sensors

MDPI

Article

Kernel Probabilistic K-Means Clustering

Bowen Liu [1], Ting Zhang [1], Yujian Li [2,*], Zhaoying Liu [1] and Zhilin Zhang [1]

[1] Faculty of Information Technology, Beijing University of Technology, Beijing 100124, China; liubw2017@emails.bjut.edu.cn (B.L.); zhangting@bjut.edu.cn (T.Z.); zhaoying.liu@bjut.edu.cn (Z.L.); zhangzl@emails.bjut.edu.cn (Z.Z.)

[2] School of Artificial Intelligence, Guilin University of Electronic Technology, Guilin 541004, China

* Correspondence: liyujian@guet.edu.cn

Abstract: Kernel fuzzy c-means (KFCM) is a significantly improved version of fuzzy c-means (FCM) for processing linearly inseparable datasets. However, for fuzzification parameter $m = 1$, the problem of KFCM (kernel fuzzy c-means) cannot be solved by Lagrangian optimization. To solve this problem, an equivalent model, called kernel probabilistic k-means (KPKM), is proposed here. The novel model relates KFCM to kernel k-means (KKM) in a unified mathematic framework. Moreover, the proposed KPKM can be addressed by the active gradient projection (AGP) method, which is a nonlinear programming technique with constraints of linear equalities and linear inequalities. To accelerate the AGP method, a fast AGP (FAGP) algorithm was designed. The proposed FAGP uses a maximum-step strategy to estimate the step length, and uses an iterative method to update the projection matrix. Experiments demonstrated the effectiveness of the proposed method through a performance comparison of KPKM with KFCM, KKM, FCM and k-means. Experiments showed that the proposed KPKM is able to find nonlinearly separable structures in synthetic datasets. Ten real UCI datasets were used in this study, and KPKM had better clustering performance on at least six datsets. The proposed fast AGP requires less running time than the original AGP, and it reduced running time by 76–95% on real datasets.

Keywords: fuzzy c-means; kernel probabilistic k-means; nonlinear programming; fast active gradient projection

check for updates

Citation: Liu, B.; Zhang, T.; Li, Y.; Liu, Z.; Zhang, Z. Kernel Probabilistic K-Means Clustering. *Sensors* **2021**, *21*, 1892. https://doi.org/10.3390/s21051892

Academic Editor: Richard J. Povinelli

Received: 3 February 2021
Accepted: 5 March 2021
Published: 8 March 2021

1. Introduction

Clustering is an important unsupervised method, and the purpose of clustering is to divide a dataset into multiple clusters (or classes) with high intra-cluster similarity and low inter-cluster similarity. There have been many clustering algorithms, such as k-means (KM) and its variants [1–16]. Others are based on minimal spanning trees [17–19], density analysis [20–25], spectral analysis [26,27], subspace clustering [28,29], etc.

Generally, k-means minimizes the sum of squared Euclidean distances between each sample point and its nearest clustering center [1]. One variant of k-means is kernel k-means (KKM) [30–33], which is able to find nonlinearly separable structures by using the kernel function method. Another variant of k-means is fuzzy c-means (FCM) [2], which determines partitions by computing the membership degree of each data point to each cluster. The higher the membership degree, the greater the possibility of the data point belonging to the cluster. Although FCM is more flexible in applications [11–16], it is primarily suitable for linearly separable datasets. Kernel fuzzy c-means (KFCM) [34] is a significantly improved version of fuzzy c-means for clustering linearly inseparable datasets. However, the problem of KFCM with fuzzification parameter $m = 1$ cannot be solved by existing methods.

To solve the special case of KFCM for $m = 1$, a novel model called kernel probabilistic k-means (KPKM) is proposed. In fact, KPKM is a nonlinear programming model constrained on linear equalities and linear inequalities, and it is equivalent to KFCM at

m = 1. In theory, the proposed KPKM can be solved by the active gradient projection (AGP) method [35,36]. Since the AGP method may take too much time on large datasets, we further propose a fast AGP (FAGP) algorithm to solve KPKM more efficiently. Moreover, we report experiments demonstrating its effectiveness compared with KFCM, KKM, FCM, and KM.

The paper is organized as follows: Section 2 reviews previous work. The KPKM algorithm is proposed in Section 3. Section 4 proposes a solution for KPKM. Section 5 presents descriptions and analyses of experiments. Conclusions and future work are mentioned in Section 6.

2. Background and Related work

There has been a lot of work related to this paper, mainly including k-means, fuzzy c-means, kernel k-means and kernel fuzzy c-means.

K-means minimizes the sum of squared Euclidean distances between each sample point and its nearest cluster center. K-means first chooses initial clustering centers randomly or manually, and then partitions a dataset into several clusters (a data point belongs to the cluster whose clustering center is nearest to the data point), and computes the mean of a cluster as the clustering center. K-means repeatedly updates clustering centers and clusters until convergence. FCM has the same ideal as k-means. FCM introduces a membership degree w_{ij} and a fuzzy coefficient m into the objective function. The higher w_{ij} is, the greater possibility of the i-th data point belonging to the j-th cluster. K-means and FCM belong to partition-based clustering algorithms, and partition-based clustering algorithms usually are not able to cluster linearly inseparable datasets. Kernel method maps a linearly inseparable dataset into a linearly separable space, so kernel k-means (and FCM) using a kernel function can cluster linearly inseparable datasets.

2.1. K-Means and Fuzzy C-Means

Let $\boldsymbol{X} = \left\{\boldsymbol{x}_i | \boldsymbol{x}_i \in \mathbb{R}^D, 1 \le i \le L\right\}$ represent a dataset. K-means divides it into k clusters. If ω_j denotes the j-th cluster, we have $\boldsymbol{X} = \bigcup\limits_{j=1}^{K} \omega_j$ and $\forall 1 \le i \ne j \le K, \omega_i \bigcap \omega_j = \emptyset$. Using $L_j = |\omega_j|$ to stand for the number of elements in ω_j with the center of $\boldsymbol{c}_j$, k-means can be described as minimizing the following objective function:

$$J = \sum_{i=1}^{K} \sum_{\boldsymbol{x}_i \in \omega_j} \left\|\boldsymbol{x}_i - \boldsymbol{c}_j\right\|^2 \tag{1}$$

where

$$\boldsymbol{c}_j = \frac{1}{L_j} \sum_{\boldsymbol{x}_i \in \omega_j} \boldsymbol{x}_i. \tag{2}$$

Let $L = \sum\limits_{j=1}^{K} L_j$. By using c instead of k and assigning membership degree w_{ij} to the i-th data point in the j-th cluster for $1 \le i \le L$ and $1 \le j \le C$, the fuzzy c-means clustering model can be formulated as follows:

$$\begin{aligned} &\min\ J = \sum_{j=1}^{C} \sum_{i=1}^{L} w_{ij}^{m} \left\|\boldsymbol{x}_i - \boldsymbol{c}_j\right\|^2 \\ &s.t. \sum_{j=1}^{C} w_{ij} = 1, w_{ij} \ge 0 \end{aligned} \tag{3}$$

where $m > 1$, w_{ij} and $\boldsymbol{c}_j$ are computed alternately below [2].

$$\begin{cases} w_{ij} = \dfrac{\|\boldsymbol{x}_i - \boldsymbol{c}_j\|^{-\frac{2}{m-1}}}{\sum\limits_{k=1}^{C} \|\boldsymbol{x}_i - \boldsymbol{c}_k\|^{-\frac{2}{m-1}}} \\ \boldsymbol{c}_j = \dfrac{\sum\limits_{i=1}^{L} w_{ij}^m \boldsymbol{x}_i}{\sum\limits_{i=1}^{L} w_{ij}^m} \end{cases}. \tag{4}$$

2.2. Kernel K-Means and Kernel Fuzzy C-Means

To improve performance of k-means and fuzzy c-means in linearly inseparable datasets, we may develop their kernel versions by choosing a feature mapping $\varphi(\cdot) : \mathcal{R}^D \to \mathcal{H}$ from data points to kernel Hilbert space [37]. Though φ is usually unknown, it must satisfy

$$K(\boldsymbol{x}, \boldsymbol{y}) = (\varphi(\boldsymbol{x}))^{\mathrm{T}} \varphi(\boldsymbol{y}) \tag{5}$$

where $K(\boldsymbol{x}, \boldsymbol{y})$ is a kernel function. Commonly used kernel functions are presented in Table 1.

Table 1. Commonly used kernel functions.

Name	Code
Linear kernel	$K(\boldsymbol{x}, \boldsymbol{y}) = \langle \boldsymbol{x}, \boldsymbol{y} \rangle$
Laplace Radial Basis Function kernel	$K_{\text{lap}}(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\sigma \|\boldsymbol{x} - \boldsymbol{y}\|)$
Gaussian Radial Basis Function kernel	$K_{\text{gau}}(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{y}\|^2}{2\sigma^2}\right)$
Polynomial kernel	$K_{\text{pol}}(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x} \cdot \boldsymbol{y} + \beta)^{\alpha}$
Sigmoid kernel	$K_{\tan}(\boldsymbol{x}, \boldsymbol{y}) = \tanh(\alpha \boldsymbol{x} \cdot \boldsymbol{y} + \beta)$

In Table 1, $\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \boldsymbol{x} \cdot \boldsymbol{y}$ denotes the inner product of $\boldsymbol{x}$ and $\boldsymbol{y}$, and σ, α, β are parameters of the kernel. The objective function of kernel k-means is defined as

$$J_k = \sum_{j=1}^{K} \sum_{\boldsymbol{x}_i \in \omega_j} \|\varphi(\boldsymbol{x}_i) - \boldsymbol{c}_j\|^2 \tag{6}$$

where

$$\boldsymbol{c}_j = \frac{1}{L_j} \sum_{\boldsymbol{x}_i \in \omega_j} \varphi(\boldsymbol{x}_i). \tag{7}$$

Using (5) and (7), we have

$$\|\varphi(\boldsymbol{x}_i) - \boldsymbol{c}_j\|^2 = K(\boldsymbol{x}_i, \boldsymbol{x}_i) - \frac{2}{L_j} \sum_{\boldsymbol{x}_k \in \omega_j}^{L} K(\boldsymbol{x}_k, \boldsymbol{x}_i) + \frac{1}{(L_j)^2} \sum_{\boldsymbol{x}_l \in \omega_j}^{L} \sum_{\boldsymbol{x}_h \in \omega_j}^{L} K(\boldsymbol{x}_l, \boldsymbol{x}_h). \tag{8}$$

Let w_{ij} represent the membership degree of the i-th point belonging to the j-th class. Likewise, we can define the kernel FCM clustering model as follows.

$$\begin{aligned} &\min\ J_f(\boldsymbol{W}) = \sum_{j=1}^{C} \sum_{i=1}^{L} w_{ij}^m \|\varphi(\boldsymbol{x}_i) - \boldsymbol{c}_j\|^2 \\ &s.t. \sum_{j=1}^{C} w_{ij} = 1, w_{ij} \geq 0, m > 1 \end{aligned} \tag{9}$$

where

$$\boldsymbol{c}_j = \frac{\sum\limits_{l=1}^{L} w_{lj}^m \varphi(\boldsymbol{x}_l)}{\sum\limits_{l=1}^{L} w_{lj}^m}, \tag{10}$$

$$\left\|\varphi(\boldsymbol{x}_i) - \boldsymbol{c}_j\right\|^2 = K(\boldsymbol{x}_i, \boldsymbol{x}_i) - \frac{2}{\sum\limits_{l=1}^{L} w_{lj}^m} \sum_{k=1}^{L} w_{kj}^m K(\boldsymbol{x}_k, \boldsymbol{x}_i) + \frac{1}{\left(\sum\limits_{l=1}^{L} w_{lj}^m\right)^2} \sum_{l=1}^{L} \sum_{h=1}^{L} w_{lj}^m w_{hj}^m K(\boldsymbol{x}_l, \boldsymbol{x}_h). \tag{11}$$

The membership degree is computed via

$$w_{ij} = \frac{\left(\|\varphi(\boldsymbol{x}_i) - \boldsymbol{c}_j\|^2\right)^{-\frac{1}{m-1}}}{\sum\limits_{k=1}^{C} \left(\|\varphi(\boldsymbol{x}_i) - \boldsymbol{c}_k\|^2\right)^{-\frac{1}{m-1}}}. \tag{12}$$

3. Kernel Probabilistic K-Means

In this section, the kernel probabilistic k-means (KPKM) are proposed.

We first review the problem. When $m = 1$, the KFCM model gets into a special case, namely,

$$\begin{aligned} &\min\ J_f(\boldsymbol{W}) = \sum_{j=1}^{K} \sum_{i=1}^{L} w_{ij} \|\varphi(\boldsymbol{x}_i) - \boldsymbol{c}_j\|^2 \\ &s.t. \sum_{j=1}^{K} w_{ij} = 1, w_{ij} \geq 0 \end{aligned} \tag{13}$$

where

$$\boldsymbol{c}_j = \frac{\sum\limits_{l=1}^{L} w_{lj} \varphi(\boldsymbol{x}_l)}{\sum\limits_{l=1}^{L} w_{lj}}, \tag{14}$$

$$\left\|\varphi(\boldsymbol{x}_i) - \boldsymbol{c}_j\right\|^2 = K(\boldsymbol{x}_i, \boldsymbol{x}_i) - \frac{2}{\sum\limits_{l=1}^{L} w_{lj}} \sum_{k=1}^{L} w_{kj} K(\boldsymbol{x}_k, \boldsymbol{x}_i) + \frac{1}{\left(\sum\limits_{l=1}^{L} w_{lj}\right)^2} \sum_{l=1}^{L} \sum_{h=1}^{L} w_{lj} w_{hj} K(\boldsymbol{x}_l, \boldsymbol{x}_h). \tag{15}$$

This special case cannot be solved by Lagrangian optimization for $m > 1$, because (12) cannot be computed when $m = 1$ (when $m = 1$, $\frac{1}{m-1} = \frac{1}{0}$ cannot be computed).

In this paper, we use the optimization methods to solve this problem, but the partial derivative of (13) with respect to w_{ij} is $\|\varphi(\boldsymbol{x}_i) - \boldsymbol{c}_j\|^2$, and it does not contain w_{ij}.

In order to solve the problem, we introduce (14) into (13), and redefine the member degrees w_{ij} as probability p_{ij} for $1 \leq i \leq L$ and $1 \leq j \leq K$. Finally, we have

$$\begin{aligned} J(\boldsymbol{P}) = \sum_{j=1}^{K} \sum_{i=1}^{L} p_{ij} \left\| \varphi(\boldsymbol{x}_i) - \frac{\sum_{l=1}^{L} p_{lj} \varphi(\boldsymbol{x}_l)}{\sum_{l=1}^{L} p_{lj}} \right\|^2 \\ s.t. \sum_{j=1}^{K} p_{ij} = 1, p_{ij} \geq 0 \end{aligned} \tag{16}$$

where probability vector

$$\boldsymbol{P} = \{p_{11}, \ldots, p_{1K}, p_{21}, \ldots p_{2K}, \ldots, p_{LK}\}^{\mathrm{T}}.$$

(16) is the proposed kernel probabilistic k-means.

The proposed KPKM is a soft clustering method. The $p_{ij}(0 \leq p_{ij} \leq 1)$ is the probability of the *i*-th data point belonging to the *j*-th cluster. The higher p_{ij} is, the greater possibility of the *i*-th data point belonging to the *j*-th cluster. In KKM, the membership degree has only two values (0 and 1). In the proposed KPKM, $w_{ij} \in [0,1]$ (although final $w_{ij} = 0 \texttt{ or } 1$).

4. A Fast Solution to KPKM

KPKM is a nonlinear programming problem with linear equalities and linear inequalities constraints, and it is able to be solved by active gradient projection [35,36] theoretically. In this section, on the basis of AGP, we first calculate the gradient of of the objection function of KPKM, and then design a fast AGP algorithm to solve the KPKM model.

The fast AGP has two advantages: iteratively updating the projection matrix and estimating the maximum step length.

4.1. Gradient Calculation

For convenience, we define F_{kj} as

$$F_{kj} = \left\| \varphi(\boldsymbol{x}_k) - \frac{\sum\limits_{l=1}^{L} p_{lj}\varphi(\boldsymbol{x}_l)}{\sum\limits_{l=1}^{L} p_{lj}} \right\|^2. \tag{17}$$

Using the chain rule on (16), we obtain

$$\frac{\partial J}{\partial p_{ij}} = \sum_{k=1}^{L} p_{kj}\frac{\partial F_{kj}}{\partial p_{ij}} + F_{ij}. \tag{18}$$

According to (17), we can further derive

$$\frac{\partial F_{kj}}{\partial p_{ij}} = \frac{-2}{\sum_{i=1}^{L} p_{ij}}\left(K(\boldsymbol{x}_k,\boldsymbol{x}_i) - \frac{\sum_{l=1}^{L} p_{lj}K(\boldsymbol{x}_k,\boldsymbol{x}_l)}{\sum_{i=1}^{L} p_{ij}} - \frac{\sum_{l=1}^{L} p_{lj}K(\boldsymbol{x}_i,\boldsymbol{x}_l)}{\sum_{i=1}^{L} p_{ij}} + \frac{\sum_{l=1}^{L}\sum_{h=1}^{L} p_{lj}p_{hj}K(\boldsymbol{x}_l,\boldsymbol{x}_h)}{\left(\sum_{i=1}^{L} p_{ij}\right)^2} \right). \tag{19}$$

By substituting (19) into (18), we finally get

$$\frac{\partial J}{\partial p_{ij}} = K(\boldsymbol{x}_i,\boldsymbol{x}_i) + \frac{1}{\left(\sum_{l=1}^{L} p_{lj}\right)^2}\sum_{l=1}^{L}\sum_{h=1}^{L} p_{lj}p_{hj}K(\boldsymbol{x}_l,\boldsymbol{x}_h) - \frac{2}{\sum_{l=1}^{L} p_{lj}}\sum_{k=1}^{L} p_{kj}K(\boldsymbol{x}_k,\boldsymbol{x}_i). \tag{20}$$

and the gradient

$$\nabla J = \begin{bmatrix} \frac{\partial J}{\partial p_{11}} & \cdots & \frac{\partial J}{\partial p_{1K}} & \cdots & \frac{\partial J}{\partial p_{Lj}} & \cdots & \frac{\partial J}{\partial p_{LK}} \end{bmatrix}^{\mathrm{T}}. \tag{21}$$

4.2. Fast AGP

In the constraints of KPKM, there are L linear equalities and $K \times L$ linear inequalities,

$$\forall 1 \leq i \leq L, \sum_{j=1}^{K} p_{ij} = 1, \tag{22}$$

$$\forall 1 \leq i \leq L, 1 \leq j \leq K, p_{ij} \geq 0. \tag{23}$$

Let $\phi = K \times L$. Let $\boldsymbol{I}_{\phi\times\phi}$ be the identity matrix of size $\phi \times \phi$. Define two matrices, inequality matrix A and equality matrix E, where

$$\boldsymbol{A} = \boldsymbol{I}_{\phi\times\phi}, \tag{24}$$

$$\boldsymbol{E}=\begin{bmatrix} \underbrace{1 \ \ldots \ 1}_{K} & 0 \ \ldots \ 0 & 0 \ \ldots \ 0 \\ \underbrace{0 \ \ldots \ 0}_{K} & \underbrace{1 \ \ldots \ 1}_{K} & 0 \ \ldots \ 0 \\ & \ldots & \\ 0 \ \ldots \ 0 & 0 \ \ldots \ 0 & \underbrace{1 \ \ldots \ 1}_{K} \end{bmatrix}_{L\times\phi}. \tag{25}$$

Note that each row of $\boldsymbol{A}$ corresponds to one and only one inequality in (23), and each row of $\boldsymbol{E}$ corresponds to one and only one equality in (22). Accordingly, the KPKM's constraints can be simply expressed as

$$\boldsymbol{AP} \geq \boldsymbol{0}, \boldsymbol{EP} = \boldsymbol{1} \tag{26}$$

where $\boldsymbol{1} = [1, \ldots, 1]^{\mathrm{T}}_{L\times 1}$. Let $\boldsymbol{P}^{(0)}$ stand for a randomly initialized probability vector, and $\boldsymbol{P}^{(k)}$ for the probability vector at iteration k. The rows of inequality matrix $\boldsymbol{A}$ can be broken into two groups: one is active; the other is inactive. The active group is composed of all inequalities that must work exactly as an equality at $\boldsymbol{P}^{(k)}$, whereas the inactive group is composed of the left inequalities. If $\boldsymbol{A}_1^{(k)}$ and $\boldsymbol{A}_2^{(k)}$ respectively denote the active group and the inactive group, we have

$$\boldsymbol{A}_1^{(k)}\boldsymbol{P}^{(k)} = \boldsymbol{0}, \tag{27}$$

$$\boldsymbol{A}_2^{(k)}\boldsymbol{P}^{(k)} > \boldsymbol{0}. \tag{28}$$

At iteration k, the active matrix $\boldsymbol{N}^{(k)}$ is defined as

$$\boldsymbol{N}^{(k)} = \begin{bmatrix} \boldsymbol{A}_1^{(k)} \\ \boldsymbol{E} \end{bmatrix}. \tag{29}$$

When $\boldsymbol{N} = \boldsymbol{N}^{(k)}$ is not a square matrix, we can construct its projection matrix $\boldsymbol{G}^{(k)}$ and the corresponding orthogonal projection matrix $\boldsymbol{Q}^{(k)}$ as follows:

$$\boldsymbol{G}^{(k)} = \boldsymbol{N}^{\mathrm{T}}(\boldsymbol{N}\boldsymbol{N}^{T})^{-1}\boldsymbol{N}, \tag{30}$$

$$\boldsymbol{Q}^{(k)} = \boldsymbol{I} - \boldsymbol{G}^{(k)}. \tag{31}$$

Suppose that $\boldsymbol{n}$ from $\boldsymbol{A}_2^{(k-1)}$ is the active row vector at iteration k, satisfying $\boldsymbol{n}^{\mathrm{T}}\boldsymbol{P}^{(k-1)} > \boldsymbol{0}$ and $\boldsymbol{n}^{\mathrm{T}}\boldsymbol{P}^{(k)} = \boldsymbol{0}$. According to matrix theory [38], we can more efficiently compute $\boldsymbol{G}^{(k)}$ and $\boldsymbol{Q}^{(k)}$ by

$$\boldsymbol{G}^{(k)} = \boldsymbol{G}^{(k-1)} + \boldsymbol{Q}^{(k-1)}\boldsymbol{n}^{\mathrm{T}}\left\langle \boldsymbol{Q}^{(k-1)}\boldsymbol{n}^{\mathrm{T}}, \boldsymbol{Q}^{(k-1)}\boldsymbol{n}^{\mathrm{T}} \right\rangle^{-1}\boldsymbol{n}\boldsymbol{Q}^{(k-1)}. \tag{32}$$

Furthermore, we can compute the projected gradient by

$$\boldsymbol{d}^{(k)} = -\boldsymbol{Q}^{(k)}\nabla J(\boldsymbol{P}^{(k)}). \tag{33}$$

Using $\boldsymbol{d}^{(k)}$, we update the probability vector $\boldsymbol{P}^{(k+1)}$ as

$$\boldsymbol{P}^{(k+1)} = \boldsymbol{P}^{(k)} + t^{(k)}\boldsymbol{d}^{(k)} \tag{34}$$

where $t^{(k)}$ is the step length. Usually, $t^{(k)}$ is chosen as a small number. For fast convergence, we estimate the maximum step length as follows.

$$t^{(k)} = t^{(k)}_{\max}. \tag{35}$$

(1) Let $p_{ij}^{(k+1)} = p_{ij}^{(k)} + t_{ij} d_{ij}^{(k)} = 0$ for $p_{ij}^{(k)} > 0$;

(2) Compute $t_{ij} = -p_{ij}^{(k)} \Big/ d_{ij}^{(k)}$ for $p_{ij}^{(k)} > 0$ and $d_{ij}^{(k)} < 0$;

(3) $t_{\max}^{(k)} = \min\{t_{ij}\}$.

As shown in Figure 1, maximum step length is compared with small step length.

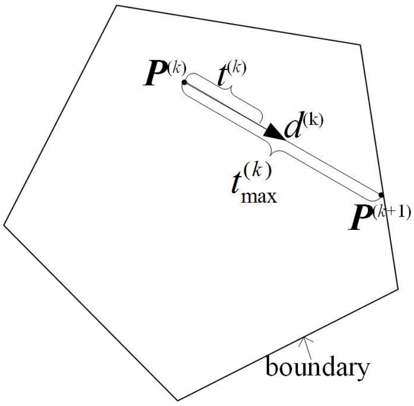


Figure 1. Maximum step length and small step length.

When $\boldsymbol{N} = \boldsymbol{N}^{(k)}$ is a square matrix, it must be invertible. In this case, we actually have $\boldsymbol{G}^{(k)} = \boldsymbol{N}^{\mathrm{T}}(\boldsymbol{N}\boldsymbol{N}^{T})^{-1}\boldsymbol{N} = \boldsymbol{I}_{LK\times LK}$, and $\boldsymbol{Q}^{(k)} = \boldsymbol{0}$. Thus, $\boldsymbol{d}^{(k)} = -\boldsymbol{Q}^{(k)}\nabla J(\boldsymbol{P}^{(k)}) = \boldsymbol{0}$ is not a feasible descent direction. A new descent direction can be computed as follows:

(1) Compute a new vector,

$$\boldsymbol{q}^{(k)} = (\boldsymbol{N}\boldsymbol{N}^{T})^{-1}\boldsymbol{N}\nabla J = \left(\boldsymbol{N}^{\mathrm{T}}\right)^{-1}\nabla J. \tag{36}$$

(2) Break $\boldsymbol{q}^{(k)}$ into two parts $\boldsymbol{q}_1^{(k)}$ and $\boldsymbol{q}_2^{(k)}$, namely,

$$\boldsymbol{q}^{(k)} = \left[\begin{array}{cc} \left(\boldsymbol{q}_1^{(k)}\right)^{\mathrm{T}} & \left(\boldsymbol{q}_2^{(k)}\right)^{\mathrm{T}} \end{array}\right]^{\mathrm{T}} \tag{37}$$

where the size of $\boldsymbol{q}_1^{(k)}$ is the number of rows of $\boldsymbol{A}_1^{(k)}$, and that of $\boldsymbol{q}_2^{(k)}$ is the number of rows of $\boldsymbol{E}$.

(3) If $\boldsymbol{q}_1^{(k)} \geq \boldsymbol{0}$, stop. Otherwise, choose any element from $\boldsymbol{q}_1^{(k)}$ that is less than 0 and delete the corresponding row of $\boldsymbol{A}_1^{(k)}$; then use (29)–(31) and (33) to compute $\boldsymbol{d}^{(k)}$.

The above fast AGP solution to KPKM is outlined in Algorithm 1. Compared with the original AGP, the fast AGP has two advantages: iteratively updating the projection matrix (shown in (32)) and estimating the maximum step length (shown in (35)).

4.3. Analysis of Complexity

In this section, the computational complexities for the traditional AGP algorithm and the proposed FAGP algorithm are analyzed. Let T_A represent the iteration number of AGP. Let T_F represent the iteration number of FAGP. In computations of all matrices, the computational complexity of the projection matrix $\boldsymbol{G}$ is the highest. In the AGP algorithm, computing $\boldsymbol{G}$ via (30) requires $O(K^3L^3)$, so the total computational complexity for AGP algorithm is $O(T_A K^3 L^3)$. In the FAGP algorithm, $\boldsymbol{G}$ is computed via (32), and it does not require to compute the inverse of matrices (i.e., $(\boldsymbol{N}\boldsymbol{N}^{T})^{-1}$), and computing $\boldsymbol{G}$ via (32) requires $O(K^3L^2)$, so the total computational complexity for FAGP algorithm (i.e., Algorithm 1) is $O(T_F K^3 L^2)$.

Algorithm 1 Fast active gradient projection (FAGP).

```
Input: $\boldsymbol{X}$ and $\boldsymbol{K}$
Do:
    (1) Let $k = 0$, $\boldsymbol{N} = \boldsymbol{N}^{(0)} = \boldsymbol{E}$,$\boldsymbol{G}^{(0)} = \boldsymbol{N}^{\mathrm{T}}(\boldsymbol{N}\boldsymbol{N}^{T})^{-1}\boldsymbol{N}$,
initialize $\boldsymbol{P}^{(0)}$, go to (4);
    (2) Find $\boldsymbol{n} \in \boldsymbol{A}_2^{(k-1)}$ meeting $\boldsymbol{n}^{\mathrm{T}}\boldsymbol{P}^{(k-1)} > 0$ and $\boldsymbol{n}^{\mathrm{T}}\boldsymbol{P}^{(k)} = 0$;
    (3) Compute $\boldsymbol{G}^{(k)}$ by (32);
    (4) Compute $\boldsymbol{Q}^{(k)}$ by (31), and $\boldsymbol{d}^{(k)}$ by (33);
    (5) Compute $t^{(k)}$ by (35), and $\boldsymbol{P}^{(k+1)}$ by (34);
    (6) Let $k = k + 1$, if $\boldsymbol{G}^{(k)} \neq \boldsymbol{I}_{LK \times LK}$, go to (2);
    (7) Construct $\boldsymbol{N} = \boldsymbol{N}^{(k)}$ by (29), and by (36);
    (8) Break into $\boldsymbol{q}_1^{(k)}$ and $\boldsymbol{q}_2^{(k)}$ by (37);
    (9) If all elements of $\boldsymbol{q}_1^{(k)}$ are more than 0, stop;
    (10) Choose any element less than 0 from $\boldsymbol{q}_1^{(k)}$,
and delete the corresponding row of $\boldsymbol{A}_1^{(k)}$;
    (11) Reconstruct $\boldsymbol{N} = \boldsymbol{N}^{(k)}$ by (29);
    (12) Compute $\boldsymbol{G}^{(k)}$ by (30), and $\boldsymbol{Q}^{(k)}$ by (31);
    (13) Compute$\boldsymbol{d}^{(k)}$ by (33), and $t^{(k)}$ by (35);
    (14) Compute $\boldsymbol{P}^{(k+1)}$ by (34);
    (15) Let $k = k + 1$ , and go to (7).
Output: the probability vector $\boldsymbol{P}$
```

5. Experimental Results

In order to evaluate performance of the proposed KPKM model (equivalent to KFCM at m = 1) solved by the FAGP algorithm, we have conducted a lot of experiments on one synthetic dataset, ten UCI datasets (http://archive.ics.uci.edu/ml (accessed on 8 March 2021)) and the MNIST dataset (http://yann.lecun.com/exdb/mnist/ (accessed on 8 March 2021)). These datasets are detailed in Sections 5.1–5.3. In Section 5.1, we use a synthetic dataset to compare KPKM using a Gaussian kernel with KPKM using a linear kernel. In Sections 5.2 and 5.3, we compare the proposed KPKM with KFCM, KKM, FCM, and KM to evaluate the performance of KPKM solved by the proposed fast AGP. Moreover, the Sections 5.4 and 5.5 evaluate the descent stability and convergence speed of the proposed FAGP, respectively. In the experiments, we implemented our own MATLAB code for KPKM, KFCM, and KKM with two build-in functions, *sparse* and *full*, employed for matrix optimization. Moreover, we called MATLAB's build-in functions *fcm* and *kmeans* for FCM and KM, respectively.

All experiments were carried out on a PC with an Intel(R) Core(TM) i7-4790 CPU at 3.60 GHz, 8.00 G RAM, running Windows 7 and MATLAB 2015a.

5.1. The Experiment on the Synthetic Dataset

In this experiment, we analyzed the influences of the Gaussian radial basis function kernel and the linear kernel on KPKM when clustering one synthetic dataset, which is shown in Figure 2. The synthetic dataset contained 300 points, with 100 and 200 being from two linearly inseparable classes, disc and ring, respectively. The result is displayed in Figure 3. Obviously, Gaussian radial basis function KPKM can cluster perfectly, whereas linear KPKM cannot.

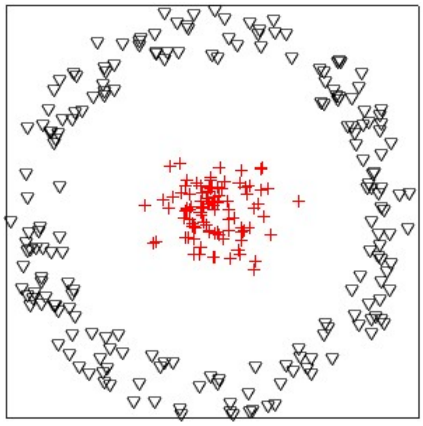


Figure 2. The synthetic dataset, which is composed of two linearly inseparable classes: disc and ring.

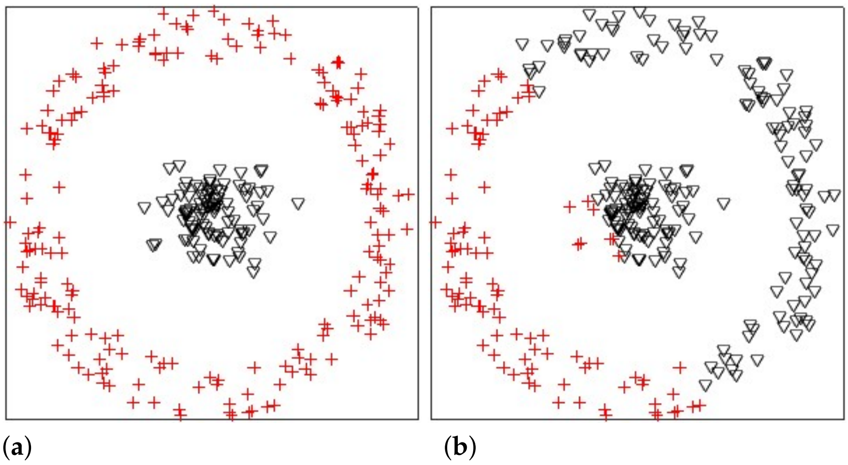


Figure 3. The results of the synthetic dataset clustered by Gaussian radial basis function kernel probabilistic k-means (KPKM) (**a**) and linear KPKM (**b**).

5.2. Experiment on Ten UCI Datasets

In this experiment, we compare the clustering performance of KPKM with the performances of KFCM, KKM, FCM, and KM in terms of three measures: normalized mutual information (NMI), adjusted Rand index (ARI), and v-measure (VM) [39,40]. NMI is a normalization of the mutual information score to scale the results between 0 (no mutual information) and 1 (perfect correlation). The NMI is defined as

$$\mathrm{NMI}(U,V) = \frac{\mathrm{MI}(U,V)}{\mathrm{mean}(H(U),H(V))} \tag{38}$$

where $H(U)$ is the entropy of U; MI is given by

$$\mathrm{MI}(U,V) = \sum_{i=1}^{|U|}\sum_{j=1}^{|V|} \frac{|U_i \cap V_j|}{N} \log \frac{N|U_i \cap V_j|}{|U_i||V_j|} \tag{39}$$

where $|U_i|$ is the number of the samples in cluster U_i. ARI is an adjusted similarity measure between two clusters. The ARI is defined as

$$\mathrm{ARI} = \frac{\mathrm{RI} - E(\mathrm{RI})}{\max(\mathrm{RI}) - E(\mathrm{RI})} \tag{40}$$

where RI is the ratio of data points clustered correctly to all data points, and E(RI) is the expectation of RI. VM is a harmonic mean between homogeneity and completeness, where homogeneity means that each cluster is a subset of a single class, and completeness means that each class is a subset of a single cluster. The VM is defined as

$$\mathrm{VM} = \frac{(1+\beta) \times \text{homogeneity} \times \text{completeness}}{(\beta \times \text{homogeneity} + \text{completeness})} \tag{41}$$

The higher the NMI, ARI and VM, the better the performance.

We describe the ten UCI datasets represented in Table 2 by name, code, number of instances, number of classes and number of dimensions. A set of experiential parameters was selected. We set m = 2 for KFCM, and m = 1.3 for FCM. With the Gaussian radial basis function kernel, we also selected appropriate σ (shown in Table 3), and report the clustering results in Table 4. The better results in each case is highlighted in bold. (The performance of many a method depends on parameters [41]. By experiments we found that for different algorithms with the same kernel function, the most appropriate parameters are usually different, so we used different settings to make the methods have the best clustering performances they could. We also tried to select the most appropriate parameter by using our experience.) We ran every algorithm 10 times, and average results were calculated.

From Table 4, we can see that

(1) For NMI, KPKM had the best clustering results on nine datasets, so the clustering performance of KPKM was the best for NMI.
(2) For ARI, KPKM had the best clustering results on six datasets. KFCM, KKM, FCM, and KM performed the best on 2, 1, 0, and 1 datasets, respectively, so KPKM performed the best for ARI.
(3) For VM, KPKM, KFCM, KKM, FCM, and KM had the best clustering results on 7, 0, 2, 1, and 0 datasets, respectively. Thus, KPKM had the best clustering performance for VM.

Overall, KPKM is better than the other models.

Table 2. The ten UCI datasets used.

Name	Code	Instances	Classes	Dimensions
Iris	R1	150	3	4
Seeds	R2	210	3	7
Segmentation	R3	210	7	19
Glass	R4	214	6	9
Ionosphere	R5	351	2	33
Dermatology	R6	358	6	34
Breast-cancer	R7	683	2	9
Natural	R8	2000	9	294
Yeast	R9	2426	3	24
Waveform	R10	5000	3	21

Table 3. Parameter σ selected for kernel clustering methods.

Dataset	KPKM	KFCM	KKM
R1	1.08	1.22	0.9
R2	1.9	2	2
R3	510	530	540
R4	510	100	510
R5	1.5	1	1.3
R6	3.3	2	18
R7	12	10	15
R8	3	0.9	2.7
R9	15	10	25
R10	10	15	13

Table 4. Comparisons of KPKM with kernel fuzzy c-means (KFCM), kernel k-means (KKM), fuzzy c-means (FCM), and k-means (KM) on ten UCI datasets (DS) in terms of normalized mutual information (NMI), adjusted Rand index (ARI), and v-measure (VM).

DS	Method	KPKM	KFCM	KKM	FCM	KM
R1	NMI	**0.8146**	0.7900	0.7820	0.6723	0.6733
	ARI	**0.8119**	0.8015	0.7590	0.5763	0.5779
	VM	**0.8146**	0.7900	0.7820	0.7081	0.7149
R2	NMI	**0.7073**	0.6949	0.7038	0.6949	0.7025
	ARI	0.7141	0.7166	**0.7231**	0.7166	0.7135
	VM	**0.7073**	0.6949	0.7038	0.6949	0.6999
R3	NMI	**0.5555**	0.5503	0.5154	0.4678	0.5132
	ARI	0.3909	**0.4141**	0.3429	0.3172	0.3313
	VM	0.5553	0.5503	0.5139	**0.5729**	0.5252
R4	NMI	**0.4436**	0.3594	0.3943	0.3489	0.4178
	ARI	**0.2796**	0.2137	0.2542	0.2126	0.2551
	VM	**0.4424**	0.3593	0.3934	0.3807	0.3857
R5	NMI	0.2476	0.2390	**0.2715**	0.1299	0.1349
	ARI	0.1747	0.1098	0.1657	0.1727	**0.1777**
	VM	0.2476	0.2390	**0.2715**	0.1298	0.1348
R6	NMI	**0.2919**	0.2068	0.2778	0.1046	0.1032
	ARI	**0.1795**	0.1388	0.1698	0.0261	0.0266
	VM	**0.2919**	0.2065	0.2776	0.1095	0.1056
R7	NMI	**0.7903**	0.7825	0.7741	0.7478	0.7478
	ARI	**0.8796**	0.8741	0.8674	0.8464	0.8464
	VM	**0.7903**	0.7825	0.7741	0.7478	0.7478
R8	NMI	**0.0531**	0.0326	0.0556	0.0521	0.0536
	ARI	0.0253	**0.0303**	0.0283	0.0273	0.0261
	VM	0.0525	0.0312	**0.0550**	0.0495	0.0529
R9	NMI	**0.0052**	0.0031	0.0050	0.0043	0.0050
	ARI	**0.0118**	0.0084	0.0118	0.0109	0.0117
	VM	**0.0052**	0.0031	0.0050	0.0045	0.0045
R10	NMI	**0.3654**	0.3162	0.3637	0.3606	0.3622
	ARI	**0.2546**	0.2377	0.2541	0.2529	0.2536
	VM	**0.3654**	0.3162	0.3637	0.3559	0.3622

5.3. Experiment on the MNIST Dataset

In this experiment, we used the MNIST dataset to compare KPKM with KFCM and KKM when clustering digital images. The MNIST dataset was composed of handwritten digits, with 60,000 examples for training and 10,000 examples for testing. All digits were size-normalized and centered in fixed-size images. To evaluate the clustering performances of KPKM, KFCM, and KKM, we randomly chose 1000 training examples; 100 are shown in Figure 4.

Moreover, we defined a CWSS kernel based on complex wavelet structural similarity (CWSS) [42]. CWSS may be regarded as a coefficient that does not change the structural content of an image. By using $CWSS(\boldsymbol{x}, \boldsymbol{y})$ to denote the similarity between two images x and y, we can express the CWSS kernel as

$$K_{\text{cwss}}(\boldsymbol{x}, \boldsymbol{y}) = \exp\left(-\frac{1 - CWSS(\boldsymbol{x}, \boldsymbol{y})}{2\sigma^2}\right) \tag{42}$$

where $\sigma = 5$ is set for KFCM, and $\sigma = 1$ for both KPKM and KKM. Additionally, m = 1.3 is set for KFCM. We present the results in Table 5, with the examples of Figure 4 correspondingly being displayed in Figure 5. From Table 5, we can see that KPKM outperformed KFCM and KKM in terms of NMI, ARI, and VM. From Figure 5, we can observe that both KPKM and KKM found ten clusters, whereas KFCM found only seven clusters, although the number of clusters was set to ten.

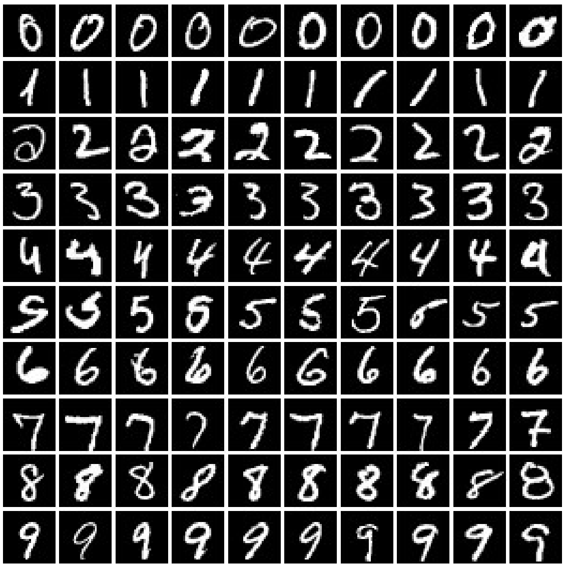


Figure 4. One-hundred digital examples, 10 per class.

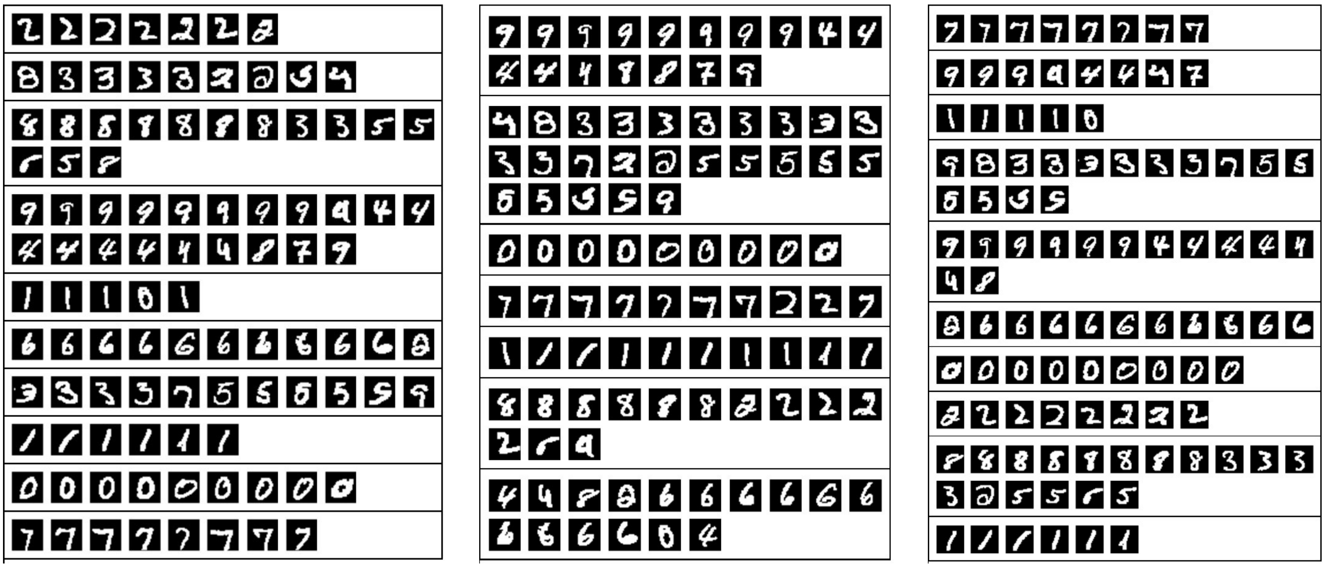


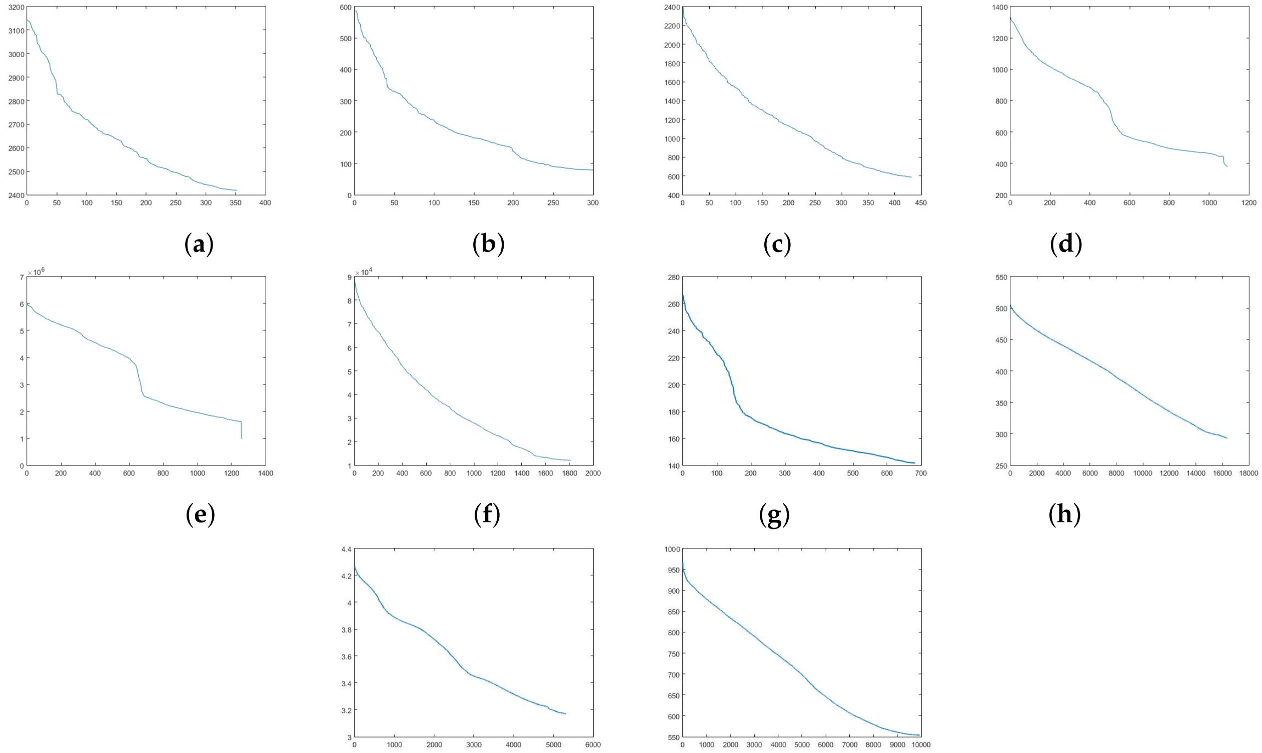
Figure 5. The results of the examples of Figure 2 clustered by KPKM (**a**), KFCM (**b**), and KKM (**c**).

Table 5. Comparisons of KPKM with KFCM and KKM on the MNIST dataset in terms of NMI, ARI, and VM.

Measure	KPKM	KFCM	KKM
NMI	**0.6830**	0.5438	0.6613
ARI	**0.5685**	0.4047	0.5256
VM	**0.6829**	0.5411	0.6613

5.4. Experiment for Descent Stability

In this experiment, we used 10 UCI datasets to evaluate descent stability of FAGP. AGP selects a small step length to converge. If AGP selects a large step length, the objective function value may descend with oscillation, and even does not converge. FAGP iteratively estimates a maximum step length at each iteration for speeding up its convergence. However, does it have any serious influence on the convergence? we ran the proposed FAGP on 10 UCI datasets to demonstrate the descent stability of FAGP. As shown in Figure 6, we can see that FAGP descends stably without oscillation at each iteration.



(**a**) (**b**) (**c**) (**d**)

(**e**) (**f**) (**g**) (**h**)

(**i**) (**j**)

Figure 6. Descent stability of FAGP on 6 UCI datasets: Ionosphere (**a**), Iris (**b**), Seeds (**c**), Glass (**d**), Segmentation (**e**), Dermatology (**f**), Breast (**g**), Natural (**h**), Yeast (**i**), and Waveform (**j**). The x-axis shows the iteration number. The y-axis shows the objective function value.

5.5. Convergence Speed Comparison between FAGP and AGP

In this experiment, we compared the convergence speeds of FAGP and AGP by using running time on 10 UCI datasets. η is the ratio of FAGP's running time to AGP's. FAGP and AGP used the same initializations in each case. The results are presented in Tables 6 and 7 ("–" means the running time was too long to obtain the final clustering results), and we can observe the proposed FAGP ran faster than AGP on all the 6 datasets. For Iris and Seeds datasets, FAGP used less than 10% of running time of AGP. FAGP also required fewer iterations than AGP. For large datasets, the running time of AGP was too long, but the proposed fast AGP could obtain the final clustering results.

Table 6. Running time (s) comparison of FAGP and AGP.

DS	FAGP	AGP	η
Ionosphere	0.719763	2.579700	27.90%
Iris	0.397728	9.410401	4.22%
Seeds	0.693613	8.650179	8.01%
Glass	3.482856	22.644511	15.38%
Segmentation	4.706316	19.125480	24.60%
Dermatology	10.659099	50.796225	20.98%
Breast	1.0153	6.1325	16.55 %
Natural	2309	–	–
Yeast	197.37	–	–
Waveform	213.64	–	–

Table 7. Comparison of the numbers of iterations used by FAGP and AGP.

DS	FAGP	AGP
Ionosphere	354	975
Iris	357	6506
Seeds	423	4061
Glass	1123	5616
Segmentation	1275	3986
Dermatology	1768	6784
Breast	682	694
Natural	16,378	–
Yeast	5609	–
Waveform	9939	–

6. Conclusions

In this paper, a novel clustering model (i.e., KPKM) was proposed. The proposed KPKM solves the problem of KFCM for $m = 1$, and this problem cannot be solved by existing method. The traditional AGP method can solve the proposed KPKM, but the efficiency of AGP is low. A fast AGP was proposed to speed up the AGP. The proposed fast AGP uses a maximum step length strategy to reduce the iteration number and uses an iterative method to update the projection matrix. The experimental results demonstrated that the fast AGP is able to solve the KPKM and the fast AGP requires less running time than AGP (the proposed FAGP requires 4.22–27.90% of the running time of AGP on real UCI datasets). The convergence of the proposed method was also analyzed by experiments. Additionally, in the experiments, the KPKM model could produce overall better clustering results than the other models, including KFCM, KKM, FCM, and KM. The proposed KPKM obtained the best clustering results on at least 6 real UCI datasets (a total of 10 real UCI datasets were used).

As future work, the proposed KPKM using other kernels will be evaluated in a variety of applications. For large datasets, the proposed method still has some disadvantages, so the next project will include speeding up the AGP on large datasets.

Author Contributions: Data curation, Z.L. and Z.Z.; writing—original draft, B.L.; writing—review and editing, T.Z. and Y.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Natural Science Foundation of China (61876010, 61806013, and 61906005).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: All authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

$\boldsymbol{X}$	dataset
$\boldsymbol{x}_i$	i-th data points of $\boldsymbol{X}$
$\varphi(\cdot)$	feature mapping from data points to kernel Hilbert space
$K_{\text{lap}}(\cdot,\cdot)$	Laplace Radial Basis Function kernel
$K_{\text{gau}}(\cdot,\cdot)$	Gaussian Radial Basis Function kernel
$K_{\text{pol}}(\cdot,\cdot)$	polynomial kernel
$K_{\text{tan}}(\cdot,\cdot)$	sigmoid kernel
ω_j	j-th cluster
L	number of elements in $\boldsymbol{X}$
L_j	number of elements in ω_j
$\boldsymbol{c}_j$	center of ω_j
K and C	number of clustering center
$\boldsymbol{W}$	membership degree matrix
w_{ij}	membership degree to the i-th data point in the j-th cluster
$\boldsymbol{A}$	inequality matrix
$\boldsymbol{E}$	equality matrix
$\boldsymbol{P}$	probability matrix
$\boldsymbol{I}$	identity matrix
p_{ij}	probability to the i-th data point in the j-th cluster
$\boldsymbol{G}$	projection matrix
$\boldsymbol{Q}$	orthogonal projection matrix
$\boldsymbol{N}$	active matrix
$\boldsymbol{n}$	row vector of $\boldsymbol{N}$
$\boldsymbol{d}$	projected gradient
t	step length

References

1. Lloyd, S. Least Squares Quantization in PCM. *IEEE Trans. Inf. Theory* **1982**, *28*, 129–137.
2. Bezdek, J.C. Pattern Recognition with Fuzzy Objective Function Algorithms. *Adv. Appl. Pattern Recognit.* **1981**, *22*, 203–239.
3. Jing, L.; Ng, M.K.; Huang, J.Z. An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data. *IEEE Trans. Knowl. Data Eng.* **2007**, *19*, 1026–1041.
4. Xenaki, S.; Koutroumbas, K.; Rontogiannis, A. Sparsity-aware possibilistic clustering algorithms. *IEEE Trans. Fuzzy Syst.* **2016**, *24*, 1611–1626, .
5. Yang, M.S.; Nataliani, Y. A feature-reduction fuzzy clustering algorithm based on feature-weighted entropy. *IEEE Trans. Fuzzy Syst.* **2018**, *26*, 817–835.
6. Gu, J.; Jiao, L.; Yang, S.; Liu, F. Fuzzy double c-means clustering based on sparse self-representation. *IEEE Trans. Fuzzy Syst.* **2018**, *26*, 612–626.
7. Hamasuna, Y.; Endo, Y.; Miyamoto, S. On tolerant fuzzy c-means clustering and tolerant possibilistic clustering. *Soft Comput.* **2010**, *14*, 487–494, .
8. Li, Y.; Yang, G.; He, H. A study of large-scale data clustering based on fuzzy clustering. *Soft Comput.* **2016**, *20*, 3231–3242.
9. Zhu, L.F.; Wang, J.S.; Wang, H.Y. A Novel Clustering Validity Function of FCM Clustering Algorithm. *IEEE Access* **2019**, *7*, 152289–152315.
10. Sinaga, K.P.; Yang, M. Unsupervised K-Means Clustering Algorithm. *IEEE Access* **2020**, *8*, 80716–80727.
11. Wang, C.; Pedrycz, W.; Yang, J.; Zhou, M.; Li, Z. Wavelet Frame-Based Fuzzy C-Means Clustering for Segmenting Images on Graphs. *IEEE Trans. Cybern.* **2020**, *50*, 3938–3949.
12. Wang, C.; Pedrycz, W.; Li, Z.; Zhou, M.; Ge, S.S. G-image Segmentation: Similarity-preserving Fuzzy C-Means with Spatial Information Constraint in Wavelet Space. *IEEE Trans. Fuzzy Syst.* **2020**, doi:10.1109/TFUZZ.2020.3029285.
13. Zhang, R.; Li, X.; Zhang, H.; Nie, F. Deep Fuzzy K-Means With Adaptive Loss and Entropy Regularization. *IEEE Trans. Fuzzy Syst.* **2020**, *28*, 2814–2824.
14. Wang, C.; Pedrycz, W.; Zhou, M.; Li, Z. Sparse Regularization-Based Fuzzy C-Means Clustering Incorporating Morphological Grayscale Reconstruction and Wavelet Frames. *IEEE Trans. Fuzzy Syst.* **2020**, doi:10.1109/TFUZZ.2020.2985930.
15. Wang, C.; Pedrycz, W.; Li, Z.; Zhou, M.; Zhao, J. Residual-sparse Fuzzy C-Means Clustering Incorporating Morphological Reconstruction and Wavelet frame. *IEEE Trans. Fuzzy Syst.* **2020**, doi:10.1109/TFUZZ.2020.3029296.

16. Zhang, R.; Nie, F.; Guo, M.; Wei, X.; Li, X. Joint Learning of Fuzzy k-Means and Nonnegative Spectral Clustering With Side Information. *IEEE Trans. Image Process.* **2019**, *28*, 2152–2162.
17. Wang, X.; Yu, F.; Pedrycz, W.; Wang, J. Hierarchical clustering of unequal-length time series with area-based shape distance. *Soft Comput.* **2019**, *23*, 6331–6343.
18. Li, Y. A Clustering Algorithm Based on Maximal θ-Distant Subtrees. *Pattern Recognit.* **2007**, *40*, 1425–1431.
19. Esgario, G.M.; Krohling, R.A. Clustering with Minimum Spanning Tree using TOPSIS with Multi-Criteria Information. In Proceedings of the IEEE International Conference on Fuzzy Systems, Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–7.
20. Rodriguez, A.; Laio, A. Clustering by Fast Search and Find of Density Peaks. *Science* **2014**, *344*, 1492–1496.
21. Bryant, A.; Cios, K. RNN-DBSCAN: A density-based clustering algorithm using reverse nearest neighbor density estimates. *IEEE Trans. Knowl. Data Eng.* **2018**, *30*, 1109–1121.
22. Xu, X.; Ding, S.; Xu, H.; Liao, H.; Xue, Y. A feasible density peaks clustering algorithm with a merging strategy. *Soft Comput.* **2019**, *23*, 5171–5183.
23. Wang, X.; Zhang, Y.; Xie, J. A density-core-based clustering algorithm with local resultant force. *Soft Comput.* **2020**, *24*, 6571–6590.
24. Wu, C.; Lee, J.; Isokawa, T.; Yao, J.; Xia, Y. Efficient Clustering Method Based on Density Peaks With Symmetric Neighborhood Relationship. *IEEE Access* **2019**, *7*, 60684–60696.
25. Liu, T.; Li, H.; Zhao, X. Clustering by Search in Descending Order and Automatic Find of Density Peaks. *IEEE Access* **2019**, *7*, 133772–133780.
26. Luxburg, U.V. A Tutorial on Spectral Clustering. *Stat. Comput.* **2007**, *17*, 395–416.
27. Chen, J.; Li, Z. Huang B. Linear Spectral Clustering Superpixel. *IEEE Trans. Image Process.* **2017**, *26*, 3317–3330.
28. Elhamifar, E.; Vidal, R. Sparse Subspace Clustering: Algorithm, Theory, and Applications. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 2765–2781.
29. Lu, C.; Feng, J.; Lin, Z.; Mei, T.; Yan, S. Subspace Clustering by Block Diagonal Representation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *41*, 487–501.
30. Gu, Y.; Chanussot, J.; Jia, X.; Benediktsson, J.A. Multiple Kernel Learning for Hyperspectral Image Classification: A Review. *IEEE Trans. Geosci. Remote Sens.* **2017**, *55*, 6547–6565.
31. Nguyen, B.; Baets, B.D. Kernel-Based Distance Metric Learning for Supervised k-Means Clustering. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *30*, 3084–3095.
32. Liu, X.; Zhu, X.; Li, M.; Wang, L.; Zhu, E.; Liu, T. Multiple Kernel K-means with Incomplete Kernels. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *42*, 1191–1204.
33. Marin, D.; Tang, M.; Ben, A.I.; Boykov, Y.Y. Kernel Clustering: Density Biases and Solutions. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *41*, 136–147.
34. Huang, H.C.; Chuang, Y.Y.; Chen, C.S. Multiple Kernel Fuzzy Clustering. *IEEE Trans. Fuzzy Syst.* **2012**, *20*, 120–134.
35. Rosen, J.B. The Gradient Projection Method for Nonlinear Programming. Part I. Linear Constraints. *J. Soc. Ind. Appl. Math.* **1961**, *9*, 514–532.
36. Goldfarb, D.; Lapidus, L. Conjugate Gradient Method for Nonlinear Programming Problems with Linear Constraints. *Ind. Eng. Chem. Fundam.* **1968**, *7*, 142–151.
37. Girolami, M. Mercer Kernel-Based Clustering in Feature Space. *IEEE Trans. Neural Netw.* **2002**, *13*, 780–784.
38. Honig, M.; Madhow, U.; Verdu, S. Blind Adaptive Multiuser Detection. *IEEE Trans. Inf. Theory* **1995**, *41*, 944–960.
39. Vinh, N.X.; Epps, J.; Bailey, J. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *J. Mach. Learn. Res.* **2010**, *11*, 2837–2854.
40. Rosenberg, A.; Hirschberg, J. V-measure: A conditional entropy-based external cluster evaluation measure. In Proceedings of the Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Prague, Czech Republic, 28–30 June 2007; pp. 410–420.
41. Gao, S.; Zhou, M.; Wang, Y.; Cheng, J.; Yachi, H.; Wang, J. Dendritic Neuron Model With Effective Learning Algorithms for Classification, Approximation, and Prediction. *IEEE Trans. Neural Netw. Learn. Syst.* **2019**, *30*, 601–614.
42. Sampat, M.P.; Wang, Z.; Gupta, S.; Bovik, A.C.; Markey, M.K. Complex Wavelet Structural Similarity: A New Image Similarity Index. *IEEE Trans. Image Process.* **2009**, *18*, 2385–2401.