff)**2)/boot))/(boot-1))).
compute llcif=make(1,ncol(eff),-999).
compute ulcif=make(1,ncol(eff),-999).
loop #e = 1 to ncol(eff).
.
compute temp = eff(:,#e).
compute temp(GRADE( eff(:,#e) )) = eff(:,#e).

```

```

compute badlo = 0.
compute badhi = 0.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(effobs(1,#e)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.
do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (effobs(1,#e)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.

```



```

compute llcif(1,#e)=llcit.
compute ulcif(1,#e)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute temp2 = stdindf(1:ncol(indboot),1).
compute temp3 = effobs(:,1:ncol(indboot)).
compute templo = llcif(1,1:ncol(indboot)).
compute temphi = ulcif(1,1:ncol(indboot)).
compute outp = {t(temp3), temp2,t(templo),
t(temphi)}.
print outp/title = 'Partially standardized indirect
effect of X on Y'/cnames = clbs/rnames =
indlbl/format = F10.4.
compute temp2 =
stdindf((ncol(indboot)+1):(2*ncol(indboot)),1).
compute temp3 =
effobs(:,(ncol(indboot)+1):(2*ncol(indboot))).
compute templo =
llcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute temphi =
ulcif(1,(ncol(indboot)+1):(2*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templo),
t(temphi)}.
print outp/title = 'Completely standardized
indirect effect of X on Y'/cnames = clbs/rnames
= indlbl/format = F10.4.
compute temp2 =
stdindf((2*(ncol(indboot))+1):(3*ncol(indboot)),1
).
compute temp3 =
effobs(:,(2*(ncol(indboot))+1):(3*ncol(indboot)))
.

```

```

compute templow =
llcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute temphi =
ulcif(1,(2*(ncol(indboot))+1):(3*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Ratio of indirect to total effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
compute temp =
eff(:,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temp2 =
stdindf((3*(ncol(indboot))+1):(4*ncol(indboot)),1
).
compute temp3 =
effobs(:,(3*(ncol(indboot))+1):(4*ncol(indboot)))
.
compute templow =
llcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute temphi =
ulcif(1,(3*(ncol(indboot))+1):(4*ncol(indboot))).
compute outp = {t(temp3), temp2,t(templow),
t(temphi)}.
print outp/title = 'Ratio of indirect to direct effect
of X on Y'/rnames = indlbl/cnames =
clbs/format = F10.4.
end if.
end if.
do if (nmeds = 2).
compute effkey = {xname, '->', mnames(1,1), '-
>', yname, '', ''}.
compute effkey = {effkey; xname, '->',
mnames(1,1), '->', mnames(1,2), '->', yname}.
compute effkey = {effkey; xname, '->',
mnames(1,2), '->', yname, '', ''}.
compute effkey = {indlbl(2:4,1), effkey}.
end if.
do if (nmeds = 3).
compute effkey = {xname, '->', mnames(1,1), '-
>', yname, '', '', ''}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, '',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, '',' ',' ',' '}.
compute effkey = {indbl(2:8,1), effkey}.
end if.
do if (nmeds = 4).
compute effkey = {xname, '->' , mnames(1,1), '-
>', yname, '',' ',' ',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,4), '->', yname, '
',' ',' ',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', yname, '',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,4), '->', yname, '',' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, '',' '}.

```

```

compute effkey = {effkey; xname, '->' ,
mnames(1,1), '->', mnames(1,2), '->',
mnames(1,3), '->', mnames(1,4), '->', yname}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,4), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,2), '->', mnames(1,3), '->',
mnames(1,4), '->', yname, ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,3), '->', mnames(1,4), '->', yname, '
', ',', ',', ' '}.
compute effkey = {effkey; xname, '->' ,
mnames(1,4), '->', yname, ',', ',', ',', ',', ',', ' '}.
compute effkey = {indlbl(2:16,1), effkey}.
end if.
print effkey/title = 'Indirect effect key'/format =
A8.
do if (conmake = 1).
compute conkey=conkey(2:nrow(conkey),:).
compute conlbs = cntname(1:ncol(indcon),1).
print conkey/title = 'Specific indirect effect
contrast definitions'/rnames = conlbs/format =
A10.
end if.
end if.
end if.
DO IF (medte2=1).
compute
name=t(mdlnms2(2:nrow(mdlnms2),1)).
compute tmpdat=datayed(:,2:ncol(datayed)).
compute dd=tmpdat(:,1).

```

```

compute
dd={dd,tmpdat(:,(2+nmeds)),tmpdat(:,2:(1+nmeds))}.
compute stratify=0.
do if (ncovs > 0).
compute cname = cnames.
compute
dd={dd,tmpdat(:,(3+nmeds):(2+nmeds+ncovs))}.
end if.
compute method = 1.
compute samples = boot.
do if (mc > 0).
compute samples = mc.
compute method=2.
end if.
compute omnibus = 1.
compute booterr = 0.
do if (mcx < 0 or mcx > 4).
compute mcx = 1.
end if.
compute ovals = ncol(design(dd(:,1))).
compute nte = notes.
compute criterr = 0.
compute xskip = 0.
compute dumok = 0.
compute badend = 0.
compute alpha2 = (1-(conf/100))/2.
compute y5=sqrt(-2*ln(alpha2)).
compute xp2=-
(y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/((((y5
*q4+q3)*y5+q2)*y5+q1)*y5+q0)).
compute priorlo = -9999999.
compute priorhi = 9999999.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute reps = samples.
do if (contrast = 1).
compute note(nte,1) = 26.
compute nte=nte+1.

```

```

end if.
do if (effsize = 1).
compute note(nte,1) = 27.
compute nte=nte+1.
end if.
do if (normal = 1).
compute note(nte,1) = 28.
compute nte=nte+1.
end if.
do if (method = 2).
do if (omnibus = 1).
compute note(nte,1) = 25.
compute nte=nte+1.
end if.
compute samples = 0.
do if (reps < 1000).
compute reps = 1000.
end if.
end if.
do if (method = 1).
compute samples = reps.
end if.
compute temp = ncol(dd).
compute n = nrow(dd).
compute ones = make(n,1,1).
compute temp=dd(:,2).
compute nxd = ncol(temp).
compute nx = nxd.
do if (nx = 1 and mcx > 0).
compute temp = dd.
compute temp(GRADE(dd(:,2)),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,2)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute mnvls = cmin(t(nnvls)).
do if (mnvls < 2).
compute errs = errs + 1.
compute runerrs(errs,1) = 37.
compute criterr = 1.

```

```

end if.
do if (nvls > 9).
compute errs = errs+1.
compute runerrs(errs,1) = 36.
compute criterr = 1.
end if.
do if (criterr = 0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,2).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,2) <> nnvls((temp-1),1)).
compute nnvls(temp,1)=dd(i,2).
compute temp = temp+1.
end if.
end loop.
do if (mcx > 0).
compute x = dummy(:,2:ncol(dummy)).
compute nx = ncol(x).
compute minus1 = make(1,ncol(x),-1).
compute xdes=make((nx+1),3,0).
compute xdes(1,1)=dd(1,2).
compute xdes(1,2)=1.
compute temp = 2.
loop k = 2 to n.
do if (dd(k,2) <> dd((k-1),2)).
compute xdes(temp,2) = k.
compute xdes(temp,1) = dd(k,2).
compute xdes((temp-1),3) = k-1.
compute temp=temp+1.
end if.
end loop.
compute xdes((temp-1),3)=n.
compute xdes = {xdes, (xdes(:,3)-xdes(:,2)+1)}.
do if (stratify <> 0).
compute note(nte,1) = 7.
compute nte=nte+1.
end if.
do if (mcx = 4).

```

```

loop k = 1 to n.
do if (rsum(x(k,:)) = 0).
compute x(k,:) = minus1.
end if.
end loop.
end if.
do if (mcx = 2 or mcx = 3).
compute note(nte,1)=4.
compute nte = nte+1.
loop k = 1 to n.
do if (rsum(x(k,:)) > 0).
loop i = 1 to ncol(x).
do if (x(k,i) = 0).
compute x(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcx = 3).
compute conmat1={-8,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1; 0,0,0,0,0,-
3,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.
loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,:)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute x(k,:)=conmat1((rsum(x(k,:))+1),:).
end loop.
end if.
end if.
compute xdata = x.
compute xskip = 1.
compute xname = ddd(1,1:nx).
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).

```



```

compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcx = 4).
compute dummat(1,:) = minus1.
end if.
do if (mcx = 2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcx = 3).
compute dummat=conmat1.
end if.
compute dummat={nnvls, dummat}.
compute xname2={name(1,2), xname}.
end if.
end if.
compute nm=nmeds.
compute r2b = make(nm,1,-999).
compute indeff = make((nx*nm),1,-999).
compute cov =
make((nx*nm+nm),(nx*nm+nm),0).
compute mns=make(nrow(cov),1,0).
compute ny = ncol(dd)-nm-nxd-ncovs.
compute y = dd(:,1).
compute ydata = y.
compute yname = name(1,1).
do if (xskip = 0).
compute x = dd(:,2:(1+nxd)).
compute xdata = x.
compute xname = name(1,2:(1+nxd)).
compute indlbs = t(xname).
end if.
compute m = dd(:,(2+nxd):(1+nxd+nm)).
compute mdata = m.

```

```

compute mlab = { ' M1 ='; ' M2 ='; ' M3 ='; '
M4 ='; ' M5 ='; ' M6 ='; ' M7 ='; ' M8 ='; '
M9 ='; ' M10 = '}.
compute mlab = {mlab; ' M11 ='; ' M12 ='; '
M13 ='; ' M14 ='; ' M15 = '}.
compute mname =
name(1,(2+nxd):(1+nxd+nm)).
do if (ncovs > 0).
compute c =
dd(:,(2+nxd+nm):(1+nxd+nm+ncovs)).
compute x = {x, c}.
compute xdata = x.
compute cdata = c.
compute xname = {xname, cname}.
end if.
compute nms={yname; t(mname);
t(xname(1,1:(ncol(xname)-ncovs)))}.
do if (dumok = 1).
compute nms={yname; t(mname); name(1,2)}.
end if.
do if (nm < 16).
do if (dumok = 1).
print dummat/title = 'Coding of categorical X
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
compute
boots=make((samples+1),((nx+1)*nm),-999).
do if (errs = 0 and criterr = 0).
loop k = 1 to (samples+1).
do if (k = 2 and savboot = 1).
compute bootstrp =
make(boot,(ncol(bootcoef)), -999).
end if.
compute bootcoef=0.
do if (k > 1).
compute u=trunc(uniform(n,1)*n)+1.
compute xdata = x(u,:).
compute mdata = m(u,:).

```

```

compute ydata = y(u,:).
do if (ncovs > 0).
compute cdata = c(u,:).
end if.
end if.
loop i = 1 to nm.
compute start = 1+(i-1)*nx.
compute xmat = {ones, xdata}.
do if (covmy=2).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = mdata(:,i).
compute amat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*amat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-(((1-r2)*(n-1))/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute r2cha = r2.
compute adjr2cha = adjr2.
do if (ncovs > 0 and covmy <> 2).
compute xmat = {ones, cdata}.
compute atmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*atmat)**2).
compute r2step1 = (sstotal-ssresid2)/sstotal.
compute r2cha = r2-r2step1.
compute adjr2cha = adjr2- (1-(((1-r2step1)*(n-
1))/(n-ncol(xmat)))).
compute xmat = {ones, xdata}.
end if.
compute bootcoef={bootcoef,t(amat)}.
compute r2b(i,1)=adjr2cha.
do if (k = 1).

```

```

compute hcinvtX=inv(t( xmat )' xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)'*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
))&/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat (:,i3) = sqrt( mse )&* xmat (:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvtX*t( xmat )' xmat
*hcinvtX
.
compute tratio = amat&/sqrt(diag(covmat)).
compute rnms = mname(1,i).
print rnms/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.
compute lmat = ident(nrow(amat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*amat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*amat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/'format = F10.4
/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2', 'p'.

```

```

compute op = {amat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*(xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute cov(start:(start+(nx-
1)),start:(start+(nx-
1)))=covmat(2:(ncol(covmat)-(ncovs*(covmy <>
2))),2:(ncol(covmat)-(ncovs*(covmy <> 2)))).
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns(start:(start+(nx-
1)),1)=amat(2:(1+nx)).
end loop.
compute xmat = {ones,mdata,xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (k = 1).
print yname/title =
*****
*****'/rlabels = 'Outcome:'/'format
= A8.

```

```

do if (ovals = 2).
compute nmsd = {yname, 'Analysis'}.
print rcd/title = 'Coding of binary DV for
analysis:'/cnames = nmsd/format = F9.2.
end if.
end if.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(yamat)/n)).
compute LL3 = yamat*&ln(pt2)+(1-yamat)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= yamat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).

```

```

compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute bootcoef={bootcoef,t(bmat)}.
compute LL1 = LL2.
compute covmat=varb.
end if.
do if (ovals <> 2).
compute bmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*bmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-(((1-r2)*(n-1))/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).
compute bootcoef={bootcoef,t(bmat)}.
end if.
do if (k = 1).
do if (ovals = 2).
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.

```

```

compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).
compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms2/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute
opdirsum=outp((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
end if.
do if (ovals <> 2).
.
compute hcinvXtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtXt*( xmat
(i3,:)).

```



```

end loop.
loop i3=1 to k3.
compute xmat(:,i3) = ( resid(:,ncol( resid
))&/(1-h))&* xmat(:,i3).
end loop.
end if.
do if ( hc3 <> 1).
loop i3=1 to k3.
compute xmat(:,i3) = sqrt( mse )&* xmat(:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(bmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/(ncol(xmat)-1).
compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat((2+nmeds):(2+nmeds+(nx-1)),:)=
lmat2.
compute fratiodr =
(t(t(lmat)*bmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*bmat)))/nx.
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute df2=(n-ncol(xmat)).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), df2, pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = bmat&/sqrt(diag(covmat)).
compute rnms2 = {'Constant'; t(mname);
t(xname)}.
compute op = {bmat, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat)))) }.

```

```

compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp))))*(xp2/(temp-
(2/3)+(.11/temp)))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))*op(:,2).
compute op={op,temp1,temp2}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms2/format =
F10.4.
compute
opdirsum=op((2+nmeds):(1+nmeds+nx),:).
compute cov((ncol(cov)-
nm+1):ncol(cov),(ncol(cov)-
nm+1):ncol(cov))=covmat(2:(1+nm),2:(1+nm)).
compute xmat = {ones,mdata}.
do if (ncovs > 0).
compute xmat = {xmat, cdata}.
end if.
compute btmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*btmat)**2).
compute r2direct = r2-((sstotal-
ssresid2)/sstotal).
compute pfr = 1-fcdf(fratiodr,(nx),(n-1-nx-nm-
ncovs)).
compute r2direct = {r2direct, fratiodr, nx, (n-1-
nx-nm-ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms2/cnames=rnms2/form
at= F10.4.
end if.
end if.

```

```

do if (k = 1 and toteff = 1).
print yname/title = '*****'
TOTAL EFFECT MODEL
*****'/rlabels =
'Outcome:.'/format = A8.
compute xmat = {ones, xdata}.
do if (covmy=1).
compute xmat=xmat(:,1:(ncol(xmat)-ncovs)).
end if.
compute ymat = ydata.
do if (ovals = 2).
compute pt2 = make(n,1,(csum(ymat)/n)).
compute LL3 = ymat*&ln(pt2)+(1-ymat*)&ln(1-
pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(xmat),1,0).
compute LL1 = 0.

.
compute pt1lp= pt1.
compute xlp= xmat.
compute ylp= ymat.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.

```

```

compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
compute bmat=b.
compute LL1 = LL2.
compute covmat=varb.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(bmat)-
1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
compute rnms = {'Constant'; t(xname)}.
compute tstat = bmat&/sqrt(diag(covmat)).

```

```

compute outp =
{bmat,sqrt(diag(covmat)),tstat,(2*(1-
cdfnorm(abs(tstat))))}.
compute temp=bmat-
abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
compute
temp=bmat+abs(xp2)&*sqrt(diag(covmat)).
compute outp = {outp, temp}.
print outp/title = 'Logistic Regression
Model'/rnames = rnms/clabels = 'Coeff' 'se' 'Z'
'p' 'LLCI' 'ULCI'/format= F10.4.
compute optotsum=outp(2:(1+nx),:).
end if.
do if (ovals <> 2).
compute cmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute sstotal = t(ymat-
(csum(ymat)/n))*(ymat-(csum(ymat)/n)).
compute resid=ymat-xmat*cmat.
compute ssresid = csum((resid)**2).
compute r2 = (sstotal-ssresid)/sstotal.
compute adjr2 = 1-((1-r2)*(n-1)/(n-ncol(xmat))).
compute mse=ssresid/(n-ncol(xmat)).

.
compute hcinvXtX=inv(t( xmat )* xmat ).
compute k3 = ncol( xmat ).
do if ( hc3 = 1).
compute h = xmat (:,1).
loop i3=1 to nrow( xmat ).
compute h(i3,1)= xmat (i3,:)*hcinvtX*t( xmat
(i3,:)).
end loop.
loop i3=1 to k3.
compute xmat (:,i3) = ( resid (:,ncol( resid
)))/(1-h))&* xmat (:,i3).
end loop.
end if.
do if ( hc3 <> 1).

```

```

loop i3=1 to k3.
compute xmat(:,i3) = sqrt( mse )&* xmat(:,i3).
end loop.
end if.
compute hcdat= xmat.
compute covmat=hcinvXtx*t( xmat )* xmat
*hcinvXtX
.
compute lmat = ident(nrow(cmat)).
compute lmat = lmat(:,2:ncol(lmat)).
compute fratio =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/(ncol(xmat)-1).
compute pfr = 1-fcdf(fratio,(ncol(xmat)-1),(n-
ncol(xmat))).
compute op = {sqrt(r2), r2, mse, fratio,
(ncol(xmat)-1), (n-ncol(xmat)), pfr}.
print op/title = 'Model Summary'/format =
F10.4/clabels = 'R', 'R-sq', 'MSE', 'F', 'df1', 'df2',
'p'.
compute tratio = cmatrix/sqrt(diag(covmat)).
compute op = {cmatrix, sqrt(diag(covmat)), tratio,
(2*(1-tcdf(abs(tratio), n-ncol(xmat))))}.
compute temp=(n-ncol(xmat)).
compute temp = (temp* (exp((temp-
(5/6))*((xp2/(temp-
(2/3)+(.11/temp)))*xp2/(temp-
(2/3)+(.11/temp))))-1)).
compute temp1 = op(:,1)-
sqrt(abs(temp))&*op(:,2).
compute temp2 =
op(:,1)+sqrt(abs(temp))&*op(:,2).
compute op={op,temp1,temp2}.
compute rnms = {'Constant'; t(xname)}.
print op/title = 'Model'/clabels = 'coeff', 'se', 't',
'p', 'LLCI', 'ULCI'/rnames = rnms/format =
F10.4.
compute optotsum=op(2:(1+nx),:).
compute ssresid2 = ssresid.
compute r2total = r2.

```

```

compute lmat = make(ncol(xmat),nx,0).
compute lmat2 = ident(nx).
compute lmat(2:(1+nx),:) = lmat2.
compute fratio3 =
(t(t(lmat)*cmat)*inv(t(lmat)*covmat*lmat)*((t(lm
at)*cmat)))/nx.
do if (ncovs > 0).
compute xmat = {ones, cdata}.
compute ctmat =
inv(t(xmat)*xmat)*t(xmat)*ymat.
compute ssresid2 = csum((ymat-
xmat*ctmat)**2).
compute r2total = r2-((sstotal-ssresid2)/sstotal).
end if.
compute pfr = 1-fcdf(fratio3,(nx),(n-1-nx-
ncovs)).
compute r2tot2 = {r2total, fratio3, nx, (n-1-nx-
ncovs), pfr}.
end if.
do if (covcoeff=1).
print covmat/title='Covariance matrix of
regression parameter
estimates'/rnames=rnms/cnames=rnms/format
= F10.4.
end if.
end if.
compute mns((nrow(mns)-
nm+1):nrow(mns),1)=bmat(2:(1+nm),1).
compute tmp = 1.
loop i = 1 to (nm*nx).
compute
boots(k,i)=mns(i,1)*mns(((nm*nx)+tmp),1).
do if ((i/nx)=trunc(i/nx)).
compute
boots(k,((nm*nx)+tmp))=r2b(tmp,1)*mns(((nm*
nx)+tmp),1).
compute tmp = tmp+1.
end if.
end loop.
do if (savboot = 1 and k > 1).

```

```

compute bootstrp((k-1),:)=bootcoef.
end if.
end loop.
do if (method = 1).
compute llci=make(1,ncol(boots),-999).
compute ulci=make(1,ncol(boots),-999).
compute indeff = t(boots(1,1:(nm*nx))).
compute indeff22=t(boots(1,:)).
compute omni = t(boots(1,((nm*nx)+1) :
ncol(boots))).
do if (nrow(boots) > 1).
compute boots = boots(2:nrow(boots),:).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
loop i = 1 to ncol(boots).
.
compute temp = boots(:,i).
compute temp(GRADE( boots(:,i) )) = boots(:,i).
compute badlo = 0.
compute badhi = 0.
do if ( indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) <> 9999).
compute pv=csum(temp <
(indeff22(i,1)*bconoff)+(9999*(1-bconoff))
)/boot.
compute ppv = pv.
do if (pv > .5).
compute ppv = 1-pv.
end if.
compute y5=sqrt(-2*ln(ppv)).
compute
xp=y5+((((y5*p4+p3)*y5+p2)*y5+p1)*y5+p0)/(((
(y5*q4+q3)*y5+q2)*y5+q1)*y5+q0).
do if (pv <= .5).
compute xp = -xp.
end if.
compute cilow=rnd(boot*(cdfnorm(2*xp+xp2))).
compute cihigh=trunc(boot*(cdfnorm(2*xp+(-
xp2))))+1.

```



```

do if (cilow < 1).
compute cilow = 1.
compute booterr=1.
compute badlo = 1.
end if.
do if (cihigh > boot).
compute cihigh = boot.
compute booterr=1.
compute badhi = 1.
end if.
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if.
do if ( (indeff22(i,1)*bconoff)+(9999*(1-
bconoff)) = 9999).
compute llcit=temp(cilow,1).
compute ulcit=temp(cihigh,1).
end if
.
compute llci(1,i)=llcit.
compute ulci(1,i)=ulcit.
do if (badlo = 1 and llcit <> priorlo).
compute badend={badend, llcit}.
compute priorlo = llcit.
end if.
do if (badhi = 1 and ulcit <> priorhi).
compute badend={badend, ulcit}.
compute priorhi = ulcit.
end if.
end loop.
compute llci = t(llci).
compute ulci= t(ulci).
compute llci={llci,ulci}.
end if.
end if.
do if (method = 2).
compute tmp=1.
loop i = 1 to (nm*nx).
compute
indeff(i,1)=mns(i,1)&*mns(((nm*nx)+tmp),1).

```

```

do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = sqrt(-
2*ln(uniform(reps,nrow(cov))))&*cos((2*3.1415
9265358979)*uniform(reps,nrow(cov))).
compute boots=boots*chol(cov).
loop i = 1 to ncol(boots).
compute boots(:,i)=boots(:,i)+mns(i,1).
end loop.
compute tmp=1.
loop i = 1 to (nm*nx).
compute
boots(:,i)=boots(:,i)&*boots(:,((nm*nx)+tmp)).
compute temp = boots(:,i).
compute temp(GRADE(boots(:,i))) = boots(:,i).
compute boots(:,i) = temp.
do if ((i/nx)=trunc(i/nx)).
compute tmp = tmp+1.
end if.
end loop.
compute boots = boots(:,1:(nm*nx)).
compute bootse = t(sqrt(((reps*cssq(boots))-
(csum(boots)**2))/((reps-1)*reps))).
compute llci =
t(((boots(cilow,:));(boots(cihigh,:)))).
end if.
do if (toteff = 0).
print/title = '***** DIRECT AND
INDIRECT EFFECTS *****'.
else.
print/title = '***** TOTAL, DIRECT,
AND INDIRECT EFFECTS *****'.
compute rnmssum=rnms(2:(1+nx),1).
do if (ovals <> 2).
print optotsum/title = 'Relative total effects of X
of Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.

```

```

print r2tot2/title = 'Omnibus test of total effect of
X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print optotsum/title = 'Relative total effects of X
on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
end if.
compute
rnmssum=rnms2((2+nmeds):(1+nmeds+nx),1).
do if (ovals <> 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff', 'se', 't', 'p', 'LLCI',
'ULCI'/rnames = rnmssum/format = F10.4.
print r2direct/title = 'Omnibus test of direct
effect of X on Y'/clabels = 'R-sq', 'F', 'df1', 'df2',
'p'/format = F10.4.
end if.
do if (ovals = 2).
print opdirsum/title = 'Relative direct effects of
X on Y'/clabels = 'coeff' 'se' 'Z' 'p' 'LLCI'
'ULCI'/rnames = rnmssum/format = F10.4.
end if.
print/title= '====='/space=0.
do if (omnibus = 1 and (method = 1 or method
= 3)).
compute indlbs = {indlbs; 'Omnibus'}.
end if.
loop i = 1 to nm.
compute op = mname(1,i).
print op/title = 'Relative indirect effect(s) of X on
Y through:.'/format = A8/space=0.
do if (method = 2).
compute clbs = {'Effect', 'SE MC', 'MC LLCI',
'MC ULCI'}.
compute op = {indeff, bootse(1:(nm*nx)), lici}.
end if.

```

```

do if (method = 1).
compute clbs = {'Effect', 'SE(boot)', 'LLCI',
'ULCI'}.
compute op = indeff.
do if (samples > 0).
compute op = {indeff, bootse(1:(nm*nx),:),
llci(1:(nm*nx),:)}).
end if.
end if.
compute temp = op((((i-1)*nx)+1):(nx*i),:).
do if ((method = 1) and omnibus = 1).
do if (samples = 0).
compute temp={temp;omni(i,1)}.
end if.
do if (samples > 0).
compute temp = {temp; omni(i,1),
bootse(((nm*nx)+i),:), llci(((nm*nx)+i),:)}).
end if.
end if.
print temp/title = ' '/rnames = indlbs/cnames =
clbs/format = F10.4 /space=0.
print/title = '-----'.
end loop.
end if.
do if (savboot = 1).
compute bootstrp = bootstrp(:,2:ncol(bootstrp)).
save bootstrp/outfile = *.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
END IF.
DO IF (mdpbe=1).
compute nm={yname,xname,mnames}.
do if (ncovs > 0).

```

```

compute
nm={yname,cnames,xname,mnames}.
end if.

compute tpx=xname.
compute x = dd(:,2:ncol(dd)).
compute tpy = yname.
compute tempy= dd(:,1).
compute n = nrow(dd).
compute nx=1.
compute criterr=0.
compute ddd = {'D1', 'D2', 'D3', 'D4', 'D5', 'D6',
'D7', 'D8', 'D9'}.
compute ddd1 = {'int_1', 'int_2', 'int_3', 'int_4',
'int_5', 'int_6', 'int_7', 'int_8', 'int_9'}.
compute dumok = 0.
compute mcfoc=mcx.
compute mcmmod=mcm.
compute mcloc=(mcfoc>0).
compute con = make(n,1,1).
compute ncovs=ncol(x)-2.
do if (mcfoc > 0 or mcmmod > 0).
compute temp = dd.
compute temp(GRADE(dd(:,(ncol(dd)-
mcloc))),:) = dd.
compute dd = temp.
compute dummy = design(dd(:,ncol(dd)-
mcloc)).
compute nvls = ncol(dummy).
compute nnvls = csum(dummy).
compute toosmall=rsum(nnvls < 2).
compute mnvls = cmin(t(nnvls)).
do if ((rsum(nnvls < 2)) > 0).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=37.
end if.
do if (nvls > 10).
compute criterr = 1.
compute errs=errs+1.
compute runerrs(errs,1)=36.

```

```

end if.
do if (criterr=0).
compute dumok = 1.
compute nnvls=make(nvls,1,0).
compute nnvls(1,1)=dd(1,ncol(dd)-mcloc).
compute temp = 2.
loop i = 2 to n.
do if (dd(i,ncol(dd)-mcloc) <> nnvls((temp-
1),1)).
compute nnvls(temp,1)=dd(i,ncol(dd)-mcloc).
compute temp = temp+1.
end if.
end loop.
compute dummy = dummy(:,2:ncol(dummy)).
do if (mcfoc=4 or mcmmod=4).
compute minus1=make(1,ncol(dummy),-1).
loop k = 1 to n.
do if (rsum(dummy(k,:)) = 0).
compute dummy(k,:)=minus1.
end if.
end loop.
end if.
do if (mcfoc = 2 or mcfoc = 3) or (mcmmod = 2 or
mcmmod = 3)).
loop k = 1 to n.
do if (rsum(dummy(k,:)) > 0).
loop i = 1 to ncol(dummy).
do if (dummy(k,i) = 0).
compute dummy(k,i) = 1.
else.
break.
end if.
end loop.
end if.
end loop.
do if (mcfoc = 3 or mcmmod=3).
compute conmat1={-8,1,1,1,1,1,1,1,1,1; 0,-
7,1,1,1,1,1,1,1,1,1; 0,0,-6,1,1,1,1,1,1,1; 0,0,0,-
5,1,1,1,1,1,1,1; 0,0,0,0,-4,1,1,1,1,1; 0,0,0,0,0,-
3,1,1,1,1; 0,0,0,0,0,0,-2,1,1; 0,0,0,0,0,0,0,-1,1}.

```

```

loop i = 1 to 8.
compute conmat1(i,:)=conmat1(i,:)/(10-i).
end loop.
compute conmat1=t(conmat1((10-nvls):8,(10-
nvls):9)).
loop k=1 to n.
compute
dummy(k,:)=conmat1((rsum(dummy(k,:))+1),:).
end loop.
end if.
end if.
compute nx = ncol(dummy).
compute xname = ddd(1,1:nx).
compute xdata = dummy.
compute xname = ddd(1,1:nx).
compute xname2={nm(1,(ncol(nm)-mcloc)),
xname}.
compute indlbs = t(xname).
compute dummat = make((nx+1),nx,0).
compute
dummat((2:nrow(dummat)),:)=ident(nx).
do if (mcfoc = 2 or mcmmod=2).
loop i = 2 to nrow(dummat).
loop j = 1 to (i-1).
compute dummat(i,j) = 1.
end loop.
end loop.
end if.
do if (mcfoc = 3 or mcmmod=3).
compute dummat=conmat1.
end if.
do if (mcfoc = 4 or mcmmod = 4).
compute dummat(1,:)=minus1.
end if.
compute dummat={nnvls, dummat}.
end if.
end if.
do if (criterr=0).
compute y = dd(:,1).
compute ovals = ncol(design(dd(:,1))).

```

```

compute itprob = 0.
do if (ovals = 2).
compute omx = cmax(y(:,1)).
compute omn = cmin(y(:,1)).
compute y(:,1) = (y(:,1) = omx).
compute rcd = {omn, 0; omx, 1}.
end if.
compute sstotal = csum((y-(csum(y)/n))&**2).
compute outv = t(nm(1,1)).
compute mdtr = nm(1,ncol(dd)).
compute fciv = nm(1,(ncol(dd)-1)).
do if (mcfoc <> 0 or mcmod <> 0).
compute inter=make(n,nx,0).
loop i = 1 to nx.
compute inter(:,i)=dummy(:,i)&*dd(:,ncol(dd)-
(1-mcloc)).
end loop.
do if (ncovs = 0).
compute x={con,dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
do if (ncovs > 0).
compute
x={con,dd(:,2:(1+ncovs)),dd(:,ncol(dd)-(1-
mcloc)),dummy}.
compute x={x,inter}.
compute nms={t(nm(1,2:(1+ncovs)));
nm(1,ncol(dd)-(1-mcloc));
t(ddd(1,1:ncol(dummy)));t(ddd1(1,1:ncol(dumm
y))))}.
end if.
end if.
compute xmns = csum(x)/n.
compute focvals = {1,0,0}.
compute highwarn = 0.
compute lowwarn = 0.

```



```

do if (ncol(x) > (2*nx+2)).
compute covmns = {1, xmns(1,2:(ncol(x)-1-
(2*nx)))}.
compute focvals = {covmns,0,0}.
end if.
compute dfres = n-ncol(x).
do if (ovals = 2).
compute pt2 = make(n,1,(csum(y)/n)).
compute LL3 = y*&ln(pt2)+(1-y)&ln(1-pt2).
compute LL3 = -2*csum(LL3).
compute pt1 = make(n,1,0.5).
compute bt1 = make(ncol(x),1,0).
compute LL1 = 0.

.

compute pt1lp= pt1.
compute xlp= x.
compute ylp= y.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp&*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.000000000000001) or (pt1lp >
.999999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .999999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).

```

```

compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
do if (jjj > iterate).
compute itprob = 2.
end if.
compute tval=xp2*xp2.
else if (ovals <> 2).
compute invXtX=inv(t(x)*x).
compute b = invXtX*t(x)*y.
compute resid = y-(x*b).
compute ssresid=csum(resid*resid).
compute msresid=ssresid/dfres.
compute varb = msresid*invXtX.
compute
seint=sqrt(varb(nrow(varb),nrow(varb))).
compute k3 = nrow(b).
do if (hc3 = 1).
compute xhc=x.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)= xhc(i3,:)*invXtX*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).
end loop.
compute varb=(invXtX*t(xhc)*xhc*invXtX).
end if.
compute seb = sqrt(diag(varb)).
compute r2 = 1-(ssresid/sstotal).

```

```

do if (mcfoc <> 0 or mcmod <> 0).
compute ytp=y.
compute xtp=x(:,1:(ncol(x)-nx)).
compute btp = inv(t(xtp)*xtp)*t(xtp)*ytp.
compute residt = ytp-(xtp*btp).
compute ssresidt=csum(residt*residt).
compute r2noint = 1-(ssresidt/sstotal).
end if.
compute pr = ncol(x)-1.
compute lmat = ident(nrow(b)).
compute lmat = lmat(:,2:ncol(lmat)).
compute f =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b)))/
pr).
compute pf = 1-fcdf(f,pr,dfres).
compute pf = {sqrt(r2),r2,msresid,f,pr,dfres,pf}.
compute xd=abs(xp2).
end if.
do if (itprob = 0).
compute tstat = b&/seb.
do if (ovals <> 2).
compute p = 2*(1-tcdf(abs(tstat), dfres)).
else if (ovals = 2).
compute p = 2*(1-cdfnorm(abs(tstat))).
end if.
compute outp = {b,seb,tstat,p}.
compute nms = {'constant'; nms; 'interact'}.
do if (ovals <> 2).
compute tval = (dfres* (exp((dfres-
(5/6))*((xd/(dfres-(2/3)+(.11/dfres)))*(xd/(dfres-
(2/3)+(.11/dfres)))))-1)).
compute outp={outp,(b-
sqrt(tval)&*seb),(b+sqrt(tval)&*seb)}.
end if.
compute bb = tval.
do if (dumok = 1).
do if (mcfoc > 0).
print dummat/title = 'Coding of categorical X
variable for analysis:'/cnames = xname2/format
= F5.2.

```

```

else if (mcmod > 0).
print dummat/title = 'Coding of categorical M
variable for analysis: '/cnames = xname2/format
= F5.2.
end if.
end if.
print yname/title =
*****
*****'/rlabels = 'Outcome: '/format
= A8.
do if (ovals = 2).
compute nmsd = {outv, 'Analysis'}.
print rcd/title = 'Coding of binary Y for
analysis: '/cnames = nmsd/format = F10.4.
compute outp = {outp,(outp(:,1)-
sqrt(tval)&*outp(:,2)),(outp(:,1)+sqrt(tval)&*outp
(:,2))}.
compute LLdiff = LL3-LL2.
compute pvalue=1-chicdf(LLdiff,(nrow(b)-1)).
compute LL4 = LL2.
compute mcF = LLdiff/LL3.
compute cox = 1-exp(-LLdiff/n).
compute nagel = cox/(1-exp(-(LL3)/n)).
compute pf = {LL2, LLdiff, pvalue, mcF, cox,
nagel, n}.
print pf/title = 'Logistic Regression
Summary'/clabels = '-2LL' 'Model LL' 'p-value'
'McFadden' 'CoxSnell' 'Nagelkrk' 'n'/format
F10.4.
print outp/title = 'Logistic Regression
Model'/rnames = nms/clabels 'Coeff' 'se' 'Z' 'p'
'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
compute varbtmp=varb.
compute btmp=b.

```

```

compute LL2f=LL2.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=x(:,1:(ncol(x)-nx)).
compute bt1 = make(ncol(xlp),1,0).

.

compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(y-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.
end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .00000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)*&ln(1-
pt1lp).
compute LL2 = -2*csum(LL).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.

```

```

compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL2f.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(LLdiff,nx).
compute rcha={LLdiff,nx,pchi}.
end if.
do if (ovals <> 2).
print pf/title = 'Model Summary'/clabels = 'R' 'R-
sq' 'MSE' 'F' 'df1' 'df2' 'p'/format F10.4.
compute lmat=make(nrow(b),1,0).
compute lmat(nrow(lmat),1)=1.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
1).
compute
rcha=((b(nrow(b),1)/seint)*(b(nrow(b),1)/seint))*
(1-r2)/dfres.
compute rcha = {rcha, fcha, 1, dfres,
outp((pr+1),4)}.
print outp/title = 'Model'/rnames = nms/clabels
'coeff' 'se' 't' 'p' 'LLCI' 'ULCI'/format F10.4.
do if (covcoeff=1).
print varb/title='Covariance matrix of regression
parameter
estimates'/rnames=nms/cnames=nms/format=
F10.4.
end if.
do if (mcfoc > 0 or mcmod > 0).
compute rcha=r2-r2noint.
compute lmat=make((nrow(b)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha =
(t(t(lmat)*b)*inv(t(lmat)*varb*lmat)*((t(lmat)*b))/
nx).
compute pvalr2c=1-fcdf(fcha,nx,dfres).

```

```

compute rcha = {rcha, fcha, nx, dfres,pvalr2c}.
end if.
end if.
do if (mcfoc > 0 or mcmod > 0).
compute intkey = {'a', 'b', 'c', 'd', 'e'}.
loop i = 1 to nx.
compute intkey={intkey; ddd1(1,i), ' : ', ddd(1,i),
' X ', nm(1,ncol(dd)-(1-mcloc))}.
end loop.
compute intkey=intkey(2:nrow(intkey),:).
print intkey/title='Product terms key:.'/format =
a8.
end if.
do if (ovals <> 2).
print rcha/title = 'R-square increase due to
interaction:.'/clabels 'R2-chng' 'F' 'df1' 'df2'
'p'/format F10.4.
else if (ovals = 2).
print rcha/title = 'Likelihood ratio test for
interaction:.'/clabels 'Chi-sq' 'df' 'p'/format F10.4.
end if.
print/title =
*****
*****'.
compute mdvar = x(:,(ncol(x)-1)).
do if (mcfoc = 0 and mcmod = 0).
compute g1 = b((ncol(x)-2),1).
compute g3 = b(ncol(x),1).
compute vg1 = varb((ncol(x)-2),(ncol(x)-2)).
compute vg3 = varb(ncol(x),ncol(x)).
compute covg1g3 = varb((ncol(x)-2), ncol(x)).
end if.
do if (mcfoc > 0 or mcmod > 0).
compute mdvar = x(:,(ncol(x)-(2*nx))).
end if.
compute mdmin = cmin(mdvar).
compute mdmax = cmax(mdvar).
compute fvar = x(:,(ncol(x)-2)).
compute nval = ncol(design(mdvar)).
compute fvmin = cmin(fvar).

```

```

compute fvmax = cmax(fvar).
do if (mmodval = 999).
compute mnmd = csum(mdvar)/n.
compute tmp = make(n,1,mnmd).
compute sdmd = sqrt(csum(((mdvar-
tmp)**2))/(n-1)).
compute probeval = {mnmd-sdmd; mnmd;
mnmd+sdmd}.
do if (probeval(1,1) < mdmin).
compute lowwarn = 1.
compute probeval(1,1)=mdmin.
end if.
do if (probeval(nrow(probeval),1) > mdmax).
compute highwarn = 1.
compute probeval(nrow(probeval),1)=mdmax.
end if.
do if (quantile = 1).
compute tmp = mdvar.
compute tmp(GRADE(mdvar(:,1)),:)= mdvar.
compute mdvar = tmp.
compute
probeval={mdvar(trunc(0.1*n),1);mdvar(trunc(0
.25*n),1);mdvar(trunc(0.5*n),1);mdvar(trunc(0.7
5*n),1);mdvar(trunc(0.9*n),1)}.
end if.
end if.
do if (nval = 2).
compute probeval = make(2,1,0).
compute probeval(1,1)=cmin(mdvar).
loop i = 1 to n.
do if (mdvar(i,1) <> probeval(1,1)).
compute probeval(2,1) = mdvar(i,1).
BREAK.
end if.
end loop.
end if.
do if (mmodval <> 999).
compute probeval = mmodval.
end if.
compute outp = make(nrow(probeval),7,0).

```



```

do if (mcfoc > 0 or mcmmod > 0).
compute focvals=make(1,ncol(x)+1,1).
do if (mcfoc > 0).
print/title = 'Conditional effect of X on Y at
values of the moderator:'.
print/title = ' '/space=0.
compute rnn2=mdtr.
compute matt=make(nx,6,0).
end if.
loop jj=1 to nrow(probeval).
do if (mcfoc > 0).
loop ii=1 to nx.
compute g1=b((ncol(x)-(2*nx)+ii),1).
compute g3=b((ncol(x)-nx+ii),1).
compute vg1=varb((ncol(x)-(2*nx)+ii),(ncol(x)-
(2*nx)+ii)).
compute vg3=varb((ncol(x)-nx+ii),(ncol(x)-
nx+ii)).
compute covg1g3=varb((ncol(x)-
(2*nx)+ii),(ncol(x)-nx+ii)).
compute x2 = probeval(jj,1).
compute w1 = g1+g3*x2.
compute varw1 =
vg1+(2*x2*covg1g3)+((x2*x2)*vg3).
compute sew1 = sqrt(varw1).
compute t1 = w1/sew1.
do if (ovals <> 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-tcdf(abs(t1), dfres)).
compute cnms = {'Coeff', 'se', 't', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)= {w1,sew1,t1,p,llci,ulci}.
else if (ovals = 2).
compute LLCI = (w1-sqrt(tval)&*sew1).
compute ULCI = (w1+sqrt(tval)&*sew1).
compute p = 2*(1-cdfnorm(abs(t1))).
compute cnms = {'Coeff', 'se', 'Z', 'p', 'LLCI',
'ULCI'}.
compute matt(ii,:)= {w1,sew1,t1,p,llci,ulci}.

```

```

end if.
end loop.
compute rnms=t(xname).
compute mdvalpr=probeval(jj,1).
print mdvalpr/title = 'Moderator
value: '/rnames=rnn2/format= F10.4 /space=0.
print matt/title='
'/cnames=cnms/rnames=rnms/format= F10.4
/space=0.
compute xprob=x(:,(ncol(x)-(2*nx)))-mdvalpr.
loop kk = 1 to nx.
compute xprob={xprob,
(xprob(:,1)&*x(:,(ncol(x)-(2*nx)+kk)))}.
end loop.
compute xprob={x(:,1:((ncol(x)-(2*nx))-
1)),xprob}.
do if (ovals <> 2).
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute residc = y-(xprob*bmultc).
compute ssresidc=csum(residc&*residc).
compute r2c = r2-(1-(ssresidc/sstotal)).
compute fcha2=(dfres*r2c)/(nx*(1-r2)).
do if (hc3 = 1).
loop kk = 1 to nx.
compute xprob={xprob, x(:,(ncol(x)-
(2*nx)+kk))}.
end loop.
compute bmultc =
inv(t(xprob)*xprob)*t(xprob)*y.
compute k3 = nrow(bmultc).
compute xhc=xprob.
compute h = xhc(:,1).
loop i3=1 to n.
compute h(i3,1)=
xhc(i3,1)*inv(t(xprob)*xprob)*t(xhc(i3,:)).
end loop.
loop i3=1 to k3.
compute xhc(:,i3) = (resid(:,ncol(resid))&/(1-
h))&*xhc(:,i3).

```

```

end loop.
compute
varbc=(inv(t(xprob)*xprob)*t(xhc)*xhc*inv(t(xprob)*xprob)).
compute lmat=make((nrow(bmultc)-nx),nx,0).
compute lmat2=ident(nx).
compute lmat={lmat;lmat2}.
compute fcha2 =
(t(t(lmat)*bmultc)*inv(t(lmat)*varbc*lmat)*((t(lmat)*bmultc)))/nx).
end if.
compute pvalr2cc=1-fcdf(fcha2,nx,dfres).
compute rcha2 = {r2c, fcha2, nx,
dfres,pvalr2cc}.
print rcha2/title = 'Test of equality of conditional
means at this value of the moderator'/clabels
'R2-chng' 'F' 'df1' 'df2' 'p'/format F10.4.
else if (ovals = 2).
compute btmp=b.
compute pt1lp = make(n,1,0.5).
compute ylp=y.
compute xlp=xprob.
compute bt1 = make(ncol(xlp),1,0).

.
compute pt1lp= pt1lp.
compute xlp= xlp.
compute ylp= ylp.
loop jjj = 1 to iterate.
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute b = bt1+inv(t(xlp)*vt1*xlp)*t(xlp)*(ylp-
pt1lp).
compute pt1lp = 1/(1+exp(-(xlp*b))).
compute itprob = csum((pt1lp <
.00000000000001) or (pt1lp >
.99999999999999)).
do if (itprob > 0).
loop kkk = 1 to nrow(pt1lp).
do if (pt1lp(kkk,1) = 1).
compute pt1lp(kkk,1) = .99999999999999.

```

```

end if.
do if (pt1lp(kkk,1) = 0).
compute pt1lp(kkk,1) = .000000000000001.
end if.
end loop.
compute itprob = 0.
end if.
do if (itprob = 0).
compute LL = ylp*&ln(pt1lp)+(1-ylp)&ln(1-
pt1lp).
compute LL2 = -2*csum(ll).
end if.
do if (abs(LL1-LL2) < converge).
compute vt1 = mdiag(pt1lp*(1-pt1lp)).
compute varb = inv(t(xlp)*vt1*xlp).
compute seb = sqrt(diag(varb)).
break.
end if.
compute bt1 = b.
compute LL1 = LL2.
end loop
.
compute LLdiff=LL2-LL4.
compute b=btmp.
compute varb=varbtmp.
compute pchi=1-chicdf(lldiff,nx).
compute rcha2 = {LLdiff, nx,pchi}.
print rcha2/title = 'Test of equality of log odds
conditioned on this moderator value'/clabels
'Chi-sq' 'df' 'p'/format F10.4.
end if.
end if.
compute
tttt=make(nrow(dummat),1,probeval(jj,1)).
compute tttt={tttt,dummat(:,2:ncol(dummat))}.
loop kkk=1 to nx.
compute tttt={tttt,tttt(:,1)*tttt(:,1+kkk)}.
end loop.
compute ones=make(nrow(dummat),1,1).
do if (ncol(x) > (2*nx+2)).

```

```

compute
covmnmat=make(nrow(tttt),ncol(covmns),0).
loop kkk=1 to nrow(tttt).
compute covmnmat(kkk,:)=covmns.
end loop.
compute tttt={covmnmat,tttt}.
else.
compute tttt={ones,tttt}.
end if.
compute focvals={focvals;dummat(:,1),tttt}.
do if (mcfoc > 0).
compute yhat={dummat(:,1),(tttt*b)}.
compute cmnms={fciv, 'yhat'}.
do if (ovals <> 2).
print yhat/title = 'Estimated conditional means
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (ovals = 2).
print yhat/title = 'Estimated conditional log odds
at this value of the
moderator'/cnames=cmnms/format F10.4.
end if.
do if (jj <> nrow(probeval)).
print/title='-----'/space=0.
end if.
end if.
end loop.
compute focvals=focvals(2:nrow(focvals),:).
compute yhat=focvals(:,2:ncol(focvals))*b.
compute
focvals={focvals(:,1),focvals(:,(ncol(focvals)-
(2*nx))),yhat}.
compute cnms={fciv,mdtr,'yhat'}.
do if (mcmod > 0).
compute cnms={mdtr,fciv,'yhat'}.
end if.
end if.
do if (mcmod > 0).
compute outp=make((nx+1),7,0).

```

```

compute bcatm={b((ncol(x)-
(2*nx)),1);b((ncol(x)-nx+1):ncol(x),1)}.
compute outp(:,1)=dummat(:,1).
compute bcatcov=varb((ncol(x)-
nx):ncol(x),(ncol(x)-nx):ncol(x)).
compute bcatcov(1,1)=varb((ncol(x)-
(2*nx)),(ncol(x)-(2*nx))).
compute
bcatcov(2:nrow(bcatcov),1)=varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx)).
compute
bcatcov(1,2:nrow(bcatcov))=t(varb((ncol(x)-
nx+1):ncol(x),(ncol(x)-2*nx))).
loop i = 1 to nrow(dummat).
compute
catmval={1,dummat(i,2:ncol(dummat))}.
compute condeff=catmval*bcatm.
compute
condse=sqrt(catmval*bcatcov*t(catmval)).
compute outp(i,2:3)={condeff,condse}.
end loop.
compute outp(:,4)=outp(:,2)&/outp(:,3).
compute outp(:,5) = 2*(1-tcdf(abs(outp(:,4)),
dfres)).
compute outp(:,6) = (outp(:,2)-
sqrt(tval)&*outp(:,3)).
compute outp(:,7) =
(outp(:,2)+sqrt(tval)&*outp(:,3)).
do if (ovals <> 2).
compute cnmms = {xname2(1,1), 'coeff', 'se',
't', 'p', 'LLCI', 'ULCI'}.
end if.
do if (ovals = 2).
compute outp(:,5) = 2*(1-
cdfnorm(abs(outp(:,4)))).
compute cnmms = {xname2(1,1), 'coeff', 'se',
'Z', 'p', 'LLCI', 'ULCI'}.
end if.

```

```

print outp/title = 'Conditional Effect of Focal
Predictor in Groups Defined by the Moderator
Variable:.'/cnames=cnmms/format = F10.4.
end if.
do if (nval > 2 and (mmodval = 999) and
mcmmod = 0).
do if (quantile <> 1).
print/title = 'Moderator values are the sample
mean and plus/minus one SD from mean'.
else.
print/title = 'Values for quantitative moderators
are 10th, 25th, 50th, 75th, and 90th
percentiles'.
end if.
do if (highwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD above the mean was'.
print/title = 'replaced with the maximum
because one SD above the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
do if (lowwarn = 1 and quantile = 0).
print/title = 'NOTE: For the moderator values
above, one SD below the mean was'.
print/title = 'replaced with the minimum
because one SD below the mean is
outside'/space=0.
print/title = 'of the range of the data.'/space=0.
end if.
end if.
do if (nval = 2 and (mmodval = 999)).
print/title = 'The moderator variable is
dichotomous'.
end if.
do if (mmodval <> 999 and mcmmod = 0 and
((mmodval < mdmin) or (mmodval > mdmax))).
print/title = 'Warning: Moderator value specified
is outside of the range of the data'.
end if.

```

```

do if (plot = 1).
do if (mcfoc = 0 and mcmmod=0).
compute cnms = {t(nms((ncol(x)-2):(ncol(x)-
1),1)), 'yhat'}.
end if.
do if (ovals = 2).
compute prob =
exp(focvals(:,3))/(1+exp(focvals(:,3))).
compute focvals = {focvals, prob}.
compute cnms = {cnms, 'prob'}.
end if.
print/title =
*****
*****!

print/title = 'Data for visualizing conditional
effect of X on Y'.
print/title = 'Paste text below into a SPSS
syntax window and execute to produce
plot.'/space=0.
do if (ovals = 2).
do if (mcfoc=0).
print/title = 'DATA LIST
FREE/Technological_Advan Comm_Pract
logodds prob.' /space=1.
end if.
do if (mcmmod=0).

```



```

print/title = 'DATA LIST FREE/Comm_Pract
Technological_Advan logodds prob.'
/space=1.
end if.
else if (ovals <> 2).
do if (mcfoc=0).
print/title = 'DATA LIST
FREE/Technological_Advan Comm_Pract
Financial_Performance.' /space=1.
end if.
do if (mcmmod=0).
print/title = 'DATA LIST FREE/Comm_Pract
Technological_Advan Financial_Performance.'
/space=1.
end if.
end if.
print/title = 'BEGIN DATA.' /space=0.
print focvals/title = ' '/format = F10.4 /space=0.
print/title = 'END DATA.' /space=1.
do if (ovals <> 2 and mcmmod=0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH Financial_Performance BY
Comm_Pract.' /space=0.
end if.
do if (ovals <> 2 and mcfoc=0).
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
Financial_Performance BY
Technological_Advan.' /space=0.
end if.
do if (ovals = 2 and mcmmod=0).
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH logodds BY Comm_Pract.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Technological_Adva
n WITH prob BY Comm_Pract.' /space=0.
end if.
do if (ovals = 2 and mcfoc=0).

```

```

print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
logodds BY Technological_Advan.' /space=0.
print/title =
'GRAPH/SCATTERPLOT=Comm_Pract WITH
prob BY Technological_Advan.' /space=0.
end if.
do if (ncovs > 0).
print/title = 'NOTE: For data above, covariates
are set to their sample means.'.
end if.
end if.
end if.
do if (itprob = 1).
print/title = 'ERROR: There was a problem
during iteration.'.
end if.
do if (itprob = 2).
compute errs = errs+1.
compute runerrs(errs,1) = 22.
end if.
end if.
END IF.
end if.
do if (bad > 0).
compute note(notes,1) = 9.
compute notes = notes + 1.
end if.
print/title = '***** ANALYSIS
NOTES AND WARNINGS
*****'.
loop i = 1 to errs.
do if (runerrs(i,1) = 1).
print/title = 'ERROR: One of your declared
mediators is dichotomous. This procedure
can"t be used.'.
end if.
do if (runerrs(i,1) = 2).

```

```
print/title = 'ERROR: For model 6, this
procedure limits the number of mediators to
four.'.
end if.
do if (runerrs(i,1) = 3).
print/title = 'ERROR: For models 1, 2, and 3,
only a single variable can be listed in the M
list.'.
end if.
do if (runerrs(i,1) = 4).
print/title = 'ERROR: You requested a model
involving W but did not provide a valid W
variable name.'.
end if.
do if (runerrs(i,1) = 5).
print/title = 'ERROR: You requested a model
involving Z but did not provide a valid Z
variable name.'.
end if.
do if (runerrs(i,1) = 6).
print/title = 'ERROR: You requested a model
involving Q but did not provide a valid Q
variable name.'.
end if.
do if (runerrs(i,1) = 7).
print/title = 'ERROR: You requested a model
involving V but did not provide a valid V
variable name.'.
end if.
do if (runerrs(i,1) = 8).
print/title = 'ERROR: You specified a W
variable for a model that does not need it'.
end if.
do if (runerrs(i,1) = 9).
print/title = 'ERROR: You specified a Z variable
for a model that does not need it'.
end if.
do if (runerrs(i,1) = 10).
print/title = 'ERROR: You specified a Q variable
for a model that does not need it'.
```

```

end if.
do if (runerrs(i,1) = 11).
print/title = 'ERROR: You specified a V variable
for a model that does not need it.'.
end if.
do if (runerrs(i,1) = 12).
print/title = 'ERROR: The variable specified for
W has already been assigned.'.
end if.
do if (runerrs(i,1) = 13).
print/title = 'ERROR: The variable specified for
Z has already been assigned.'.
end if.
do if (runerrs(i,1) = 14).
print/title = 'ERROR: The variable specified for
Q has already been assigned.'.
end if.
do if (runerrs(i,1) = 15).
print/title = 'ERROR: The variable specified for
V has already been assigned.'.
end if.
do if (runerrs(i,1) = 16).
print/title = 'ERROR: You did not provide a
valid Y variable name.'.
end if.
do if (runerrs(i,1) = 17).
print/title = 'ERROR: The variable specified for
Y has already been assigned.'.
end if.
do if (runerrs(i,1) = 18).
print/title = 'ERROR: Model 6 requires more
than one mediator.'.
end if.
do if (runerrs(i,1) = 19).
print/title = 'ERROR: You have not specified a
valid model number.'.
end if.
do if (runerrs(i,1) = 20).
print/title = 'ERROR: At least one and only one
variable must be listed for X.'.

```

```
end if.  
do if (runerrs(i,1) = 21).  
print/title = 'ERROR: At least one and only one  
variable must be listed for Y.'.  
end if.  
do if (runerrs(i,1) = 22).  
print/title = 'ERROR: Iteration didn"t converge  
to a solution. Interpret results with caution.'.  
end if.  
do if (runerrs(i,1) = 23).  
print/title = 'ERROR: You specified a clustering  
variable that does not exist in your variable  
list.'.  
end if.  
do if (runerrs(i,1) = 24).  
print/title = 'ERROR: You specified a clustering  
variable that has already been assigned.'.  
end if.  
do if (runerrs(i,1) = 25).  
print/title = 'ERROR: One or more of your M  
variables is not listed in the variables list.'.  
end if.  
do if (runerrs(i,1) = 26).  
print/title = 'ERROR: A maximum of 20 cluster  
units is allowed. Use multilevel modeling  
instead.'.  
end if.  
do if (runerrs(i,1) = 27).  
print/title = 'ERROR: One of the variables in  
your model is a constant.'.  
end if.  
do if (runerrs(i,1) = 28).  
print/title = 'ERROR: Dichotomous Y is not  
permitted with WS option.'.  
end if.  
do if (runerrs(i,1) = 29).  
print/title = 'ERROR: Insufficient number of  
variables in vars= list when using WS option.'.  
end if.  
do if (runerrs(i,1) = 30).
```

```
print/title = 'ERROR: Too many variables in
vars= list when using WS option. Covariates
not allowed.'.
end if.
do if (runerrs(i,1) = 31).
print/title = 'ERROR: mmodval and wmodval
can"t both be set to zero with contrast option.'.
end if.
do if (runerrs(i,1) = 32).
print/title = 'ERROR: You did not provide a
valid X variable name.'.
end if.
do if (runerrs(i,1) = 33).
print/title = 'ERROR: PROCESS requires all
variable names to be eight characters or
fewer.'.
print/title = '      Please shorten variable names
and reexecute.'/space=0.
end if.
do if (runerrs(i,1)=34).
print/title = 'ERROR: X and M cannot both be
specified as multicategorical.'.
end if.
do if (runerrs(i,1)=35).
print/title = 'ERROR: Multicategorical variable
specification available only for models 1 and
4.'.
end if.
do if (runerrs(i,1)=36).
print/title = 'ERROR: Categorical variables
cannot have more than 10 categories.'.
end if.
do if (runerrs(i,1)=37).
print/title = 'ERROR: Each group must have at
least two cases.'.
end if.
do if (runerrs(i,1)=38).
print/title = 'ERROR: Cluster option not
available with multicategorical variables.'.
end if.
```

```

do if (runerrs(i,1)=39).
print/title = 'ERROR: A maximum of 10
variables is allowed in the M= list.'.
end if.
do if (runerrs(i,1) = 41).
print/title = 'ERROR: MCM option not permitted
in model 4.'.
end if.
end loop.
do if (errs = 0).
do if (boot > 1 or mc > 0).
do if (bconoff = 1 and boot > 1).
print boot/title = 'Number of bootstrap samples
for bias corrected bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (bconoff = 0 and boot > 1).
print boot/title = 'Number of bootstrap samples
for percentile bootstrap confidence
intervals:'/format = F8.0.
end if.
do if (mc > 1).
print mc/title = 'Number of samples for Monte
Carlo confidence intervals:'/format = F8.0.
end if.
do if (booterr = 1).
compute badend = badend(1,2:ncol(badend)).
print badend/title = 'WARNING: Bootstrap CI
endpoints below not trustworthy. Decrease
confidence or increase bootstraps'/format =
F10.4.
end if.
end if.
print conf/title = 'Level of confidence for all
confidence intervals in output:'/format = F8.2.
do if ((center = 1 or ws=1) and (ncol(centvar) >
1)).
compute centvar = centvar(1,2:ncol(centvar)).

```

```

print centvar/title = 'NOTE: The following
variables were mean centered prior to
analysis:.'/format = a8.
end if.
loop i = 1 to notes.
do if (note(i,1) = 1).
print/title = 'NOTE: Confidence level restricted
to between 50 and 99.9999%. 95% confidence
is provided in output'.
end if.
do if (note(i,1) = 2).
print/title = 'NOTE: Effect size measures for
indirect effects are not available for models
with dichotomous outcomes'.
end if.
do if (note(i,1) = 3).
print/title = 'NOTE: All standard errors for
continuous outcome models are based on the
HC3 estimator'.
end if.
do if (note(i,1) = 6).
print/title = 'NOTE: The number of bootstrap
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 7).
print/title = 'NOTE: The Johnson-Neyman
method is available only for Models 1 and 3'.
end if.
do if (note(i,1) = 8).
print/title = 'NOTE: The Johnson-Neyman
method cannot be used with a dichotomous
moderator'.
end if.
do if (note(i,1)=9).
print bad/title = 'NOTE: Some bootstrap
samples had to be replaced. The number of
such replacements was:'.
end if.
do if (note(i,1) = 11).

```



```

print nmiss/title = 'NOTE: Some cases were
deleted due to missing data. The number of
such cases was:'.
end if.
do if (note(i,1) = 12).
print /title = 'NOTE: Monte Carlo method
available only for models 4 and 5.
Bootstrapping was used instead.'.
end if.
do if (note(i,1) = 13).
print/title = 'NOTE: The number of Monte Carlo
samples was adjusted upward given your
desired confidence'.
end if.
do if (note(i,1) = 19).
print/title = 'NOTE: Effect sizes not available for
within-subject analyses.'.
end if.
do if (note(i,1) = 16).
print/title = 'NOTE: Normal theory tests not
available for within-subject analyses.'.
end if.
do if (note(i,1) = 17).
print/title = 'NOTE: Monte Carlo confidence
intervals not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 18 and warnrep=0).
print/title = 'WARNING: You have requested
OLS estimation with a dichotomous criterion.'.
print/title = 'Interpret model coefficients and
inferential statistics with caution.'/space=0.
compute warnrep=1.
end if.
do if (note(i,1) = 20).
print/title = 'NOTE: Saving of bootstrap
estimates not available for within-subject
analyses.'.
end if.
do if (note(i,1) = 22).

```

```
print/title = 'NOTE: Effect size option with
covariates requires covariates in models of M
and Y.'.
end if.
do if (note(i,1) = 23).
print/title = 'NOTE: Johnson-Neyman method
not available with multicategorical X'.
end if.
do if (note(i,1) = 24).
print/title = 'NOTE: MMODVAL option not
available with a multicategorical moderator'.
end if.
do if (note(i,1) = 25).
print/title = 'NOTE: Omnibus test for relative
indirect effects is not available with the MC
option.'.
end if.
do if (note(i,1) = 26).
print/title = 'NOTE: CONTRAST option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 27).
print/title = 'NOTE: EFFSIZE option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 28).
print/title = 'NOTE: NORMAL option not
available with multicategorical X.'.
end if.
do if (note(i,1) = 31).
print/title = 'NOTE: The TOTAL option is not
available when Y is dichotomous.'.
end if.
do if (note(i,1) = 32).
print/title = 'NOTE: Kappa-squared is disabled
from output as of version 2.16.'.
end if.
do if (note(i,1) = 33).
print/title = 'NOTE: COVMY option ignored in
models 1, 2, and 3.'.
```

| | | | |
|-----------|----------------|-------------|-------------|
| | | end if. | |
| | | end loop. | |
| | | end if. | |
| | | end matrix. | |
| Resources | Processor Time | | 00:00:01.00 |
| | Elapsed Time | | 00:00:01.02 |

Run MATRIX procedure:

***** PROCESS Procedure for SPSS Release 2.16.3 *****

Written by Andrew F. Hayes, Ph.D. www.afhayes.com
Documentation available in Hayes (2013). www.guilford.com/p/hayes3

Model = 1
Y = Financia
X = Comm_Pra
M = Technolo

Sample size
856

Outcome: Financia

Model Summary

| R | R-sq | MSE | F | df1 | df2 | p |
|-------|-------|-------|-----------|--------|----------|-------|
| .9124 | .8324 | .2035 | 1410.3858 | 3.0000 | 852.0000 | .0000 |

Model

| | coeff | se | t | p | LLCI | ULCI |
|----------|--------|-------|---------|-------|--------|--------|
| constant | -.3526 | .1206 | -2.9230 | .0036 | -.5893 | -.1158 |
| Technolo | .8043 | .0515 | 15.6064 | .0000 | .7032 | .9055 |
| Comm_Pra | .4948 | .0515 | 9.6007 | .0000 | .3936 | .5960 |
| int_1 | -.0535 | .0115 | -4.6718 | .0000 | -.0760 | -.0311 |

Product terms key:

int_1 Comm_Pra X Technolo

R-square increase due to interaction(s):

| | R2-chng | F | df1 | df2 | p |
|-------|---------|---------|--------|----------|-------|
| int_1 | .0043 | 21.8261 | 1.0000 | 852.0000 | .0000 |

Conditional effect of X on Y at values of the moderator(s):

| Technolo | Effect | se | t | p | LLCI | ULCI |
|----------|--------|-------|--------|-------|-------|-------|
| 2.7372 | .3482 | .0361 | 9.6572 | .0000 | .2774 | .4190 |
| 3.8107 | .2907 | .0361 | 8.0455 | .0000 | .2198 | .3617 |
| 4.8843 | .2333 | .0402 | 5.8054 | .0000 | .1544 | .3121 |

Values for quantitative moderators are the mean and plus/minus one SD from mean.

Values for dichotomous moderators are the two values of the moderator.

***** JOHNSON-NEYMAN TECHNIQUE *****

There are no statistical significance transition points within the observed range of the moderator.

Data for visualizing conditional effect of X on Y
Paste text below into a SPSS syntax window and execute to produce plot.

DATA LIST FREE/Comm_Pract Technological_Advan Financial_Performance.
BEGIN DATA.

| | | |
|--------|--------|--------|
| 2.7372 | 2.7372 | 2.8022 |
| 3.8107 | 2.7372 | 3.1760 |
| 4.8843 | 2.7372 | 3.5498 |
| 2.7372 | 3.8107 | 3.5083 |
| 3.8107 | 3.8107 | 3.8204 |
| 4.8843 | 3.8107 | 4.1325 |
| 2.7372 | 4.8843 | 4.2144 |
| 3.8107 | 4.8843 | 4.4648 |
| 4.8843 | 4.8843 | 4.7152 |

END DATA.

GRAPH/SCATTERPLOT=Comm_Pract WITH Financial_Performance BY
Technological_Advan.

***** ANALYSIS NOTES AND WARNINGS *****

Level of confidence for all confidence intervals in output:
95.00

----- END MATRIX -----

restore.

Frequencies

Notes

| | | |
|------------------------|---|---|
| Output Created | 20-OCT-2020 16:08:19 | |
| Comments | | |
| Input | Active Dataset | DataSet1 |
| | Filter | <none> |
| | Weight | <none> |
| | Split File | <none> |
| | N of Rows in Working Data File | 856 |
| Missing Value Handling | Definition of Missing | User-defined missing values are treated as missing. |
| | Cases Used | Statistics are based on all cases with valid data. |
| Syntax | FREQUENCIES VARIABLES=Respont Gend MS Age Edu Exp Incm /ORDER=ANALYSIS. | |
| Resources | Processor Time | 00:00:00.00 |
| | Elapsed Time | 00:00:00.00 |

Statistics

| | | Respondents | Gender | Marital Status | Age | Education | Experience | |
|---|---------|-------------|--------|----------------|-----|-----------|------------|--|
| N | Valid | 0 | 856 | 856 | 856 | 856 | 856 | |
| | Missing | 856 | 0 | 0 | 0 | 0 | 0 | |

Frequency Table

Respondents

| | | Frequency | Percent |
|---------|--------|-----------|---------|
| Missing | System | 856 | 100.0 |

Gender

| | | Frequency | Percent | Valid Percent | Cumulative Percent |
|-------|------|-----------|---------|---------------|--------------------|
| Valid | Male | 468 | 54.7 | 54.7 | 54.7 |

| | | | | |
|--------|-----|-------|-------|-------|
| Female | 388 | 45.3 | 45.3 | 100.0 |
| Total | 856 | 100.0 | 100.0 | |

Marital Status

| | Frequency | Percent | Valid Percent | Cumulative Percent |
|--------------|-----------|---------|---------------|--------------------|
| Valid Single | 505 | 59.0 | 59.0 | 59.0 |
| Married | 323 | 37.7 | 37.7 | 96.7 |
| Divorced | 28 | 3.3 | 3.3 | 100.0 |
| Total | 856 | 100.0 | 100.0 | |

Age

| | Frequency | Percent | Valid Percent | Cumulative Percent |
|--------------|-----------|---------|---------------|--------------------|
| Valid 18-30 | 297 | 34.7 | 34.7 | 34.7 |
| 30-40 | 180 | 21.0 | 21.0 | 55.7 |
| 40-50 | 112 | 13.1 | 13.1 | 68.8 |
| 50-60 | 144 | 16.8 | 16.8 | 85.6 |
| More than 60 | 123 | 14.4 | 14.4 | 100.0 |
| Total | 856 | 100.0 | 100.0 | |

Education

| | Frequency | Percent | Valid Percent | Cumulative Percent |
|---------------------|-----------|---------|---------------|--------------------|
| Valid High School | 379 | 44.3 | 44.3 | 44.3 |
| Graduation | 278 | 32.5 | 32.5 | 76.8 |
| Post Graduation | 130 | 15.2 | 15.2 | 91.9 |
| Professional degree | 69 | 8.1 | 8.1 | 100.0 |
| Total | 856 | 100.0 | 100.0 | |

Experience

| | Frequency | Percent | Valid Percent | Cumulative Percent |
|-----------|-----------|---------|---------------|--------------------|
| Valid 1-5 | 225 | 26.3 | 26.3 | 26.3 |
| 5-10 | 252 | 29.4 | 29.4 | 55.7 |

| | | | | |
|--------------|-----|-------|-------|-------|
| 10-15 | 107 | 12.5 | 12.5 | 68.2 |
| 15-20 | 119 | 13.9 | 13.9 | 82.1 |
| More than 20 | 153 | 17.9 | 17.9 | 100.0 |
| Total | 856 | 100.0 | 100.0 | |

| Income | | | | | |
|--------|--------------|-----------|---------|---------------|--------------------|
| | | Frequency | Percent | Valid Percent | Cumulative Percent |
| Valid | 10-30 | 151 | 17.6 | 17.6 | 17.6 |
| | 30-50 | 369 | 43.1 | 43.1 | 60.7 |
| | 50-70 | 175 | 20.4 | 20.4 | 81.2 |
| | 70-90 | 103 | 12.0 | 12.0 | 93.2 |
| | More than 90 | 58 | 6.8 | 6.8 | 100.0 |
| | Total | 856 | 100.0 | 100.0 | |

Descriptives

| Notes | | | |
|------------------------|--------------------------------|---|--|
| Output Created | | 20-OCT-2020 16:09:41 | |
| Comments | | | |
| Input | Active Dataset | DataSet1 | |
| | Filter | <none> | |
| | Weight | <none> | |
| | Split File | <none> | |
| | N of Rows in Working Data File | 856 | |
| Missing Value Handling | Definition of Missing | User defined missing values are treated as missing. | |
| | Cases Used | All non-missing data are used. | |

| | | |
|-----------|----------------|--|
| Syntax | | DESCRIPTIVES VARIABLES=Operational_Performance Financial_Performance Pro_Env_Strgy Plan_Orga_Pract Opra_Pract Comm_Pract Eco_Innovation Technological_Advan /STATISTICS=MEAN STDDEV KURTOSIS SKEWNESS. |
| Resources | Processor Time | 00:00:00.00 |
| | Elapsed Time | 00:00:00.01 |

Descriptive Statistics

| | N | Mean | Std. Deviation | Skewness | | Kurtosis |
|-------------------------------------|-----------|-----------|----------------|-----------|------------|-----------|
| | Statistic | Statistic | Statistic | Statistic | Std. Error | Statistic |
| Operational Performance | 856 | 3.7570 | 1.09409 | -.845 | .084 | .142 |
| Financial Performance | 856 | 3.7640 | 1.09988 | -.797 | .084 | .070 |
| Proactive Environmental Strategy | 856 | 3.8879 | 1.11933 | -.896 | .084 | .180 |
| Planning & Organizational Practices | 856 | 3.8575 | 1.00909 | -.951 | .084 | 1.005 |
| Operational Practices | 856 | 3.9019 | 1.12689 | -.902 | .084 | .153 |
| Communicational Practices | 856 | 3.8107 | 1.07352 | -.886 | .084 | .384 |
| Eco-Innovation | 856 | 3.7582 | 1.07330 | -.776 | .084 | .208 |
| Technological Advances | 856 | 3.8107 | 1.07352 | -.886 | .084 | .384 |
| Valid N (listwise) | 856 | | | | | |

Reliability

Notes

| | | |
|----------------|----------------|----------------------|
| Output Created | | 20-OCT-2020 16:11:37 |
| Comments | | |
| Input | Active Dataset | DataSet1 |
| | Filter | <none> |
| | Weight | <none> |
| | Split File | <none> |

| | | |
|------------------------|--------------------------------|--|
| | N of Rows in Working Data File | 856 |
| | Matrix Input | |
| Missing Value Handling | Definition of Missing | User-defined missing values are treated as missing. |
| | Cases Used | Statistics are based on all cases with valid data for all variables in the procedure. |
| Syntax | | RELIABILITY /VARIABLES=Operational_Performance Financial_Performance Pro_Env_Strgy Plan_Orga_Pract Opra_Pract Comm_Pract Eco_Innovation Technological_Advan /SCALE('ALL VARIABLES') ALL /MODEL=ALPHA /STATISTICS=SCALE /SUMMARY=TOTAL. |
| Resources | Processor Time | 00:00:00.02 |
| | Elapsed Time | 00:00:00.01 |

Scale: ALL VARIABLES

Case Processing Summary

| | N | % |
|-----------------------|-----|-------|
| Cases Valid | 856 | 100.0 |
| Excluded ^a | 0 | .0 |
| Total | 856 | 100.0 |

a. Listwise deletion based on all variables in the procedure.

Reliability Statistics

| Cronbach's Alpha | N of Items |
|------------------|------------|
| .976 | 8 |

Item-Total Statistics

| | Scale Mean if Item Deleted | Scale Variance if Item Deleted | Corrected Item-Total Correlation | Cronbach's Alpha if Item Deleted |
|-------------------------------------|----------------------------|--------------------------------|----------------------------------|----------------------------------|
| Operational Performance | 26.7909 | 49.665 | .891 | .974 |
| Financial Performance | 26.7839 | 49.243 | .916 | .973 |
| Proactive Environmental Strategy | 26.6600 | 49.619 | .870 | .975 |
| Planning & Organizational Practices | 26.6904 | 50.904 | .880 | .974 |
| Operational Practices | 26.6460 | 48.928 | .913 | .973 |
| Communicational Practices | 26.7371 | 49.527 | .921 | .972 |
| Eco-Innovation | 26.7897 | 49.808 | .900 | .973 |
| Technological Advances | 26.7371 | 49.289 | .939 | .971 |

Scale Statistics

| Mean | Variance | Std. Deviation | N of Items |
|---------|----------|----------------|------------|
| 30.5479 | 64.597 | 8.03720 | 8 |

Factor Analysis

Notes

| | | |
|------------------------|--------------------------------|---|
| Output Created | 20-OCT-2020 16:13:34 | |
| Comments | | |
| Input | Active Dataset | DataSet1 |
| | Filter | <none> |
| | Weight | <none> |
| | Split File | <none> |
| | N of Rows in Working Data File | 856 |
| Missing Value Handling | Definition of Missing | MISSING=EXCLUDE: User-defined missing values are treated as missing. |
| | Cases Used | LISTWISE: Statistics are based on cases with no missing values for any variable used. |

| | | |
|-----------|-------------------------|--|
| Syntax | | <p> FACTOR /VARIABLES OP1 OP2 OP3 FP1 FP2 FP3 PES1 PES2 PES3 POP1 POP2 POP3 OPR1 OPR2 OPR3 CP1 CP2 CP3 EIN1 EIN2 EIN3 TEA1 TEA2 TEA3 /MISSING LISTWISE /ANALYSIS OP1 OP2 OP3 FP1 FP2 FP3 PES1 PES2 PES3 POP1 POP2 POP3 OPR1 OPR2 OPR3 CP1 CP2 CP3 EIN1 EIN2 EIN3 TEA1 TEA2 TEA3 /PRINT INITIAL CORRELATION SIG DET KMO EXTRACTION ROTATION /FORMAT BLANK(.5) /CRITERIA MINEIGEN(1) ITERATE(25) /EXTRACTION PC /CRITERIA ITERATE(25) /ROTATION VARIMAX /METHOD=CORRELATION. </p> |
| Resources | Processor Time | 00:00:00.05 |
| | Elapsed Time | 00:00:00.05 |
| | Maximum Memory Required | 68472 (66.867K) bytes |

| | | OP1 | OP2 | OP3 | FP1 | FP2 | FP3 | PES |
|-------------|------|-------|-------|-------|-------|-------|-------|-----|
| Correlation | OP1 | 1.000 | .725 | .861 | -.007 | -.053 | -.023 | |
| | OP2 | .725 | 1.000 | .729 | .011 | -.048 | .021 | |
| | OP3 | .861 | .729 | 1.000 | -.005 | -.068 | -.025 | |
| | FP1 | -.007 | .011 | -.005 | 1.000 | .723 | .861 | |
| | FP2 | -.053 | -.048 | -.068 | .723 | 1.000 | .785 | |
| | FP3 | -.023 | .021 | -.025 | .861 | .785 | 1.000 | |
| | PES1 | -.040 | -.057 | -.094 | -.012 | -.014 | -.054 | |
| | PES2 | -.019 | -.023 | -.055 | -.044 | .010 | -.012 | |
| | PES3 | -.042 | -.065 | -.096 | -.062 | -.004 | -.030 | |
| | POP1 | .050 | .017 | .076 | .081 | .008 | .030 | |
| | POP2 | .032 | .030 | .068 | .046 | -.001 | .050 | |
| | POP3 | .021 | .006 | .058 | .036 | .003 | .037 | |
| | OPR1 | .057 | .100 | .079 | -.080 | -.128 | -.101 | |
| | OPR2 | .067 | .131 | .109 | -.062 | -.133 | -.092 | |
| | OPR3 | -.007 | .004 | -.009 | -.047 | -.042 | -.050 | |
| | CP1 | .003 | -.032 | .008 | .056 | .072 | .023 | |
| | CP2 | .013 | .001 | -.007 | .050 | .080 | .041 | |
| | CP3 | -.001 | -.009 | .002 | .073 | .078 | .064 | |

| | | | | | | | |
|-----------------|------|-------|-------|-------|-------|-------|-------|
| | EIN1 | -.048 | -.014 | -.046 | -.069 | -.041 | -.047 |
| | EIN2 | -.034 | .014 | -.033 | -.047 | -.043 | -.010 |
| | EIN3 | -.068 | -.056 | -.055 | -.019 | -.074 | -.047 |
| | TEA1 | .121 | .038 | .083 | .018 | -.019 | -.017 |
| | TEA2 | .079 | .011 | .048 | .035 | .024 | .013 |
| | TEA3 | .069 | .029 | .033 | -.009 | .014 | -.022 |
| Sig. (1-tailed) | OP1 | | .000 | .000 | .418 | .059 | .247 |
| | OP2 | .000 | | .000 | .378 | .079 | .270 |
| | OP3 | .000 | .000 | | .438 | .023 | .231 |
| | FP1 | .418 | .378 | .438 | | .000 | .000 |
| | FP2 | .059 | .079 | .023 | .000 | | .000 |
| | FP3 | .247 | .270 | .231 | .000 | .000 | |
| | PES1 | .123 | .048 | .003 | .367 | .337 | .057 |
| | PES2 | .294 | .250 | .054 | .097 | .386 | .361 |
| | PES3 | .108 | .028 | .002 | .036 | .454 | .190 |
| | POP1 | .070 | .312 | .014 | .009 | .408 | .193 |
| | POP2 | .174 | .190 | .023 | .092 | .484 | .071 |
| | POP3 | .267 | .433 | .046 | .147 | .468 | .138 |
| | OPR1 | .047 | .002 | .011 | .009 | .000 | .002 |
| | OPR2 | .026 | .000 | .001 | .036 | .000 | .003 |
| | OPR3 | .420 | .454 | .398 | .083 | .111 | .071 |
| | CP1 | .468 | .178 | .406 | .051 | .018 | .248 |
| | CP2 | .357 | .489 | .420 | .074 | .009 | .114 |
| | CP3 | .489 | .394 | .480 | .016 | .011 | .030 |
| | EIN1 | .080 | .345 | .091 | .021 | .115 | .086 |
| | EIN2 | .158 | .340 | .167 | .086 | .104 | .385 |
| | EIN3 | .023 | .049 | .054 | .285 | .015 | .084 |
| | TEA1 | .000 | .136 | .007 | .295 | .284 | .305 |
| | TEA2 | .010 | .375 | .081 | .154 | .240 | .357 |
| | TEA3 | .021 | .201 | .166 | .393 | .345 | .262 |

KMO and Bartlett's Test

| | | |
|--|--------------------|-----------|
| Kaiser-Meyer-Olkin Measure of Sampling Adequacy. | | .705 |
| Bartlett's Test of Sphericity | Approx. Chi-Square | 15406.272 |
| | Df | 276 |
| | Sig. | .000 |

Communalities

| | Initial | Extraction |
|-----|---------|------------|
| OP1 | 1.000 | .881 |
| OP2 | 1.000 | .787 |
| OP3 | 1.000 | .884 |
| FP1 | 1.000 | .866 |
| FP2 | 1.000 | .812 |

| | | |
|------|-------|------|
| FP3 | 1.000 | .910 |
| PES1 | 1.000 | .869 |
| PES2 | 1.000 | .807 |
| PES3 | 1.000 | .908 |
| POP1 | 1.000 | .873 |
| POP2 | 1.000 | .824 |
| POP3 | 1.000 | .911 |
| OPR1 | 1.000 | .865 |
| OPR2 | 1.000 | .828 |
| OPR3 | 1.000 | .778 |
| CP1 | 1.000 | .878 |
| CP2 | 1.000 | .823 |
| CP3 | 1.000 | .913 |
| EIN1 | 1.000 | .849 |
| EIN2 | 1.000 | .800 |
| EIN3 | 1.000 | .756 |
| TEA1 | 1.000 | .884 |
| TEA2 | 1.000 | .850 |
| TEA3 | 1.000 | .854 |

Extraction Method: Principal Component Analysis.

| Total Variance Explained | | | | | | |
|--------------------------|---------------------|---------------|--------------|-------------------------------------|---------------|--------------|
| Component | Initial Eigenvalues | | | Extraction Sums of Squared Loadings | | |
| | Total | % of Variance | Cumulative % | Total | % of Variance | Cumulative % |
| 1 | 3.195 | 13.313 | 13.313 | 3.195 | 13.313 | 13.313 |
| 2 | 3.048 | 12.699 | 26.013 | 3.048 | 12.699 | 26.013 |
| 3 | 2.727 | 11.362 | 37.375 | 2.727 | 11.362 | 37.375 |
| 4 | 2.574 | 10.724 | 48.098 | 2.574 | 10.724 | 48.098 |
| 5 | 2.526 | 10.525 | 58.623 | 2.526 | 10.525 | 58.623 |
| 6 | 2.245 | 9.355 | 67.978 | 2.245 | 9.355 | 67.978 |
| 7 | 2.234 | 9.310 | 77.288 | 2.234 | 9.310 | 77.288 |
| 8 | 1.861 | 7.752 | 85.041 | 1.861 | 7.752 | 85.041 |
| 9 | .452 | 1.882 | 86.922 | | | |
| 10 | .375 | 1.561 | 88.483 | | | |
| 11 | .330 | 1.374 | 89.858 | | | |
| 12 | .302 | 1.259 | 91.117 | | | |
| 13 | .278 | 1.159 | 92.276 | | | |

| | | | | | | |
|----|------|-------|---------|--|--|--|
| 14 | .254 | 1.059 | 93.335 | | | |
| 15 | .243 | 1.011 | 94.346 | | | |
| 16 | .218 | .907 | 95.253 | | | |
| 17 | .208 | .866 | 96.118 | | | |
| 18 | .186 | .777 | 96.896 | | | |
| 19 | .164 | .682 | 97.578 | | | |
| 20 | .153 | .639 | 98.217 | | | |
| 21 | .132 | .548 | 98.765 | | | |
| 22 | .109 | .453 | 99.218 | | | |
| 23 | .100 | .418 | 99.635 | | | |
| 24 | .088 | .365 | 100.000 | | | |

Extraction Method: Principal Component Analysis.

Component Matrix^a

| | Component | | | | | | | |
|------|-----------|------|-------|------|------|------|------|-------|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| OP1 | | .547 | | | | | .545 | |
| OP2 | | | | | | | .551 | |
| OP3 | | .562 | | | | | .530 | |
| FP1 | | | -.505 | | | .524 | | |
| FP2 | | | | | | .507 | | |
| FP3 | | | -.524 | | | .540 | | |
| PES1 | | | .539 | .547 | | | | |
| PES2 | | | | .526 | | | | |
| PES3 | | | .551 | .545 | | | | |
| POP1 | | | | .519 | | | | |
| POP2 | | | | | | | | |
| POP3 | | | | .532 | | | | |
| OPR1 | -.546 | | | | | | | -.541 |
| OPR2 | -.533 | | | | | | | |
| OPR3 | | | | | | | | |
| CP1 | .581 | | | | .509 | | | |
| CP2 | .574 | | | | | | | |
| CP3 | .598 | | | | .512 | | | |
| EIN1 | | | | | .593 | | | |
| EIN2 | | | | | .575 | | | |
| EIN3 | | | | | .666 | | | |
| TEA1 | | .574 | | | | | | |

| | | | | | | | | |
|------|--|------|--|--|--|--|--|--|
| TEA2 | | .561 | | | | | | |
| TEA3 | | .526 | | | | | | |

Extraction Method: Principal Component Analysis.

a. 8 components extracted.

Rotated Component Matrix^a

| | Component | | | | | | | |
|------|-----------|------|------|------|------|------|------|------|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| OP1 | | | | | | .935 | | |
| OP2 | | | | | | .885 | | |
| OP3 | | | | | | .935 | | |
| FP1 | | | | .928 | | | | |
| FP2 | | | | .895 | | | | |
| FP3 | | | | .952 | | | | |
| PES1 | | | | | .929 | | | |
| PES2 | | | | | .891 | | | |
| PES3 | | | | | .949 | | | |
| POP1 | | .928 | | | | | | |
| POP2 | | .905 | | | | | | |
| POP3 | | .952 | | | | | | |
| OPR1 | | | | | | | .923 | |
| OPR2 | | | | | | | .897 | |
| OPR3 | | | | | | | .872 | |
| CP1 | .929 | | | | | | | |
| CP2 | .899 | | | | | | | |
| CP3 | .949 | | | | | | | |
| EIN1 | | | | | | | | .916 |
| EIN2 | | | | | | | | .887 |
| EIN3 | | | | | | | | .859 |
| TEA1 | | | .933 | | | | | |
| TEA2 | | | .914 | | | | | |
| TEA3 | | | .921 | | | | | |

Extraction Method: Principal Component Analysis.

Rotation Method: Varimax with Kaiser Normalization.

a. Rotation converged in 5 iterations.

Component Transformation Matrix

| Component | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|-----------|-------|-------|-------|-------|-------|-------|-------|---|
| 1 | .570 | -.159 | .281 | .434 | -.040 | -.092 | -.506 | |
| 2 | .192 | .428 | .551 | -.025 | -.245 | .529 | .356 | |
| 3 | .137 | -.458 | .450 | -.517 | .528 | -.026 | .123 | |
| 4 | .087 | .556 | .122 | .286 | .583 | -.441 | .200 | |
| 5 | .530 | -.083 | .043 | -.155 | -.343 | -.315 | .149 | |
| 6 | -.337 | -.474 | .332 | .606 | -.079 | -.071 | .379 | |
| 7 | .287 | -.136 | -.339 | .262 | .437 | .631 | .013 | |
| 8 | -.367 | .149 | .415 | -.040 | .063 | .123 | -.629 | |

Extraction Method: Principal Component Analysis.

Rotation Method: Varimax with Kaiser Normalization.