Computationally Expediting Fourier Ptychographic Microscopy

Vidya Ganapati and Alex Robey

May 14, 2018

Abstract

Fourier Ptychographic Microscopy (FPM) is a computational imaging technique that iteratively stitches together low-resolution, variably-lit images in Fourier space to produce a high-dimensional image. Unlike conventional imaging techniques, FPM substantially reduces the trade-off between field of view (FOV) and image resolution. But to overcome the limitations of an imaging system’s optics, FPM requires hundreds or thousands of low-resolution images. Not only is the process of collecting all of these images inefficient, but it also prevents researchers from using FPM to image biological processes that occur on short time scales. In this work, we introduce a deep learning architecture that computationally optimizes the illumination pattern used to generate the low-resolution images needed for high-resolution image reconstruction.

1 Introduction

In recent years, convolutional neural networks (CNNs) have been shown to be effective tools for image classification, object recognition and superresolution tasks. In particular, recent work in denoising corrupted images using CNNs ([12], [2], [7]) motivates the computational framework outlined in this paper. However, despite this recent progress in image analysis, there is relatively little research that considers the optical systems used to capture images in the first place. As opposed to the great number of projects focusing on learning from already-collected data, we focus on the following inverse problem:

Given a set of corresponding low-resolution and high-resolution images, is it possible to optimize the parameters of an imaging system to bypass the physical limitations of its optics?

Thus rather than training a computational network to recognize patterns or objects in images, we seek a system that can optimize how we capture images.

Fourier Ptychographic Microscopy (FPM) is a computational imaging technique that generates a high-resolution image by stitching together variably-lit, low-resolution images in Fourier space. These low-resolution images are measured by means of an LED array that sits directly below the sample. To generate a self-consistent, high-resolution image with a wide field of view (FOV), FPM requires thousands of low-resolution images, each of which corresponds to a unique illumination pattern. Capturing these images is time-consuming and prohibitive to the kinds of cells and processes that can be observed. Since many biological processes happen on fast time-scales, researchers are often unable to collect enough measurements during these incredibly short windows. Furthermore, the computation time needed to generate a super-resolution image using FPM with thousands of images takes is extremely large; reducing this computational time is therefore very desirable.

To expedite FPM, we propose a deep learning framework that computationally optimizes the illumination patterns used to measure the low-resolution intensity images needed for generating the high-resolution image. These optimal illumination patterns will enormously speed up FPM because they eliminate the need to collect thousands of low-resolution images and thus reduce the computation time incurred during the generative process.

Recent approaches to computationally optimizing the physical parameters of an optical system have yielded promising results. Notably, Roarke et al. introduced a convolutional learning...
framework that jointly seeks to learn the physical parameters of a microscope and to classify the captured images. Their “physical CNN” model classifies cells from microscopic images based on a simplistic learned illumination pattern [10]; this classification task marks an important step in evaluating the efficacy of an imaging system to exceed its physical limitations. However, their algorithms are primarily concerned with general classification rather than truly optimizing illumination corresponding to the FPM algorithm.

2 Background

2.1 Space-Bandwidth Product

There are several metrics for evaluating the performance of an optical or imaging system. In general, optical systems are limited in two ways: field of view (FOV) and resolution. FOV refers to the total observable area that can be viewed or captured by an optical device, whereas resolution measures the degree of detail with which an optical signal can be observed and stored (see Figure 1(a)). In general, designers of optical systems face a trade-off between FOV and resolution; this motivates the development of algorithms like FPM to exceed the limitations of a given optical system.

A third metric, known as the space-bandwidth product (SBP), is used to quantify the combination of the FOV and resolution of an optical system. Formally, the SBP of such a system is defined as the number of degrees of freedom that can be extracted from an optical signal. In essence, SBP measures the number of pixels needed to capture the full field of view at full resolution. SBP is a useful metric for evaluating the efficacy of FPM because it measures the amount of information transmitted and stored in an imaging system. To evaluate the performance of an algorithm like FPM, one need only compare the SBP of the computational results to that of the images taken from an unenhanced microscope [15].

A high SBP is very desirable. Entire fields rely on the ability to accurately image biological samples using enhanced optical systems. In particular, numerous biomedical and neuroscience applications are reliant on being able to quickly and efficiently image large numbers of biological samples for analysis. Most off-the-shelf image systems specify SBPs in the tens of megapixels. Computational imaging algorithms like FPM have become popular because they are capable of bypassing the physical limitations of optical systems to achieve gigapixel SBPs.
2.2 Optical Aberrations and the Numerical Aperture

The achievable SBP of imaging systems is inherently limited by the physical qualities of its optical elements. The most fundamental source of such limitations is introduced by what is known as the numerical aperture. The numerical aperture (NA) of an optical system characterizes the range of angles over which light can pass through the circular objective (see Figure 1(b)). The NA is a dimensionless quantity and depends only on the refractive index of the medium through which light is passing. Much of the light refracted through the sample misses the objective, and so only the spatial components within the passband actually reach the image plane; this step in the imaging process has the same impact as applying a low-pass filter. This filtering inherently limits the resolution of the image recorded by the optical system. In general, many models of microscopic imaging systems consider the NA to be the only limiting factor for achievable resolution.

There are other notable limiting factors for achievable image SBP. When a sample is imaged using an optical system such as the one depicted in Figure 1(a), light reflected from the sample passes through and is magnified by a lens or series of lenses. Ideally, all incident rays to a lens would be reflected through the focal point. However, in reality incident rays far from the center of the lens miss the focal point, as in Figure 1(b). This causes blurring and distortion, and thus limits the SBP of an optical system. In general, any deviation from the ideal model of incident light rays reflecting through the focal point are called optical aberrations.

There are several ways to correct for optical aberrations. Previous work has involved artificially increasing SBP by employing mechanical mechanisms to align and filter light passing through optical lens. Not only is this solution expensive, but it also accept the upper bound of an ideal SBP. On the other hand, computational techniques for bypassing optical aberrations promise gigapixel SBPs that are far beyond the theoretical limit of an imaging systems optics. In particular, it has already been demonstrated that PPM can extend both the FOV and resolution beyond what would be physically attainable even in an ideal scenario [8].

2.3 PFM Assumptions

To computationally construct a super-resolution image by means of PPM, we make several simplifying assumptions. Firstly, we model the illuminated sample as a pure phase-object, meaning that the light emitted from the LED array passes through and is refracted by the sample; however, we assume that the light is not magnified (i.e. that the amplitude of light waves passing through the sample is not altered). This is illustrated in Figure 2(a). This assumption disregards changes in the intensity of the plane waves traveling through the sample. In reality, this is not entirely the
case, but it is sufficiently close so that it does not hinder the generative process. In this way, we generate a complex-valued high-resolution image, rather than a pure intensity image. Therefore, while generating the high-intensity image from a set of low-resolution phase images, we train our algorithm with the goal of optimizing away the real portion of the complex image so that only the phase information remains. Thus we seek a mapping from many low-resolution pure phase images to one high-resolution phase image.

Secondly, we assume that the illuminated LEDs are mutually incoherent. That is, the plane waves emanating from a given LED do not interfere with the plane waves from any other LED in the grid. In the computational model for FPM, it is necessary to multiply each plane wave illuminating the sample by the complex-valued image, which translates to convolving the Fourier-transformed image by the delta-Dirac function in the frequency domain. Since the LEDs are modeled as mutually incoherent, we can simply perform this computation individually for each LED in the grid. This is useful because we disregard the more complicated and computationally expensive modeling of plane wave interference.

2.4 Fourier Ptychography Algorithm

At a high level, the goal of FPM is to reconstruct a high-resolution image by stitching together low-resolution images in the Fourier domain. To facilitate this process, a two-dimensional array of LEDs is placed directly underneath the sample, which is in turn placed at the focal plane of a low-NA microscope objective as in Figure 3(a). To generate the low-resolution images, a two-dimensional sample is successively illuminated by plane waves; each illumination pattern corresponds to a different low-resolution image. It is assumed that each LED is incoherent with its neighbors.

Formally, the goal of FPM is to generate a single high-resolution image \( I_h \) from \( N \) low-resolution measurements \( I_m \). Throughout this section, the subscripts \( h, l, \) and \( m \) will denote high-resolution, low-resolution, and measurement, respectively. For the steps that follow, we introduce the function \( \zeta : \mathbb{R}^2 \to \mathbb{C} \) mapping real-valued intensity \( I \) and phase \( \varphi \) predictions or measurements to a complex image so that \( \zeta(I, \varphi) = \sqrt{I}e^{i\varphi} \).

2.4.1 Initialization

1. The first step of the FPM method is to guess the spatial-domain high-resolution object function \( \zeta(I_h, \varphi_h) \). As arguments, this function takes the predicted intensity \( I_h \) and predicted phase \( \varphi_h \) of a high-resolution image. After making this guess, the image \( \zeta(I_h, \varphi_h) \) is passed into the Fourier domain via the Fourier transform. Because the initialization is usually very blurry and distorted, the Fourier transform of \( \zeta(I_h, \varphi_h) \) creates a broad spectrum in the Fourier domain.

2.4.2 Iteration

2. In the first step of the iterative recovery process, the Fourier transformation is applied to a small subregion of the spectrum created in Step 1 to generate a new low-resolution image \( \zeta(I_l, \varphi_l) \). Ultimately, the aim is to replace this subregion with a higher-resolution patch based on the measurements taken from the setup shown in Figure 3(b). In the following steps, this generated low-resolution image \( \zeta(I_l, \varphi_l) \) is compared to the measured low-resolution images in an effort to most-accurately reconstruct a high-resolution image.

3. In this step, the goal is to relate the measurements to the complex image prediction generated in Steps 1 and 2. In brief, this amounts to two processes: first replacing \( I_l \) by \( I_m \), the low-resolution measured intensity, in the image object function \( \zeta(I_l, \varphi_l) \); second, the phase of \( \zeta(I_l, \varphi_l) \) is recovered to more closely match the phase of the set of low-resolution images using the phase retrieval algorithm described by Fienup et al [6]. This image function is then moved via the Fourier transform back to the Fourier domain, where it replaces the subregion extracted from \( \zeta(I_h, \varphi_h) \).

4. Steps 2 and 3 are now repeated on different subregions of the spectrum created in Step 1. Each subregion corresponds to a different low-resolution measurement. This process is repeated for all \( N \) low-resolution measurements. In this way, the measurement information is propagated through and pervasive in the generated high-resolution complex image.
Figure 3: The unique feature of the FPM experimental setup is the array of LEDs that illuminate a sample.

2.4.3 Recovery

5. In the final step, we repeat Steps 2-4 until the high-resolution image $\zeta(I_h, \varphi_h)$ is self-consistent. Finally, $\zeta(I_h, \varphi_h)$ is moved back into the spatial domain. This generated image has a much higher SBP than any of the low-resolution measurement images.

2.5 Neural Networks

Generally speaking, artificial neural networks (ANNs) are an interconnection of nodes in a directed graph that loosely models biological neural networks found in the brain. Biological neural networks have three components: receptors, neurons and effectors. Each component has a unique purpose in the brain’s decision making process. In particular, receptors are responsible for recognizing and converting signals from the outside world into electrical impulses. These impulses are in turn passed to a network of neurons. Each neuron receives signals from other neurons in the graph; individually, each neuron is responsible for processing the electrical signals and determining whether or not the signal should be passed on to the next set of neurons. The final neurons in a biological network feed into the effectors, which translate the outputted electrical impulses into responses to an organism’s environment [13].

ANNs attempt to model the brain’s biological process for responding to stimuli. In the last decade, computer scientists have developed and refined methodologies for creating a computational model of how neurons decide how to process an electrical impulse. As in biological neural networks, each artificial neuron receives receives signals from other neurons in the graph, processes these signals and then sends them on to other neurons. The collective set of signals received by the last layer of the graph determines the prediction or output of the neural network; this is analogous to the work done by the effectors in a biological neural network.
The methodology for determining how artificial neurons should respond to a given input signal has been the subject of much study in recent years. In general, the goal of building a neural network is to create a machine that can make a meaningful prediction or perform a classification task on some set of input data. This could involve determining what’s depicted in an image or classifying whether an email is spam or not. To teach an ANN how to make these predictions, the neurons in an ANN undergo a “training” phase in which the network “learns” from a set of input data. This involves feeding a set of input signals into an ANN in which the neurons have been initialized with weights that determine how they will respond to a given signal. As signals are passed into an ANN, the weights on each node are updated by comparing the expected output to the prediction. This difference, often referred to as loss, is propagated back through the ANN in such a way as to probabilistically increase the likelihood that the network makes a better prediction during the next iteration.

2.6 Convolutional Neural Networks

Convolutional neural networks (CNNs) are a subset of ANNs specifically designed to handle image data. In particular, CNNs are often used for classifying the characteristics of an image or for identifying particular objects or features of images [1]. The fundamental idea is that by moving variably sized filters across an image, a CNN can identify unifying patterns in a given set of images; based on these patterns identified during the training phase, CNNs are able to classify the contents of images that it has never processed or seen.

CNNs have also been successfully used for image denoising and superresolution tasks ([12], [2], [7]). The fundamental idea is that a CNN is taught to predict the pixel values of a high-resolution image based on noisy or corrupted input image data. By comparing the predicted pixel values to the values of an image that has already been denoised, it is possible to create a machine that learns to remove noise and distortion from images. This technique is used extensively and described in Section 3.

3 Computational Framework

3.1 Datasets

To learn the optimal illumination pattern, the CNN architecture proposed in this paper maps complex-valued low-resolution images to complex-valued high-resolution images. We chose to test our network on two datasets: the MNIST dataset of handwritten images [17] and a dataset of phase contrast mus musculus cells [14]. Because both of these datasets have real-valued pixel intensities, we multiplied both by a complex exponential to convert them to complex-valued images. We choose to model cells as pure-phase images; in reality, this assumption means that a cell refracts all of the light passing through it. In this way, we transform each image to a complex-valued discrete scalar field in which the real component of each pixel takes on value zero. This agrees with the assumptions described in Section 2.3.

The MNIST dataset is one of the most commonly-used datasets in all of machine learning and computer vision. The images in MNIST are gray-scale 28 x 28 pixel image of handwritten digits. This dataset contains 55,000 training examples, 20,000 validation examples and 10,000 testing examples. We used this dataset to tune the architecture described in Section 3 before testing on the mus musculus dataset, which contains images of cells similar to those that we would image in practice.

Before testing each dataset on our proposed architecture, we transform each picture in several preprocessing steps. In particular, we specify the radius of each LED and the number of total LEDs, which is used to compute the theoretical low-resolution images corresponding to the images in the dataset. These parameters also impact the number of illumination patterns and the achievable NA of the optical system.

3.2 Computational Network

Our network architecture was inspired by the skip-connection convolutional autoencoder described in [16]. Fundamentally, convolutional neural networks excel at extracting features and learning
patterns from images. That is, by applying linear and nonlinear filtering operations to the pixels of a set of images, CNNs are exceedingly effective at characterizing patterns.

We pose the image restoration problem in the following way. Given a corrupted, low-resolution input grayscale image $X_i$, we seek the corresponding high-resolution $n \times m$ pixel image $Y$. The mapping can be described by

$$Y = f(X_i^u) + X_i^u$$

where $X_i^u$ represents an upsampled corrupted low-resolution image, and the function $f : \mathbb{R}^{n \times m} \rightarrow \mathbb{R}^{n \times m}$ represents the computation performed by the CNN. In this way, this network is intended to extract the missing information from the input image in order to restore and enhance the resolution of $X_i$.

The natural penalty often incurred in image restoration or superresolution tasks is a trade-off between preserving fine details in $X_i$ and eliminating low-level corruption. To resolve this issue, the authors of [16] proposed a network that makes use of convolutional and deconvolutional filtering layers. While convolutional layers can be thought of as encoding and extracting the main components and features of an input image, deconvolutional layers perform a complimentary task. That is, deconvolutional layers map from one pixel in an input layer to several pixels in an output layer; hence deconvolution operations are useful for recovering image detail and low-level content. The combination of encoding convolutional layers and decoding deconvolutional layers are thought to provide an effective architecture for recovering primary features and fine details, respectively.

The basis of our architecture relies on this idea of first extracting useful features using convolutional layers and then decoding fine details using deconvolutional layers.

A network that utilizes only convolutional and deconvolutional layers is necessary, but not sufficient for this task. While CNNs can effectively learn mappings that abstract features from input images, they are often maladroit at reconstructing convincing denoised images. In general, convolutional operations are not lossless; information about an input image is inevitably discarded at each layer of the network. To address this problem, the authors of [16] also propose the use of residual connections that add the input image back into the filtered abstractions learned from a set of input images. These residual connections facilitate the passing of abstractions from convolution layers directly to the corresponding deconvolutional layers. An overview of this architecture is shown in Figure 5.

The symmetric skip connections provide enhance the convolutional-deconvolutional architecture enormously. Not only does this improvement enhance the restoration of image details; it also increases the network depth and provides shortcuts between the output and input, which makes the network easier to train.
Our network utilizes multiple blocks of the module shown in Figure 5. Additionally, our network uses batch normalization and dropout layers to improve the robustness of the image restoration task. Batch normalization involves normalizing the inputs into each layer of an ANN such that the activation of each artificial has zero mean and a standard deviation of one. This has been shown to be useful for improving the performance of the training phase for ANNs [11]. Dropout simply involves erasing or resetting a small percentage of the weights learned during training. This improves the robustness of neural networks to over-fitting.

3.3 Loss Functions

As described in 2.5, when training a neural network, the weights of each artificial neuron are updated according to an iteratively calculated loss. In this case, the loss represents the difference between the generated high-resolution image and the ground-truth image from the dataset. To calculate this loss, we experimented with several different difference metrics. In [4], Hang Zhou et al. suggest the use of the structural similarity index (SSIM [5]) and multi-scale structural similarity index (MS-SSIM [18]) loss functions for image denoising and super-resolution. In theory, these functions represent the state-of-the-art in image comparison; however, we found that these functions were not as effective as the simple $L_1$ metric. The $L_1$ metric is simply the pixel-wise absolute difference between the generated and ground-truth images. In the end, we found that the mean-squared error error (MSE) metric outperformed the SSIM, MS-SSIM and $L_1$ metrics. The MSE of a set of $n$ images $\{X_i\}$ is given by

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (X_i - \bar{X})^2$$

where $\bar{X}$ is the mean image.

3.4 LED Array Optimization

During the training process, the intensities of the illumination LEDs are updated by the errors that are back-propagated through the CNN described above. Thus during each training step, the network attempts to computationally learn the most-informative LED pattern such that the corresponding low-resolution images most effectively reconstruct the high-resolution image.

There are two distinct issues that complicate convolutional image denoising and optimization of the LED array. Firstly, as we have already pointed out, convolutional layers tend to abstract away crucial details of an image, which greatly complicates the process of denoising. Also, we found that many purely convolutional/deconvolutional networks suffered from vanishing gradients. That is, the trainable variables in the LED array optimization process converged to zero, which is clearly a suboptimal result. Fortunately, introducing the residual skip-connections eliminated this problem, because the gradient was propagated back to the bottom layers of the network without passing through all the layers above it.
Figure 6: Two generated high-resolution images and two examples of optimal LED illumination patterns generated using the residual CNN described in Section 3.

Figure 7: Two optimized LED illumination patterns generated by the residual CNN described in Section 3.

4 Results

All experiments described in this experiment were run on the XSEDE GPU clusters XStream and Comet. The final version of the modified residual CNN described above was run with 20,000 epochs and a batch size of one image. Due to the cost of computing the Fourier transform over batch sizes exceeding one image, we did not experiment with larger batches.

Training was conducted in two phases: a static training phase in which the LED intensities were kept fixed and a dynamic training phase in which the LED intensities were updated by the backpropagated gradients. This alternating training scheme achieved sharper image reconstructions. For dynamic and static phases of 10,000 epochs, training times reached nearly six hours.

Two reconstructions of MNIST digits are shown in Figure 6. These images were reconstructed...
using two residual CNN blocks. Figure 7 shows the optimized LED patterns for these reconstructions. Figure 8 shows a generated cell phase object.

5 Discussion

In this paper, we have presented the high-resolution reconstructions and optimized illumination patterns generated by our residual CNN architecture. Although the results are not as sharp and focused as we would like, an important note is that these results have been achieved using only one illumination pattern. In contrast, FPM requires thousands of patterns to generate a single high-resolution image. Our experiments indicate that if we were to extend our algorithms to learn on the order of ten illumination patterns, image clarity could be drastically improved.

The result of the MNIST digit generation are highly encouraging. Although the results are not as sharp as the original images, they represent a relatively close match for images of this size. That is, because there are so few pixels in these images, small differences appear magnified and stand out. Therefore when we tested the network on the larger 100 × 100 pixel cell images, the generated phase objects looked much more similar to the original training images. Without the annotations on Figure 8(d), it can be hard to find differences with Figure 8(c).

We also observe two interesting properties of the optimized LED illumination patterns shown in Figure 7. First, the patterns are highly asymmetrical. That is, there is very little regularity or pattern in which LEDs are on or off. This is likely due to the fact that LEDs in symmetrical positions in the LED array provide similar or redundant information needed for the generative
process. Therefore, the neural network suppresses the redundant LEDs in the optimization process. The second observation is that LEDs around the outside of the matrix are more intense than LEDs closer to the middle. This implies that LEDs around the outside of the array provide more information for reconstruction than LEDs near the middle.

6 Conclusions and Future Work

We are still tweaking our network architecture. The parameter space for network parameters is very large. In particular, we can change the learning rate or step size of the training phase, the number of epochs during training, the image segmentation size, the number of training and testing examples, the number of LEDs in the illumination array and the number of CNN blocks. We are also considering alternative gradient update schemes, such as the method described in [3] which uses Wirtinger derivatives.

We are also working with other datasets. The results presented for *Mus musculus* are preliminary; there are an abundance of tests and datasets that we would like to test on our network. After further investigating the performance of our network on cellular data, we will attempt to test these optimized patterns on a real microscope. In the coming months, we will be setting up such a system at Swarthmore College. Over the course of this semester, a group of students at Swarthmore College have been building an experimental setup similar to the apparatus shown in Figure 3(b). Given a kind of cell to be imaged, we hope to use the computational approach outlined in this paper to learn the optimal illumination pattern to be used for FPM. We will compare the high-resolution results of images generated from our optimized illumination patterns against results from classical FPM. We expect that our results will compare favorably with existing results.
References


