A parameter refinement method for Ptychography based on Deep Learning concepts: Supplementary Material

Francesco Guzzi[1, 2, *], **George Kourousias**[1], **Alessandra Gianoncelli**[1], **Fulvio Billè**[1], **and Sergio Carrato**[2]

[1]Elettra Sincrotrone Trieste, Italy
[2]Image Processing Laboratory (IPL), University of Trieste, Italy
[*]francesco.guzzi@elettra.eu

ABSTRACT

In the following text we will expand the results and discussion section of the main text by analysing how the algorithm perform in the case of simulated datasets, where different type of perturbations are synthetically added.

Experiments on Synthetic Data

A ptychography simulator, implemented in the SciComPty modular framework, is used to create 5 synthetic datasets, starting from a set of standard test images and setup parameters which spans different propagation distances, from 0.065 m to 0.2 m. The wavelength is kept constant at 1 nm. All the computational experiments are written on PyTorch 1.2[1] and executed on a computer equipped with an Intel Xeon (R) E3-1245 v5 CPU running at 3.50 GHz. The entire code is implemented on GPU (Nvidia Quadro P2000), which is essential for this heavy-duty computational imaging. During the tests on synthetic data, the resolution for each diffraction pattern is limited to 128x128 pixels to reduce the computation time. The overlap factor is kept constant at around 70%, producing a scan pattern of 11x11 positions. A known random jitter is added to the ideal grid-based scan pattern to avoid the "raster scan pathology"[2]. The resulting total object size has a field of view of around 512x512 pixels (roughly 200 μ m). Within the context of this paper, the term "epoch", which is used interchangeably with "iteration", defines how many times the entire set of J diffraction patterns is processed through the algorithm.

The use of an autograd environment allows to easily experiment with the batch-size parameters: EPIE and DM are completely antipodal in this sense, as the first is a sequential algorithm (stochastic gradient descent), while the second employs all the measured data at once (gradient descent). The batch size hyper-parameter (here set at 5 probes) allows to span between the two worlds. Within this framework, new first-order optimisers such as Adam[3] become readily available, providing a considerable acceleration to the plain old gradient descent method described by eq. 4 and 5 in the main text.

To investigate the effects of our correction routine, for each dataset the propagation distance is initialised to a value corrupted by a 30% error, while each position is perturbed by a random jitter with a standard deviation of 10 pixels. Both the propagation distance and the position vectors are added to the optimisation pool. Apart from eye inspection, to validate the method, reconstruction quality can be analysed observing the behaviour of (I) the optimisation dissimilarity (eq. 6 in the main text), calculated between the simulated and the measured diffraction pattern, and (II) a truth-aware similarity metric, SSIM[4], which obviously can be applied for simulated data only. In the latter case, the positions error and the propagation distance estimate can be monitored at each iteration, producing informative graphics which are the base for the following analysis.

Fig. S1 and S2 show the convergence of the algorithm tested on one example dataset, depicted in Fig. S3: $O(\mathbf{r})$ is made by the "Pepperoni" (magnitude) and "Peaches" (phase) images, while $P(\mathbf{r})$ is composed by the "Chelsea cat" (magnitude) and "Astronaut" (phase) pictures. It can be seen that in around 500 epochs both the positions and the distance have recovered more than 90% of initial error, producing the object and illumination estimate which can be seen in Fig. S3. Note that the during the reconstruction of the illumination (the cat and astronaut insets in Fig. S3), no mask is used. The circular pattern is automatically retrieved by the procedure. The loss value (Fig. S1, top panel) is used as the guiding metrics to optimise the object, the illumination and the setup parameters. The use of the grid sampler is essential to increase the convergence, which tends to be instead slower for the case of optimisation-like reconstruction algorithm based on the a typical crop operator; the acceleration effect can be traced back to the inherent regularisation action of the interpolator.

In the uppermost panel of Fig. S2, the distribution of the position error is shown for each epoch, denoting a Rayleight-like distribution, which tends to be narrower as the epochs increase; the median, indeed decay towards 0, indicating a good

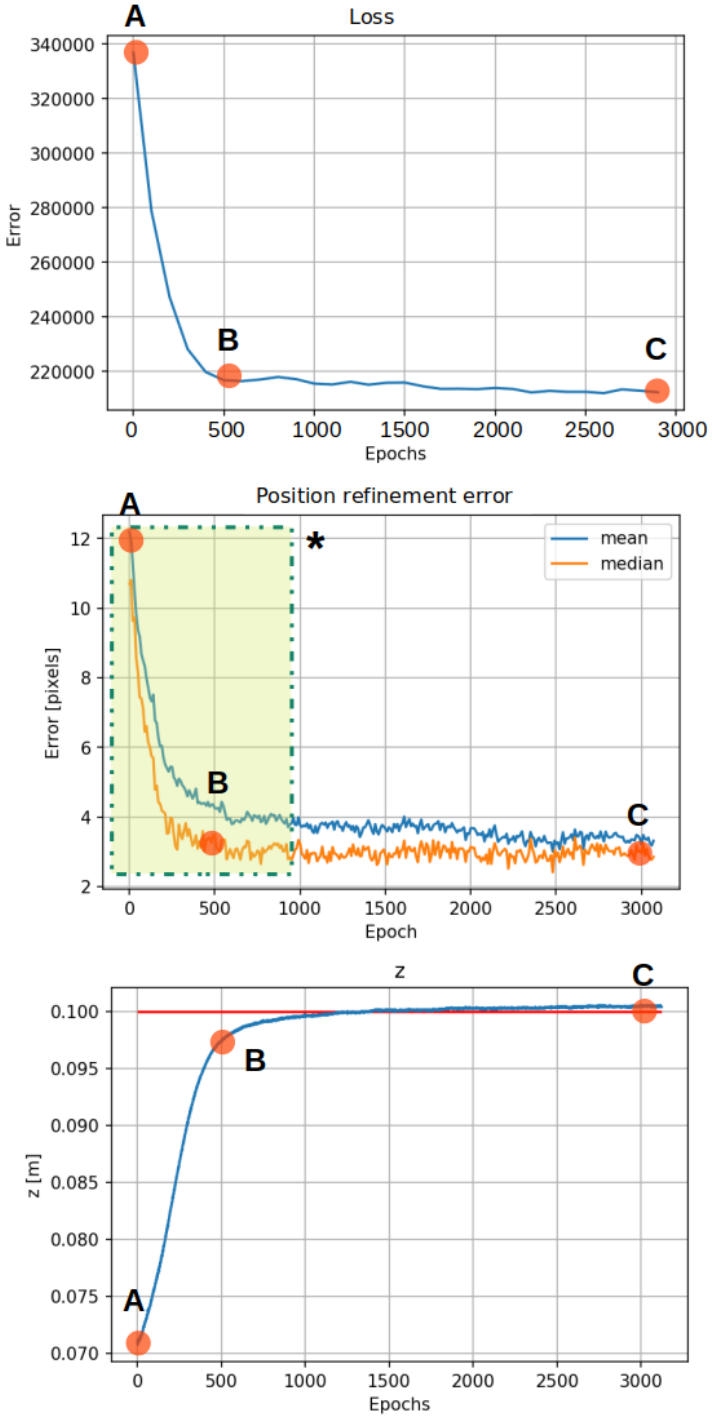

Figure S1. The convergence of our combined optimisation is analysed by observing: the loss value, the ground truth position refinement error and the propagation distance z inferred by the algorithm. In the bottom panel, the correct distance is denoted by the red horizontal line at $z = 0.1$m

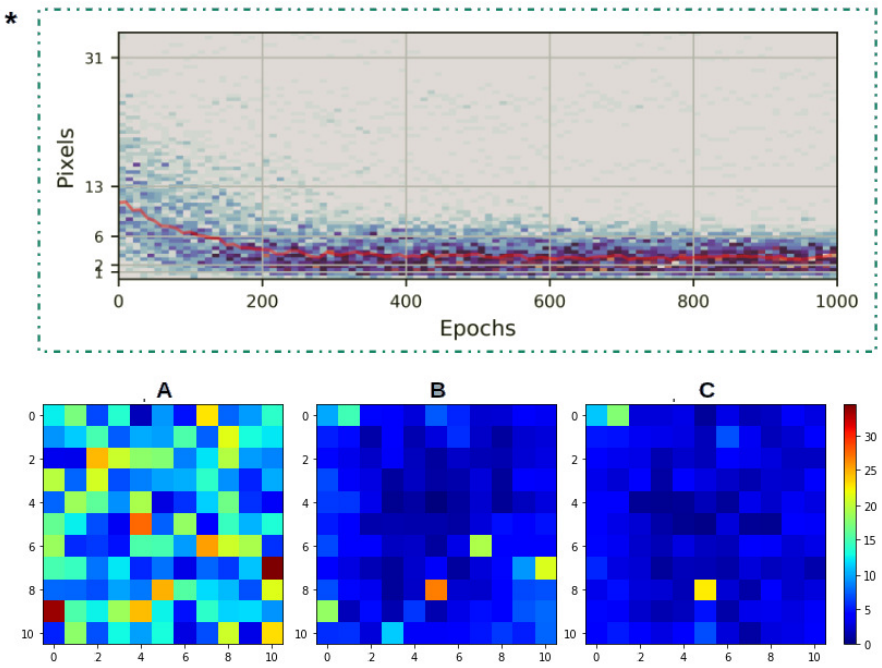

Figure S2. The uppermost graph is the zoomed version of the central panel in Fig. S1 (denoted by an asterisk). Here, the distribution of the positions errors is shown as epochs increase. It can be seen that the median decay towards 0 as the reconstruction proceeds. Panel A,B and C shows the ground truth error for each of the 11x11 positions.

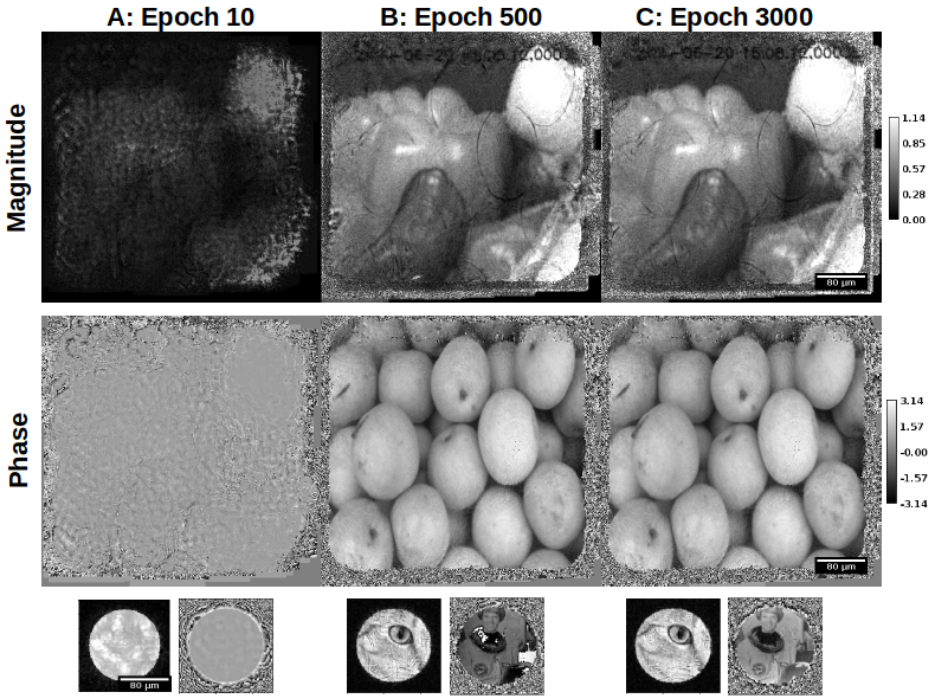

Figure S3. Evolution of the reconstruction for a synthetic dataset, shown at the epoch denoted by A, B and C in Fig. S1.

correction. The desired single-modality form of the distribution is an effect of a refined version of the problem expression: we have to optimise only additive correction factors for the positions, not just the positions. In this manner, an L_1 metric can be used to create a regularisation term on these correction factors.
In the bottom panel of Fig. S1 the convergence of the z is shown: similarly to the other two panels in Fig. S1, the optimisation is moving fast towards the exact value within the first 1000 iterations of the algorithm, denoting how the convergence of each trained variable is aligned with the others, manifesting an ensemble behaviour.

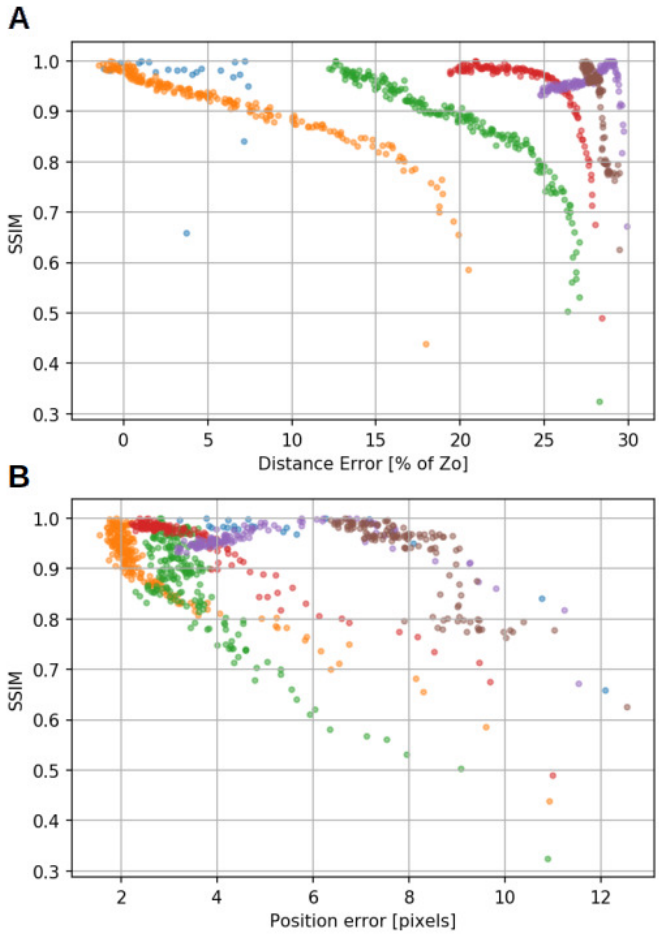

Figure S4. Normalised SSIM as a function of the propagation distance error (panel A) and the median of the scan positions error (panel B), calculated as the reconstruction progresses for many dataset (different colours). Convergence speed in the various phases can be guessed by observing the sparsity of the points for each error value.

To better analyse this aspect, Fig. S4 panel A shows how during the reconstruction, the SSIM value increases as the propagation distance (panel A) approaches to the correct value. Each color is relative to a different dataset. The longest trails are the one for which the propagation distance is larger, then the optimisation, initialised with a 30% error, spans for an extended range of z. Conversely, for small propagation distances, the regression of the correct value tends to be faster. Panel B shows a similar graph but for the median of the positions error. Convergence speed in the various phases can be guessed by observing the sparsity of the points for each error value.

References

1. Paszke, A. *et al.* PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Wallach, H. M. *et al.* (eds.) *Advances in Neural Information Processing Systems 32: NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, 8024–8035 (2019).

2. Thibault, P. *et al.* High-resolution scanning x-ray diffraction microscopy. *Science* **321**, 379–382, DOI: 10.1126/science.1158573 (2008).

3. Kingma, D. P. & Ba, J. Adam: A Method for Stochastic Optimization. In Bengio, Y. & LeCun, Y. (eds.) *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings* (2015).

4. Wang, Z., Bovik, A., Sheikh, H. & Simoncelli, E. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. on Image Process.* **13**, 600–612, DOI: 10.1109/tip.2003.819861 (2004).