Distributed Automatic Differentiation for Ptychography

Youssef S. G. Nashed† Tom Peterka, Junjing Deng, and Chris Jacobsen
Argonne National Laboratory
Argonne, IL 60439, USA

Abstract
Synchrotron radiation light source facilities are leading the way to ultrahigh resolution X-ray imaging. High resolution imaging is essential to understanding the fundamental structure and interaction of materials at the smallest length scale possible. Diffraction based methods achieve nanoscale imaging by replacing traditional objective lenses by pixelated area detectors and computational image reconstruction. Among these methods, ptychography is quickly becoming the standard for sub-30 nanometer imaging of extended samples, but at the expense of increasingly high data rates and volumes.

This paper presents a new distributed algorithm for solving the ptychographic image reconstruction problem based on automatic differentiation. Input datasets are subdivided between multiple graphics processing units (GPUs); each subset of the problem is then solved either entirely independent of other subsets (asynchronously) or through sharing gradient information with other GPUs (synchronously). The algorithm was evaluated on simulated and real data acquired at the Advanced Photon Source, scaling up to 192 GPUs. The synchronous variant of our method outperformed an existing multi-GPU implementation in terms of accuracy while running at a comparable execution time.

Keywords: inverse problems, image reconstruction, gradient methods, distributed algorithms, X-ray scattering

1 Introduction
In order to understand the behavior of heterogeneous materials at nanometer length scales, one must see their structure. This applies to integrated circuits where the as-manufactured structure may depart from the design, to the interaction of metals with organics in contaminated soil, and to molecular compartmentalization and transport within cells. While super-resolution light microscopy is providing new insights in biology when using fluorescence from one or a few specific molecule types, only electron and X-ray microscopy offer the possibility of nanoscale imaging of a material in its entirety, and it is only with X-rays that one can image specimens much thicker than a micrometer. However, in spite of demonstrations of about 10 nm resolution

*email: ynashed@anl.gov
Figure 1: Simplified ptychography experiment setup. A Cartesian grid is used for the overlapping raster scan positions.

X-ray lens-based imaging in high contrast test structures [7] or in inferred wavefront profiles [31], most lens-based X-ray microscopes are limited to 20-30 nm resolution for practical studies. An alternative approach is to collect the far-field X-ray diffraction pattern at large angles and use iterative phase retrieval to obtain higher resolution than X-ray lenses permit [30], but this basic approach of coherent diffraction imaging or CDI requires samples of very limited extent. X-ray ptychography [11, 20, 37], where far field coherent X-ray diffraction patterns are collected as a finite-sized coherent beam is scanned across the specimen with significant illumination spot overlaps, provides an alternative approach which is compatible with both large imaging fields of view and freedom from lens-imposed resolution limits.

While the set of far-field X-ray diffraction patterns recorded in ptychography capture the Fourier plane magnitudes of the scattered light, the phase is lost so that these diffraction patterns cannot be directly inverted to reveal the samples structure within each illumination spot. Mathematically, the phase inverse problem is ill-posed, meaning its solution is underdetermined and nonunique [21].

Phase retrieval algorithms [12] are designed to solve the phase problem by iteratively trying to find phases for the measured magnitudes that satisfy a set of constraints. In ptychography, the constraints are derived from diffraction data redundancy. Data redundancy is achieved by scanning the coherent illumination spot across the sample, and collecting a different diffraction pattern at each partially overlapping scan position (see Figure 1). Ptychography has been successful in imaging frozen-hydrated cells at 30 nm resolution [10], bacteria at 20 nm resolution [45], and integrated circuits at 41 nm resolution [18]. Ptychographic tomography has been used to image nanoporous glass to 16 nm 3D resolution [19].

Many methods have been proposed for solving the phase problem. The most commonly used algorithms follow an alternating projections scheme [39], where a randomly initialized object wave function guess is iteratively improved by replacing the magnitude of the predicted wave with the measurement. Variants of this method for ptychography include the Ptychographical Iterative Engine (PIE) [37], later extended and named ePIE [28], and the Difference Map algorithm [42]. The alternating projections approach follows naturally from the underlying physics, starting with a model that is sequentially updated using the forward problem formulation and heuristics about the experimental setup. There is, however, no guarantee that these
algorithms converge to the optimal solution. Moreover, it is inherently difficult to measure their computational complexity [35].

Alternatively, phase retrieval can be formulated as a nonlinear optimization problem [17]. The forward problem is mathematically described using a cost function, taking into account different noise models and regularization; the inverse problem is then solved by finding the global minimum of this cost function. This requires calculating the gradient of the cost function to direct the search algorithm. Typically, a gradient expression is explicitly derived by symbolically differentiating the cost function. This has the advantage of using ‘out-of-the-box’ optimization methods, like Gauss-Newton [49], conjugate gradient [17], or quasi-Newton [26], and being more computationally efficient than finite-difference methods, when appropriate approximations are used [35]. The major drawback of this method is the manual derivation of the gradient expression. Besides requiring assumptions and approximations in the forward model cost function, for which there exist symbolic derivatives, it also keeps the forward model tightly coupled with the gradient calculation. With any updates to the forward model, for example due to new experimental capabilities, the gradient expression needs to be derived again.

Automatic differentiation (AD) [36], also known as Algorithmic Differentiation, offers the simple expressibility of alternating projections methods, along with the power of gradient directed optimization methods. AD calculates partial derivatives of a function with respect to each of its input parameters by using the chain rule. The chain rule states that a derivative of a complex function can be automatically computed by combining derivatives of elementary operations, like arithmetic and trigonometric functions, that make up this function. The application of AD to the phase problem was highlighted by Jurling and Fienup who derived the complex-valued elementary operations specific to phase retrieval algorithms [24]. They applied ‘manual automatic differentiation’ by computing the gradient by hand, but using the principles of AD to convert forward model code to a series of gradient calculations. A related approach was recently presented to retrieve the 3D structure of a thick specimen [22], but in optical microscopy, employing a different forward model, error metric, and optimization strategy than what is presented here.

This paper investigates the use of AD for ptychographic phase retrieval. We focus on method accuracy when the gradient calculation is shared among distributed many-core computing resources and execution time on these resources. To summarize, the main contributions of this paper are:

- a fully automatic gradient calculation from source code for ptychography as will be discussed in Section 2.2;
- an algorithm for distributing a ptychographic dataset and gradient synchronization among multiple GPUs;
- a comparison of the new algorithm, in terms of convergence and performance, with an existing method.

2 Background

This section defines the notation and terminology used throughout the paper. It overviews the ptychographic phase retrieval forward model, and introduces AD basics and tools utilized in calculating gradients for this model.
2.1 Ptychography

Ptychography solves the phase problem using interfering diffraction patterns from overlapping scan areas in the object space. The idea itself is decades old [20], but its adoption accelerated in the last decade [8,15,23,38,42,47]. This is mainly due to increased robustness of the inversion process, compared to previous CDI techniques [2,48], and the ability to image objects much larger than the focused beam at a resolution that is, theoretically, only limited by the beam wavelength.

Figure 1 shows a simplified diagram of a ptychography experiment. A focused beam is used to raster scan an object in a predefined arrangement of spatially overlapping beam locations, generating a set of \( J \) "far-field" diffraction patterns at the detector plane. The forward model for ptychography can be written as

\[
\psi_j(r) = P(r - r_j) \circ O(r). \tag{1}
\]

The complex-valued object wave function \( O(r) \) interacts with the beam wave function (termed \( \text{Probe} \) hereafter) \( P(r) \) at position \( r_j \). This interaction is approximated by the complex element-wise Hadamard product operator \( \circ \) of the two wave functions, generating an exit wave function \( \psi_j(r) \) that is propagated to the detector plane, and then measured as a real-valued diffraction pattern \( I_j(q) \) defined as

\[
I_j(q) = |\mathcal{F}[\psi_j(r)]|^2, \tag{2}
\]

where \( \mathcal{F}[\cdot] \) denotes the Fourier transform from real space \( r \) to reciprocal space \( q \).

Alternating projections reconstruction algorithms start with an initial guess of \( O(r) \), and sometimes \( P(r) \), then proceed with calculating the exit wave function as in Equation (1). A new estimate \( \Psi_j(q) \) is then computed by replacing the modulus of the Fourier transform of the \( j \)-th exit wave with the square root of the measured diffraction pattern intensity, such that

\[
\Psi_j(q) = \sqrt{I_j(q)} \frac{\mathcal{F}[\psi_j(r)]}{|\mathcal{F}[\psi_j(r)]|}. \tag{3}
\]

A new exit wave \( \psi_j'(r) \) can then be computed by means of an inverse Fourier transform, as in

\[
\psi_j'(r) = \mathcal{F}^{-1}[\Psi_j(q)]. \tag{4}
\]

This can be done for all \( J \) relative shifts in \( r \) at once [42], or sequentially and in random order [28]. The residual \( \{\psi_j'(r) - \psi_j(r)\} \) is then used to refine the object guess \( O(r) \) in an iterative fashion.

As was stated in Section 1, gradient methods formulate an error metric based on the given information and forward model specification. The simplest form of this error metric for ptychography can be written as

\[
\mathcal{E} = \frac{1}{J} \sum_{j=1}^{J} \{ |\mathcal{F}[\psi_j(r)]|^2 - I_j(q) \}^2, \tag{5}
\]

essentially the Mean Squared Error (MSE) estimator measuring how well a forward model agrees with measured data. More sophisticated error metrics can be defined, including regularization and weighting functions to account for noise, numerical instability, bad detector pixels, or beam stops [13,17]. An analytic gradient expression is manually derived from Equation (5) by expanding and differentiating the error metric function. Guided by the gradient information, finding the minimum of this error function often yields a good estimate of the object wave function.
2.2 Automatic Differentiation

AD solves many problems with symbolic and numerical differentiation, such as computational inefficiency and errors introduced by domain discretization. It can automatically provide derivatives, high-order derivatives, and partial derivatives with respect to many input parameter functions defined in computer source code. The majority of modern programming languages have tools for AD [3, 5, 16, 46], which work either by source code transformation (SCT) or operator overloading (OO). SCT tools run before the language compiler, generating source code for derivative calculation from existing functions source code. OO tools provide the datatypes and elementary operators to compute partial derivatives, apply the chain rule, and define the function to be differentiated. Both approaches typically use a computational graph [43], a directed acyclic graph in which the vertices are operators or independent variables, and the edge weights equal the partial derivative of a node with respect to the edge source node. The computational graph is traversed to compute the root node derivative by aggregating the partial derivatives along all paths to leaf nodes, applying the chain rule for every edge weight [6].

The past few years have seen a growing interest in AD from both academia and industry, fueled by a need for generic, user-friendly deep learning toolkits. The backpropagation learning algorithm used to train deep neural networks is a special case of reverse mode AD [27, 40]. Consequently, several libraries available for deep learning also include high performance routines for constructing computational graphs and traversing them, from top to bottom, computing partial derivatives along the way to tune ‘neurons’ in a multilayer neural network [1, 4]. TensorFlow [1] is a Python based deep learning API provided by Google with OO AD and GPU implementation. In this paper, we use TensorFlow to automatically calculate gradients for the ptychography forward model and error metric previously defined.

3 AD for Ptychography

Ptychographical reconstruction algorithms typically try to estimate the Fourier phases lost during measurement of the diffraction patterns (Fourier magnitudes). Once the two components, phase and magnitude, of the Fourier function are known, the object wave function is obtained by the inverse Fourier transform in Equation (4). In our case, however, the error metric comparison defined in Equation (5) is performed directly on the Fourier magnitude values. Instead of solving for the Fourier phases first (in order to obtain the object function), the object function is updated iteratively using gradient information given by AD. AD computes the partial derivative of $E$ with respect to each pixel value in $O$. In practice, the complex-valued object function is decomposed into its real and imaginary components, $O^r$ and $O^i$ respectively, and two partial derivatives are computed for each component separately, $\frac{\partial E}{\partial O^r}$ and $\frac{\partial E}{\partial O^i}$. Other independent variables, such as the probe function $P$, can also be added to this framework. Currently, our algorithm requires a known probe function and retrieves the object function.

Once the partial derivatives are computed, minimizing $E$ can be achieved by any gradient-based optimization method. In this paper we employ the Adam algorithm for stochastic optimization [25], with the following hyperparameter values: 0.8 for the learning rate, 0.9 as the exponential decay rate of the first moment estimates, and 0.99 for the exponential decay rate of the second moment estimates.
3.1 Distributed Algorithm

Input datasets for ptychography consist of diffraction pattern images measured from spatially overlapping scan positions. Therefore, there exists a decomposition of the input dataset that maintains mutual spatial information among diffraction patterns, and in which the inverse problem is locally and independently solvable. In the case of Cartesian grid scans, a regular 2D decomposition suffices \[18, 29, 33\]. Solving the ptychographic phase retrieval problem for each sub-dataset will result in a part of the reconstructed object image. Those parts can be merged to form the final reconstruction since the original sub-datasets overlap at the decomposition borders.

Similar to our previous multi-GPU implementation of the PIE and ePIE reconstruction algorithms \[33\], we implemented two variants of the new AD-based methods. An asynchronous version, AD\_async, runs the computations independently on separate GPUs without any communication except at the end of the reconstruction, when all partial results are stitched back together using a parallel reduction-with-merge algorithm \[34\]. A synchronous version, AD\_sync, communicates local gradient information globally to all running GPUs at every iteration, employing a Radix-k communication algorithm \[34\], effectively sharing a synchronized object function among all GPUs.

AD\_async delivers the best scaling performance in a multi-GPU distributed environment, due to limited communication between computing resources. Post-reconstruction stitching, however, is not as simple as a parallel gather operation. Each GPU’s partial reconstruction exhibits spatial offsets and phase shifts relative to all other GPU reconstructions. 2D registration and phase modulation methods are required to merge all partial reconstructions into one coherent object wave function. We use a phase correlation algorithm for 2D rigid registration and a gain compensation technique, such as is used for photographic image stitching, for finding a common phase offset for the final reconstruction \[33\].

Figure 2 shows a data flow chart of the AD\_sync algorithm. The object function \(O\) is randomly initialized and shared among all GPUs. The data subdivision scheme is common to both synchronous and asynchronous versions of the algorithm. Starting with scan area extents as the domain to be decomposed, regular 2D offset sets \([d_1, d_2, \ldots d_N]\) are computed based on

![Figure 2: Block diagram of the AD\_sync algorithm running on N GPUs. Input data are in gray, independent and intermediate variables are in blue, and operators are in orange.](image-url)
the number \( N \) of GPUs used. Those offsets are used to define subsets in the diffraction pattern dataset \( D_j \) and scan positions list \( r_j \). Each GPU \( n \) calculates its current diffraction pattern estimate \( I_{j+dn} \) for a specific 2D region of \( O \) indexed by a different \( r_j \) subset. A local MSE \( E_n \) is calculated from the estimate and the measured data subset \( D_{j+dn} \), for which the AD tool derives a local gradient \( g_n \). In AD_async, \( g_n \) is used to update GPU’s local copies of \( O \), while in AD_sync, local gradients are aggregated forming a global gradient \( G \), which is then used to update the shared object function. This process is repeated for a predefined number of iterations, after which the AD_sync algorithm is done. AD_async, merges local copies of \( O \) into one final reconstruction using the stitching algorithm introduced above.

### 3.2 Implementation Details

Our software stack is highlighted in Figure 3. Diffraction pattern data are stored in HDF5 format [14]. TensorFlow’s Python API is used to construct the computational graph for our forward model, derive gradients for its error metric, and adjust the model parameters accordingly. Each GPU runs a local copy of TensorFlow where distributed memory parallelism is explicitly handled using MPI. Domain decomposition is achieved using the DIY parallel programming library [32] that is written on top of MPI to facilitate communication between parallel processes. In DIY terminology, we assign a DIY block, managed by one MPI rank, to each GPU. DIY also provides the parallel reduction algorithm used in AD_async and the Radix-k communication pattern employed at every iteration of AD_sync. To bridge between TensorFlow and DIY, a wrapper layer, pyDIY, is developed. It is responsible for marshalling Python data structures to C++, without the need for copying or serialization.

![Figure 3: Software stack for distributed AD-based ptychography.](image)

### 4 Evaluation

Evaluation was performed using two datasets: a synthetic sample simulating the diffraction patterns from a known image, and on real data from the Bionanoprobe [9] at beamline 21-ID-D of the Advanced Photon Source at Argonne National Laboratory. Experiments were run on the Cooley cluster at the Argonne Leadership Computing Facility (ALCF). Cooley is a visualization platform consisting of 126 compute nodes; each node has 12 CPU cores and one NVIDIA Tesla K80 dual-GPU card. Each GPU has 12 GB of memory and a CUDA compute capability of 3.7; the code was built and run with CUDA v7.5.
Figure 4: Simulated dataset performance evaluation plots. The time reported is the total running time of the algorithms in seconds, including I/O time. NRMSE is the Normalized Root Mean Squared Error calculated per-pixel between object function estimates and the ground truth.

4.1 Simulated Data

We compared AD\_sync, AD\_async, and our previous implementations of the PIE algorithm, termed PIE\_sync and PIE\_async, in terms of accuracy and performance. A synthetic pure phase object was raster scanned using a regular 160×160 Cartesian grid, generating a total of 25,600 far-field diffraction patterns, each of 128×128 pixels, for a total of 1.56 GB of single precision floating point raw data. The diffraction patterns were generated with a single-pixel detector point-spread function, with no added noise and 90% overlap between adjacent scan points in the horizontal and vertical directions. Results were obtained for different numbers of GPUs after 200 iterations of all algorithms.

Performance and accuracy plots of different GPU configurations [1-128] are reported in Figure 4. Our new algorithm has a higher memory requirement than the previous implementation. Therefore, time was only reported for AD\_sync and AD\_async running on 4 or more GPUs, while PIE\_sync and PIE\_async were able to fit the entire simulated dataset on one GPU. This is because of TensorFlow creating auxiliary arrays for storing the computational graph and computing the gradients. The top left plot shows mean runtime of 5 independent runs of all algorithms. It is clear that scaling is almost linear for the asynchronous algorithms, AD\_async and PIE\_async, while synchronous ones suffer from communication overhead. The top right
plot shows the algorithms scaling efficiency using a 4 GPU baseline. Again, asynchronous algorithms exhibit better scaling efficiency than asynchronous algorithms. The average scaling efficiency for $AD_{sync}$ is 65.7%, $AD_{async}$ is 77.7%, $PIE_{sync}$ is 61.8%, and $PIE_{async}$ is 95.9%.

The bottom left plot is a weak scaling test for the new algorithms, in which the workload assigned to each GPU was kept constant by doubling the input dataset size with the number of assigned GPUs.

The bottom right plot of Figure 4 depicts the convergence of all algorithms, measured employing a Normalized Root Mean Squared Error (NRMSE). NRMSE was calculated pixel-wise between object estimates and their corresponding 2D regions in the ground truth, for each GPU and per iteration. Individual GPU errors are aggregated for all GPUs of a certain configuration and averaged across 5 independent runs. The plotted quantity is the mean NRMSE of all GPU configurations of each algorithm. Both versions of our new algorithm, $AD_{sync}$ and $AD_{async}$, outperform their PIE counterparts in terms of final reconstruction quality and NRMSE standard deviation between runs. It is also evident from the plot that synchronous versions of the algorithm have better convergence than the asynchronous versions. This is mainly because of the increased statistics, found along scan region decomposition boundaries, that are only available to communicating GPUs in the synchronous approach.

4.2 Experimental Data

In order to evaluate our algorithm’s applicability and performance, we tested it on real data acquired at a synchrotron radiation facility. The data was acquired at the Bionanoprobe [9] at the 21-ID-D beamline of the Advanced Photon Source. The sample is a CMOS integrated circuit (IC) fabricated in a 65 nm technology with eight copper interconnect layers. This IC was imaged using a 140×300 Cartesian grid of scan points (30% overlap), 10 keV X-rays, and a Pilatus 300K detector (Dectris Inc.) with 619×487 pixels placed 2 meters downstream to collect the diffraction patterns. The central 256×256 pixels of the detector data were selected for the reconstruction, yielding a reconstructed image pixel size of 5.6 nm and 10.25 GB of raw input data. Such large datasets usually contain noise resulting from finite photon counts, positional errors in the scanning stages, fluctuations in the beam intensity, and distortions caused by air scattering, changes in sample temperature, and bad or missing detector pixels.

Figure 5 shows the 6.64 Terapixel reconstruction of the IC using $AD_{sync}$ running for 200 iterations on 32 GPUs, along with plots for estimated resolution and performance scaling up to 192 GPUs. Because phase contrast is much stronger than absorption contrast at the X-ray energy used, we show the phase contrast image, after phase unwrapping, from the ptychographic reconstruction of the IC complex transmission function.

5 Conclusion and Future Work

In this paper we presented a new parallel multi-GPU algorithm for ptychographic reconstruction based on automatic differentiation. The algorithm utilizes a production deep learning software library, TensorFlow, for adjusting an estimate of the ptychography forward model to fit with the measured data. In order to distribute the computation among multiple GPUs, our distributed algorithm splits raw input data into spatially contiguous partitions that are allocated to each GPU. Sub-problems are then either solved independently on each GPU or by communicating current solution estimates to other GPUs. The new algorithm was evaluated on synthetic and real data acquired at the APS. The synchronous version of the new algorithm was found to
Figure 5: Ptychographic reconstruction of a CMOS IC fabricated in 65 nm technology. (a) the phase retrieved for the IC with a wafer thickness of 300µm. In this case, the image shows an overlay of features at the chip wiring and gate level, along with variations in the overall wafer thickness which are presumably due to scratches on the surface of the wafer; (b) Performance plot for the synchronous and asynchronous algorithms, scaling up to 192 GPUs; (c) Line profile plot along the red line in (a) showing 22 nm resolution.

have superior convergence properties and scaling performance when compared to an existing implementation.

Thanks to decoupling the forward model from its optimization, one can easily amend the forward model, tune the optimizer hyperparameters, or experiment with different gradient-based optimizers and error metrics. Future work includes adding more input parameters to the current forward model, such as the probe function and experimental instabilities. Expanding the gradient calculation to these new parameters with the existing error metric, and comparing the results with various modifications to the error metric, is an interesting study to conduct. Additionally, we consider the work presented here a steppingstone towards a 3D ptychographic reconstruction method, owing to similarities between multilayer neural networks and the multislice propagation theory. Multislice techniques are being incorporated into many emerging 3D X-ray ptychographic methods [41,44].

Acknowledgment

We gratefully acknowledge the use of the resources of the Argonne Leadership Computing Facility and the Advanced Photon Source at Argonne National Laboratory. This work was supported by Advanced Scientific Computing Research, Office of Science, U.S. Department of Energy, under Contract DE-FC02-06ER25777. The Bionanoprobe is funded by NIH/NCRR High End Instrumentation (HEI) grant (1S10RR029272-01) as part of the American Recovery and Reinvestment Act (ARRA). Development of ptychography with the Bionanoprobe has been supported in part by NIH grant R01 GM104530. Use of the Advanced Photon Source,
an Office of Science User Facility operated for the U.S. Department of Energy (DOE) Office of Science by Argonne National Laboratory, was supported by the U.S. DOE under Contract No. DE-AC02-06CH11357.

References


[39] G. RW and W. Saxton. Phase determination from image and diffraction plane pictures in electron-


The submitted manuscript has been created by UChicago Argonne, LLC, Operator of Argonne National Laboratory (“Argonne”). Argonne, a U.S. Department of Energy Office of Science laboratory, is operated under Contract No. DE-AC02-06CH11357. The U.S. Government retains for itself, and others acting on its behalf, a paid-up nonexclusive, irrevocable worldwide license in said article to reproduce, prepare derivative works, distribute copies to the public, and perform publicly and display publicly, by or on behalf of the Government.