# Efficient Algorithms for Solving Nonlinear Inverse Problems in Image Reconstruction

Submitted in partial fulfillment of the requirements for
the degree of
Doctor of Philosophy
in
Electrical and Computer Engineering

**Vincent J. Monardo**

B.S.E., Electrical Engineering, Arizona State University

Carnegie Mellon University
Pittsburgh, PA

March 2022

# Acknowledgments

This thesis is supported in part by Office of Naval Research, Army Research Office, National Science Foundation, the D3OM2S Center of Excellence, and the Carnegie Institute of Technology Deans Fellowship.

I would like to thank my thesis advisor and committee chair, Dr. Yuejie Chi. Yuejie's passion for research is continually inspirational; even more so is her commitment to mentoring and empowering her students and others around her. While my PhD journey has been characterized by a period of personal growth, there have also been many periods of personal hardship. Without her guidance and help, this dissertation would not have been possible. Additionally, I would like to thank my fellow research group members who made learning an enjoyable environment: Dr. Liming Wang, Dr. Myung Cho, Dr. Yuanxin Li, Dr. Haoyu Fu, Dr. Maxime Ferreira Da Costa, Dr. Harlin Lee, Dr. Tian Tong, Boyue Li, Laixi Shi, Shicong Cen, Diogo Cardoso, Pedro Valdeira, Jiin Woo, Harry Dong and Lingjing Kong.

I also would like to thank Dr. Aswin Sankaranarayanan, Dr. Marc De Graef, and Dr. Charles Bouman for serving as committee members for this thesis. Their feedback and valuable insights helped shaped the content of this thesis.

I am thankful for the many collaborations that I have been able to be a part of during my PhD. A special shout goes out to Abhiram Iyer. Watching Abhi grow as a researcher has been delightful, and I look forward to seeing what is in store for his future.

Providing a complete list of all the friends who supported me throughout my thesis would be impossible–I will list a few. Thank you Dr. Anit Sahu, Dr. Jonathan Mei and Dr. Evgenvy Toropov for immediately taking me under their wing when I first entered the Porter Hall B(asement) at CMU. Thank you, Stephen Bergauer, Sam Singh, Dr. Jonathan Zia, Peter Williams, Dr. Anna Mian, Dr. Sami Mian, Calin Solomon, Dr. Davis Gilton, Dr. Milda Zizyte, Dr. Mary Story, Dr. Maxwell Li, Dr. Joseph Tessmer, Dr. Andrew Kitahara, Dr. Sneha Gaitonde, Dr. Ian Chesser, Patience Stevens, Aayushya Agarwal, Rajshekar Das, Dr. Hae Sun La, Simin Liu, Sae Seul Park, Dr. Jon Francis, David Francis, Hane Lee, Jonah Ko, Dr. Natalie Lao, Dr. Nicholas Kwok and many, many more people (and their pets).

Since the year 2020, the entire world has been, frankly, absurd. A special thank-you goes out to Harlin Lee and Carmel Fiscko. Being able to spend time with them on a nearly daily basis during the pandemic served as a grounding force in a hectic world. I would also like to thank all the health professionals who helped me and the front-line members who helped to manage the COVID-19 pandemic. I am especially grateful to Dr. Kym Jordan Simmons, Dr. Susan Morine and Barb Benedict, for helping me navigate my health struggles. Finally, I'd like to thank my family for being my biggest fans through the years. Thank you all.

--Vincent Monardo

*For my family.*

# Abstract

Throughout many fields of science, images are used to display information in a relatable matter about objects which may not be directly visible. Instead, sensors are used–beyond standard cameras–to capture measurements of the object. These measurements are then processed to reconstruct an image of the object which created them. This process of reconstructing the cause of the observed effects is known as an inverse problem. In this thesis, algorithms are proposed for solving various inverse problems in image reconstruction. These algorithms are then analyzed to demonstrate their statistical and computational efficiency. The main through-line tying these problems together is that the proposed solutions leverage inherent structural information.

The thesis begins by demonstrating how to design effective spectral methods for estimating an image from phaseless measurements given approximate knowledge of the structure of the noise effecting the system. Next, an amplitude-based loss function is proposed for solving a generalized matrix phaseless sensing problem and algorithms are derived which reach a critical point of such a loss function. Continuing, stochastic variance-reduction gradient techniques are applied to an algorithmic framework for reconstructing an image known as Plug-and-Play (PnP) to achieve faster computation times while maintaining high accuracy. The thesis closes by analyzing how to reconstruct a large set of images when there exists a global structure relating the images to each other. The methods presented in this thesis are applied to phase retrieval, compressive sensing magnetic resonance imaging, and electron back-scattered diffraction microscopy.

# Contents

# List of Figures

# Chapter 1

# Introduction

This thesis ties together a breadth of topics applicable to inverse problems in imaging, rather than tackling many challenges within a single imaging modality. In this chapter, we cover some general background for understanding the work. Then, there are four chapters which investigate inverse problems in image reconstruction. In general, each piece of work within this thesis is tied together by a few core principles. First, we are interested in deriving algorithms which are efficient in the sense that they do not require an extraneous amount of measurements or time for arriving at accurate solutions. Secondly, we are interested in incorporating tools from statistics and mathematics at large for making the most out of the finite resources available. These are ways to tie these problems together beyond simply applying them to imaging inverse problems. Indeed, the methodology presented in this thesis is generalizable beyond imaging to inverse problems as a whole.

## 1.1 On Forward and Inverse Problems in Imaging

Scientists have extensively attempted to create mathematical models of different phenomena in the world. With these models in hand, one can understand the cause and effect behind how the world works. For example, from Newton's laws of universal gravitation, one can calculate the gravitational force between two objects by measuring their mass and the distance between them

[79]. This exemplifies the structure of a *forward problem*: given the state of the system from the physical parameters that describe it (e.g. the mass of the two objects), we can predict the behavior that we can observe (e.g. the gravitational force between the two objects).

The work in this thesis is concerned with solving *inverse problems*, which are the exact opposite. An inverse problem in science is the process of calculating the factors which caused a set of observations; the problem starts with the effects and then calculates the causes. Inverse problems are some of the most important mathematical problems in science and engineering because they tell us about parameters that we cannot directly observe. Related to the above, one may wish to find the density of the Earth by comparing its volume to its weight. While we cannot simply place the Earth on a scale, one may estimate the weight of the earth from the strength of its gravitational fields.

This thesis is concerned with reconstructing an image from a set of measurements, i.e., finding an image which caused the measurements (or effect) that one observes, in the same formulation as an inverse problem described above. For an overview of various inverse problems in imaging, see [9]. It should be emphasized that this work specifically does not propose new imaging techniques, camera designs, or ways to utilize a certain imaging apparatus. Rather, this work is focused on taking the measurements from an imaging system and reconstructing an image that represents the object being imaged by the system. This is to say that we are focused on inverse problems, not forward problems, which have already been (and, rightfully, continue to be) studied within physics and other related disciplines.

While we typically think of capturing images with a handheld camera, the work of this thesis is predominantly interested in reconstructing images of objects that cannot be directly observed. Perhaps the object of interest is too small to be seen directly, or is occluded to the naked eye. Nonetheless, understanding the structure visually may be useful for a trained professional to draw exact conclusions. For example, a doctor is able to diagnose the presence of a tumor or a bone fracture without an intrusive procedure to witness the abnormality directly. Rather, non-intrusive, sophisticated imaging techniques can capture unobservable regions at a high enough resolution to

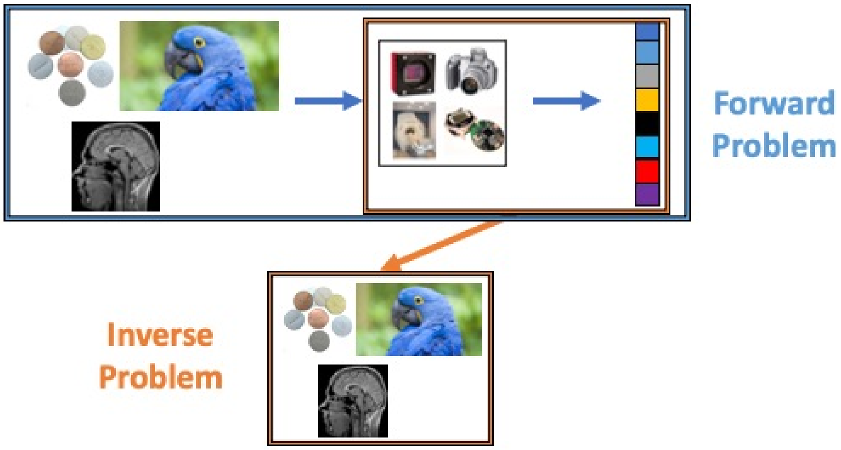


Figure 1.1: A visualization of the forward and inverse imaging problem “pipeline.” The blue box denotes the forward problem, i.e. how the measurements are created. The orange box denotes the inverse problem, i.e. reconstructing an image given the forward model and a set of measurements.

be conclusive.

In the recent past, techniques such as X-rays, ultrasounds, CT scans, and many others have become common practices for medical diagnostics. As a case study, consider Magnetic Resonance Imaging (MRI). MRI is a procedure which relies on magnetic fields and radio frequency waves in order to produce detailed images of organs and bodily structures [25]. MRIs are novel—in contrast to the imaging techniques listed above—in that they do not rely on forms of radiation which may harm the patient if the exposure time is too long. This means that more time can be spent to capture an MRI, which may be desirable (limitations are to be discussed shortly). However, as we are capturing information that is not seen by the naked eye, we need to reconstruct (the image of) the object’s structure which produced the measurements of magnetic resonance that is observed at the MRI scanner.

## 1.2 Understanding Complexity and Efficiency

A common motif of inverse problems in image reconstruction is that the accuracy and resolution of the reconstructed image is proportional to the resources invested into collecting and processing the measurements. However, many imaging systems may be prohibitively expensive—in terms of monetary cost or time—such that a less-than-ideal number of measurements are taken. Nonetheless, one still desires to reconstruct a high fidelity image. Recalling MRI, the scanning process is generally perceived as uncomfortable by patients. The scanning instrument can be quite noisy, and the patient must lay still within a tube, resulting in a claustrophobic experience. This can be especially difficult for children, for example, and relying on them to remain still for the entire duration becomes unreasonable. A scan may take upwards of 90 minutes to capture, depending on the region's size and the number of measurements required to be taken [100]. Therefore, we would like to minimize the number of measurements needed to produce a quality image.

Keeping the above relationship between cost and accuracy in mind, this thesis explores algorithms and procedures for reconstructing accurate images within certain constraints. We will now outline the background information necessary to quantify how efficient a method is in order to adequately compare the proposed method to the current state-of-the-art. We propose data- and physics-driven solutions for processing the measurements captured from a sensing or imaging device in order to reconstruct an image. With this in mind, we propose algorithms based upon optimization and statistical modeling that improve upon this sense of efficiency.

For a given inverse problem, we seek ways of comparing algorithms in order to determine which one is more efficient. Suppose that we seek an algorithm which is able to reconstruct an image with an estimate up to a certain level of accuracy. Building off the intuition provided in the Section 1.1, there are two traits of an algorithm that are desirable. We formally define two notions of complexity necessary for understanding the efficiency of an algorithm.

**Definition 1.2.1** (Computational Complexity). *The computational complexity of an algorithm is the amount of time required to generate an estimate which reconstructs the image with some measured accuracy.*

**Definition 1.2.2** (Statistical Complexity)**.** *The statistical complexity of an algorithm is the amount of measurements required (often as a ratio compared to the dimensions of the image) to generate an estimate which reconstructs the image with some measured accuracy.*

We are interested in algorithms which obtain an accurate image with minimal computational and statistical complexity. In general, there are two key metrics which will be used to determine the accuracy to which the estimate $\hat{\boldsymbol{x}}$ reconstructs the ground truth image $\boldsymbol{x}^\star$. The first metric is concerned with directly comparing the two objects. A common choice is to utilize a distance metric such as the $\ell_2$ norm difference between the two objects. This is related to the well-known metric Mean Squared Error (MSE), defined as,

$$\text{MSE} = \|\boldsymbol{x}^\star - \hat{\boldsymbol{x}}\|_2^2. \tag{1.1}$$

Additionally interested is the Normalized MSE (NMSE), i.e., the quotient of the MSE and the squared $\ell_2$ norm of the ground truth image. For determining the accuracy in an image reconstruction problem, one key metric of interest is known as the Peak Signal-to-Noise Ratio (PSNR) defined (in decibels) as follows,

$$\text{PSNR} = -10 \cdot \log_{10}(\text{MSE})\,\text{dB}. \tag{1.2}$$

Note that this definition of PSNR makes the assumption that the maximum index value possible is 1. Generally speaking, a higher PSNR indicates that the reconstruction is of higher quality.

## 1.3 Solving Nonlinear Inverse Problems

When looking to solve an inverse problem in imaging, we will universally adopt the following framework:

*Given measurements and a forward model known a priori, reconstruct an image of interest, by leveraging domain knowledge when available.*

Formally, we are given a set of measurements $m$ measurements, $\{y_i\}_{i=1}^m$, where $y_i \in \mathbb{C}, \forall i \in [m]$. Each measurement $y_i$ is observed through a known forward model, $\mathcal{A}_i : \mathbb{R}^n \mapsto \mathbb{C}$. Additionally, each measurement is assumed to be effected by some noise, $\{\varepsilon_i\}_{i=1}^m$, where $\varepsilon_i \in \mathbb{C}, \forall i \in [m]$. Thus, our goal is to reconstruct an image of interest with $n$ pixels, $\boldsymbol{x}^\star \in \mathbb{R}^n$, according to the signal model,

$$y_i = \mathcal{A}_i(\boldsymbol{x}^\star) + \varepsilon_i, \text{ for all } i = 1, \cdots, m. \tag{1.3}$$

Alternatively, it is often convenient to pose the signal model in matrix form. Let $\boldsymbol{y} = [y_1, \cdots, y_m]^\top$, $\mathcal{A} : \mathbb{R}^n \mapsto \mathbb{C}^m$ represent the set of forward models for each measurement, and $\boldsymbol{\varepsilon} = [\varepsilon_1, \cdots, \varepsilon_m]^\top$. Then, we can adopt the signal model,

$$\boldsymbol{y} = \mathcal{A}(\boldsymbol{x}^\star) + \boldsymbol{\varepsilon}. \tag{1.4}$$

Thus, the goal to is to reconstruct the image, $\boldsymbol{x}^\star$, from the set of measurements, $\boldsymbol{y}$, using our knowledge of the structure of the forward model, $\mathcal{A}$.

### 1.3.1 Leveraging Structures for Improving Efficiency

As alluded to earlier, we often operate in non-ideal regimes when solving inverse problems in imaging. Consider a simple example, where the forward model is a linear mapping. Basic intuition from linear algebra tells us that in order to solve a system of equations with $n$ unknowns, we need at least $n$ equations. This corresponds to a regime where the number of measurements is at least equal to the number of pixels within the image. However, the expectation to receive as many measurements is often ludicrous in practice. This leads us to look for deeper mathematical structures within the data, sensing models, and measurements that we can leverage when designing and implementing algorithms for image reconstruction.

Indeed, one can look towards the field of compressive sensing [17, 35] for an excellent example of leveraging structural information to solve inverse problems in under-sampled regimes. In a nutshell, consider a scenario where a general $n$-dimensional object can be accurately represented with a smaller number of dimensions $s \ll n$. When this is the case, one can leverage this information to accurately reconstruct the object of interest in a regime where the number of measurements is less than $n$ by utilizing knowledge of the latent $s$-dimensional structure. This

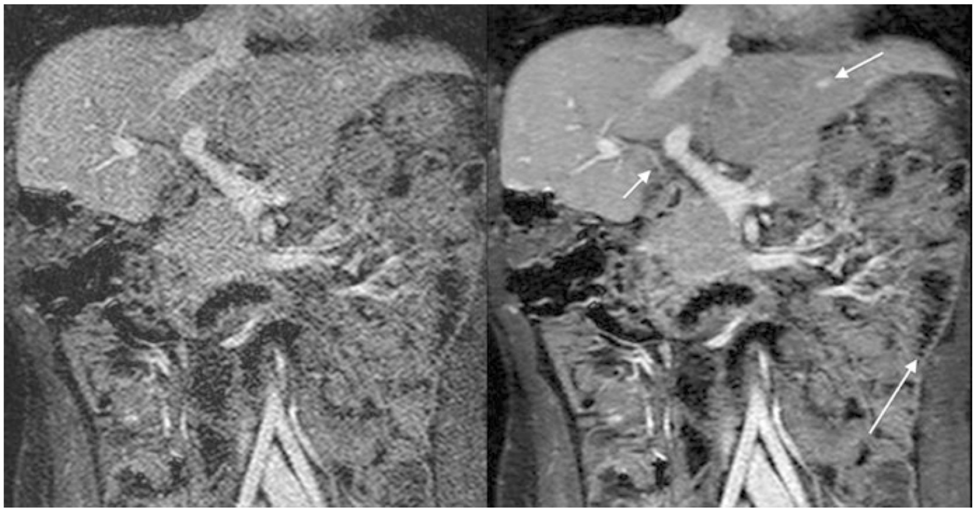


Figure 1.2: Visualizing impact of exploiting lower dimensional structure in reconstructing MRI scans. On the left, is an example of a reconstructed image when a naive linear regression method is used. On the right, is the result when sparsity is leveraged during reconstruction. Fine details are identified with white arrows. Figure credit: [41].

led to the development of many algorithms which incorporate a regularization function within our optimization formulation to exploit this lower-dimensional, "sparse," structure. Motivated by this train of thought, the algorithms proposed within this thesis will look to exploit other familiar notions of structure within the problem formulation. An example of exploiting lower dimensional structure for reconstructing MRI images is shown in Figure 1.2.

### 1.3.2 An Optimization Approach

With a signal model in hand, we can define an optimization problem for reconstructing an image from an inverse problem, bearing in mind our goal of enforcing structure within the reconstructed estimate which is consistent with our domain knowledge. Dependent on the forward model, one can handcraft a function $\ell : \mathbb{R}^n \mapsto \mathbb{R}$ which quantifies the efficacy of an estimated image based upon the system model and measurements received. Additionally, we can tailor a function $g : \mathbb{R}^n \mapsto \mathbb{R}$ which captures how consistent our estimate is with our assumed structural knowledge.

Ideally, our goal would be to find an image $\hat{\boldsymbol{x}} \in \mathbb{R}^n$ such that $\ell(\hat{\boldsymbol{x}}) \leq \ell(\boldsymbol{x})$ and $g(\hat{\boldsymbol{x}}) \leq g(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathbb{R}^n$. In reality, satisfying both functions $\ell$ and $g$ is typically an unreasonable goal, and we look to reconstruct an image which balances the requirements of both functions. In short, we aim to solve the problem,

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x} \in \mathbb{R}^n} \{\ell(\boldsymbol{x}) + \lambda g(\boldsymbol{x})\}, \tag{1.5}$$

where $\lambda > 0$ is a non-negative real number which dictates the amount of regularization enforced.

While many algorithms and procedures exist for solving optimization problems of the form (1.5), a common choice which is studied in this thesis is to utilize gradient-based methods [84]. To start, consider the function $\ell$ on its own. Our goal is to find a critical point of the function $\ell$ which is more specifically a (globally) minimum point. It is well-known in calculus that the gradient at a critical point of a function is equal to $\mathbf{0}$. Additionally, at any given point $\boldsymbol{x}$, the gradient of the function will provide the direction of the steepest descent towards a critical point. Given a function $\ell$, we can carefully choose some initial point $\boldsymbol{x}_0$ and calculate the gradient at that point and update our estimate by moving along the gradient to a new point. We iteratively update our estimate a fixed number of times to obtain our final estimate $\hat{\boldsymbol{x}}$. A brief description of gradient descent follows as,

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \eta \nabla \ell(\boldsymbol{x}_t), \quad t = 0, \cdots, T. \tag{1.6}$$

Here, $\nabla \ell(\boldsymbol{x}_t)$ is the gradient of the function $\ell$ at the current iterate $\boldsymbol{x}_t$, and $\eta > 0$ is a non-negative number known as the learning rate, which determines how far we travel in the direction determined by the gradient.

Bringing $g$ into play, additional measures are necessary to ensure that the estimate is consistent with both the observed measurements *and* our domain knowledge. One common choice is to consider a proximal operation [7]. On an intuitive level, the goal of a proximal operator is to find an estimate which minimizes some function—e.g., $g$—that is also within the proximity of the input estimate. Mathematically, the proximal operator is defined as an optimization problem itself, as,

$$\operatorname{prox}_{\lambda g}(\boldsymbol{v}) = \arg\min_{\boldsymbol{x}\in\mathbb{R}^n} \left\{ \|\boldsymbol{v}-\boldsymbol{x}\|_2^2 + \lambda g(\boldsymbol{x}) \right\}. \tag{1.7}$$

Many choices of $g$ yield closed-form solutions for the proximal operator. Combining the gradient update with respect to $\ell$ with the proximal update with respect to $g$, we arrive at the update rule for proximal gradient descent, i.e.,

$$\boldsymbol{x}_{t+1} = \operatorname{prox}_{\lambda g}\left(\boldsymbol{x}_t - \eta\nabla\ell(\boldsymbol{x}_t)\right), \quad t = 0, \cdots, T. \tag{1.8}$$

## 1.4 Summary of Contributions

As we've seen through this introductory chapter, there are many pieces to the puzzle when it comes to designing an efficient algorithm for image reconstruction. First, we need to understand the domain knowledge of how our measurements are collected and what types of structures exist in the signals that we are measuring. Then, we need to pick an optimization function to solve and analyze the structure or geometry of this objective function. Additionally, we need to understand how to enforce the structural information that we know exists within a chosen algorithmic framework. Finally, we need to keep in mind that we are often interested in reconstructing a set of images rather than only a single instance. The work in this thesis brings insight to each of these notions of structure.

The thesis begins by examining how to mitigate the effects of system noise on reconstruction quality of a (vectorized) image from phaseless quadratic measurements. In reference to the signal measurement model (1.4), it is generally understood that noise in our system negatively impacts our ability to accurately reconstruct an image. Therefore, it makes sense to attempt to estimate the level of noise in the measurements and run a preprocessing step. Given accurate knowledge of the nature of the noise (e.g. the noise follows a certain probability distribution), one may develop a so-called "optimal" preprocessing function in an asymptotic, information theoretic sense. That being said, if we presume some noise structure and the true nature of the noise deviates from our assumption, preprocessing become detrimental. In Chapter 2, we analyze the structure of

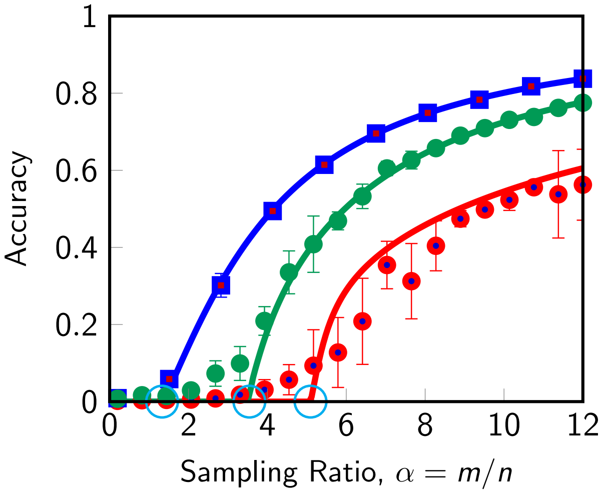


Figure 1.3: The accuracy of three different preprocessing functions as the sampling complexity is increased. The precise sampling ratios where a non-zero accuracy is obtained are highlighted with the cyan circles.

different types of noise in order to understand the benefits and drawbacks of utilizing potentially mismatched preprocessing functions. The efficiency of the preprocessing functions are determined by the sampling ratio required to obtain a certain level of accuracy, which is directly related to the sampling complexity. In Figure 1.3, we provide a simplified graph which demonstrates the desired level of accuracy when utilizing the correct preprocessing function for noisy measurements in comparison to suboptimal choices.

In Chapter 3, we consider the problem of recovering tall matrix from a set of phaseless quadratic measurements. As we discussed in motivating the optimization problem formulation in (1.5), it is important to consider the geometry of the chosen function, $\ell(\cdot)$. We demonstrate that the choice of $\ell(\cdot)$ with a more favorable curvature can impact both the convergence speed to an estimate and the number of measurements required in order to obtain an accurate estimate. We analyze the structure of a popular intensity-based loss function in comparison to a proposed amplitude-based loss function. The amplitude-based loss function is shown to have a more desirable geometry that allows for steeper descent paths. Algorithms using this choice of loss function are demonstrated to be more efficient in terms of both computational and statistical

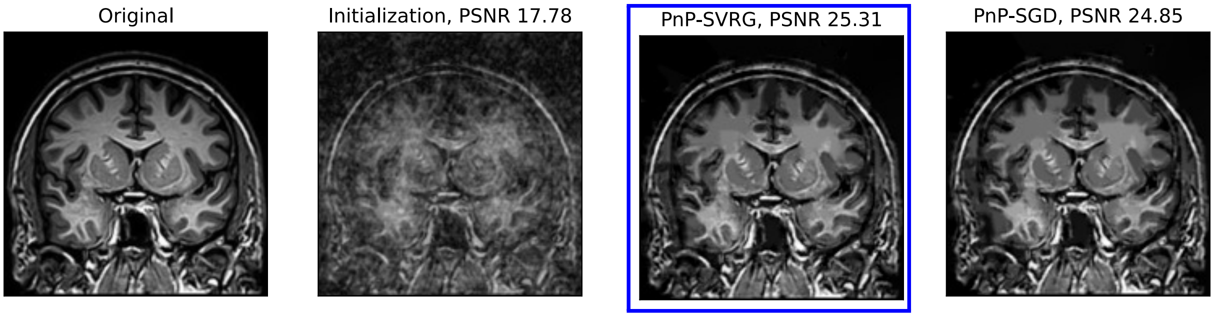


Figure 1.4: A demonstration of the improved efficiency that our proposed algorithm, PnP-SVRG, has over PnP-SGD, in terms of the PSNR of the reconstructed MRI within a fixed period of runtime.

complexity.

Moving forward, we investigate how to incorporate stronger notions of prior knowledge of images within the optimization framework. A popular image reconstruction algorithmic framework known as Plug-and-Play (PnP) has demonstrated the ability to leverage image denoisers to enforce "image-like" structure in a reconstructed signal. Previous work demonstrated how to incorporate the PnP framework into a stochastic gradient descent algorithm, (PnP-SGD). In Chapter 4, we demonstrate how to obtain the improved computational efficiency of variance-reduced gradient methods within the Plug-and-Play framework, while maintaining the same image reconstruction accuracy. The efficacy of PnP-SVRG is applied to an MRI application as well as the problem of phase retrieval. Some example MRI reconstruction results are displayed in Figure 1.4.

In many practical applications, we are not only interested in solving one instance of (1.4), but rather a set of such instances. While it is possible to process each instance in parallel, we look to improve by understanding structural knowledge present between any pair of instances. We demonstrate that when we are given a large set of related images to analyze, we are able to leverage similarities between ground truth images by introducing graph structures. This proposed methodology is demonstrated to be particularly useful in the context of indexing a set of electron back-scattered diffraction patterns obtained from a scanning electron microscope, as diffraction patterns captured at points within the same grain boundary will be identical. In the work discussed in Chapter 5, we demonstrate how to use graph structures to tie together the set of image reconstruction tasks.

Before moving into the body of this thesis, it is worth noting the role of image reconstruction within the world of computational imaging as a whole. The field of computational imaging can be understood with the help of the MRI example described above; some computation is necessary to go from the measurements that we collect (e.g. from a MRI scanner) to actual image visualization (e.g. the MRI image) for future tasks. A key part of this pipeline that is not addressed directly in this thesis is the tasks that are then done *after* the image is reconstructed. For example, we described that a doctor with a well-trained eye may be able to discern the presence and nature of a tumor within a patient's brain. However, another key component of computational imaging are tasks of computer vision, i.e., automatically detecting or predicting whether a tumor is present in an image without exterior knowledge from a doctor or human in general. While these problems are also important and interesting, they are beyond the scope of this thesis.

## 1.5 Thesis Organization and Notation

The thesis begins by demonstrating how to design effective spectral methods for estimating an image from phaseless measurements given approximate knowledge of the structure of the noise effecting the system in Chapter 2. Next, in Chapter 3, an amplitude-based loss function is proposed for solving a generalized matrix phaseless sensing problem and algorithms are derived which reach a critical point of such a loss function. Continuing, stochastic variance-reduction gradient techniques are applied to an algorithmic framework for reconstructing an image known as Plug-and-Play (PnP) is presented in Chapter 4 to achieve faster computation times while maintaining high accuracy. The thesis closes by analyzing how to reconstruct a large set of images when there exists a global structure relating the images to each other in Chapter 5. The methods presented in this thesis are applied to phase retrieval, compressive sensing magnetic resonance imaging, and electron back-scattered diffraction microscopy.

Throughout the thesis, we use boldfaced symbols to represent vectors and matrices. The symbols $\mathbb{R}$ and $\mathbb{C}$ denote the set of real and complex numbers, respectively. For any vector $\boldsymbol{v}$, we let $\|\boldsymbol{v}\|_2$ denote the $\ell_2$ norm, and let $\boldsymbol{v}^H$ indicate the conjugate transpose. For any matrix $\boldsymbol{M}$, we let $\|\boldsymbol{M}\|_F$ denote the Frobenius norm. In addition, we use $\boldsymbol{M}^H$ and $\boldsymbol{M}^\dagger$ to indicate the conjugate transpose and the pseudo-inverse of $\boldsymbol{M}$, respectively. The diagonal matrix with the diagonal entries given by the vector $\boldsymbol{v}$ is denoted as $\text{diag}(\boldsymbol{v})$. We use $\mathcal{Q}(x)$ to represent the tail distribution function of the standard normal distribution, i.e. $\mathcal{Q}(x) = \int_x^\infty \frac{1}{\sqrt{2\pi}} \exp(-z^2/2) dz$. Given two vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{C}^n$, $\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \sum_{i=1}^n x_i y_i^*$. We let $\mathbb{E}_x\{\cdot\}$ denote taking the expectation with respect to a random variable $x$. We use $\odot$ to denote the Hadamard product.

# Chapter 2

# Spectral Methods for Noisy Phase Retrieval

The spectral method is an important approach for signal estimation that is often used as an initialization to iterative methods as well as a stand-alone estimator, where the signal is estimated by the top eigenvector of certain carefully-constructed data matrix. A recent line of work has characterized the asymptotic behavior of such data matrices used in spectral methods, which reveals an interesting phase transition phenomenon: there exists a critical sampling threshold below which the estimate of the spectral method is uninformative. Furthermore, optimal preprocessing functions are developed to minimize this critical sampling threshold. In particular, most of the existing work is focused on the noiseless phase retrieval problem. In this chapter, our goal is to examine the sensitivity of such optimal preprocessing functions in noisy phase retrieval, when there is a mismatch between the noise model used in deriving the optimal preprocessing function and the actual noise model in practice. Our results provide important insights into the choice of preprocessing functions in spectral methods.

## 2.1 Introduction

Consider the problem of estimating an $n$-dimensional vector $\boldsymbol{x}^\star \in \mathbb{C}^n$ from a set of $m$ generalized linear measurements of the form

$$y_i \sim p(y \mid \langle \boldsymbol{a}_i, \boldsymbol{x}^\star \rangle), \quad i = 1, 2, ..., m, \tag{2.1}$$

where $p(\cdot \mid \cdot)$ is a known conditional probability density that describes how the measurements are obtained, $\{\boldsymbol{a}_i\}_{i=1}^m$ is an ensemble of sensing vectors, and $\langle \cdot, \cdot \rangle$ is the inner product. The spectral method [22, 28, 62] is a popular approach for estimating $\boldsymbol{x}^\star$, where it is estimated via the top eigenvector (up to scaling) of a carefully constructed data matrix that is a sum of rank-one matrices $\boldsymbol{a}_i\boldsymbol{a}_i^H$, each weighted by the corresponding measurement $y_i$, or a function $\mathcal{T}(y_i)$ of it. The spectral method can either be used as a stand-alone estimator or as the initialization of a more sophisticated method. For example, for the celebrated phase retrieval problem, the spectral method can be used to initialize a nonconvex iterative method such as gradient descent or alternating minimization [15, 20, 68, 78, 105, 109, 110] or provide an anchor vector to a convex linear program such as the Phasemax [4, 46].

Recently, a few works [66, 76] studied the asymptotic performance of the spectral method under the Gaussian design, where the sensing vectors are generated with i.i.d. standard complex Gaussian entries, in the regime where both $m$ and $n$ go to infinity with a fixed sampling ratio,

$$\alpha = \frac{m}{n}. \tag{2.2}$$

It not only provides a precise characterization of the performance of the spectral method, but also reveals an interesting phase transition phenomenon: there exists a critical sampling ratio threshold such that below which the estimate of the spectral method is uninformative, i.e., it is orthogonal to the ground truth signal. Moreover, [76] provides formula for an optimal *preprocessing* function $\mathcal{T}(y_i)$ to minimize this critical sampling threshold.

The goal of this chapter is to study the sensitivity of spectral initialization under model mismatch, when the actual signal model in practice is different from the one used to derive the optimal preprocessing function. This is of great relevance in practice, since typically the model is only imperfectly known, and may change during deployment. Therefore, it is necessary to see if the performance of the preprocessing functions are robust to model mismatch.

The work presented in Chapter 2 was published in IEEE ICASSP 2019 [73]. We demonstrate the impact of model mismatch on the performance of preprocessing functions that are derived for specific noise models. In order to reach this goal, we consider the problem of recovering an $n$-dimensional complex signal from $m$ quadratic measurements, known as *phase retrieval*, which is also studied in [66, 76] for the noiseless setting. In contrast, we consider the noisy setting, where the measurements are corrupted by either additive white Gaussian noise (AWGN) or Poisson noise. We first derive optimal preprocessing functions using the formula provided in [76], and characterize how the critical sampling threshold varies as a function of the noise level. Furthermore, through both empirical experiments and theoretical analyses, we examine the performance of the preprocessing functions when the noise level of the measurements are different from the one used in the derivation, which suggest some preprocessing functions are more sensitive to model perturbations.

### 2.1.1 Related Works

Though we use phase retrieval as an example, the spectral method has been applied to many statistical estimation problems such as low-rank matrix estimation [27, 56, 99], blind deconvolution [68], subspace estimation [26], to name a few. Many regularized variants of the spectral method have been proposed to improve its performance, which typically apply truncation or trimming to remove measurements that have high leverage [19, 20]. The asymptotic performance of the spectral method is analyzed first in [66] and then [76] for generalized linear models.

Before continuing, it is worth emphasizing the critical role of initialization in nonconvex statistical estimation. For several problems such as phase retrieval [91], low-rank matrix sensing [10] and completion [44], it is shown that with high probability, there are no spurious local minima in the landscape of the loss function except strict saddle points. Therefore, gradient descent with random initialization converge to the global optima almost surely for such problems [60], however the iteration complexity can be very high. In contrast, an optimally designed spectral initialization can provably land in a local basin of attraction near the ground truth [15], leading to faster convergence.

### 2.1.2 Chapter Organization

The rest of this chapter is organized as follows. Section 2.2 presents the signal model, and provides key metrics and backgrounds. Section 2.3 presents the optimal preprocessing functions derived for different noise models in phase retrieval. Section 2.5 provides numerical experiments to study the noise sensitivity of different preprocessing functions. Finally, we conclude in Section 2.6.

## 2.2 Backgrounds

In this section, we present the signal model for noisy phase retrieval, define key metrics and review the phase transition phenomenon as well as the design of optimal preprocessing functions in [66, 76] for the spectral method.

### 2.2.1 Signal Model and Important Definitions

We consider the problem of recovering an $n$-dimensional signal $\boldsymbol{x}^\star \in \mathbb{C}^n$ from $m$ intensity (quadratic) measurements in the presence of noise, where each measurement $y_i$ is collected according to

$$y_i \sim p(y \mid |\langle \boldsymbol{a}_i, \boldsymbol{x}^\star \rangle|^2), \qquad i = 1, 2, ..., m. \tag{2.3}$$

The set of known sensing vectors $\boldsymbol{a}_i$ is independently drawn from a complex Gaussian distribution, namely $\boldsymbol{a}_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\boldsymbol{0}, \frac{1}{2}\boldsymbol{I}_n) + j\mathcal{N}(\boldsymbol{0}, \frac{1}{2}\boldsymbol{I}_n)$, where $\boldsymbol{I}_n$ is the identity matrix of size $n$. Let $\alpha = m/n$ be defined as the sampling ratio. Furthermore, $p(\cdot)$ specifies some noise distribution such as additive white Gaussian noise (AWGN), or Poisson noise.

The spectral method first constructs a data matrix

$$\boldsymbol{D} = \frac{1}{m}\sum_{i=1}^{m} \mathcal{T}(y_i)\boldsymbol{a}_i\boldsymbol{a}_i^H, \tag{2.4}$$

where $\mathcal{T} : \mathbb{R} \longrightarrow \mathbb{R}$ is a deterministic function which we refer to as a *preprocessing function*. With spectral initialization, we take the top eigenvector $\hat{\boldsymbol{x}} \in \mathbb{C}^n$ of $\boldsymbol{D}$ as the initialization[1]. The intuition behind the spectral initialization is that, if we let $m \to \infty$, the top eigenvector of $\boldsymbol{D}$ perfectly recovers $\boldsymbol{x}^\star$ up to scaling. In practice, it is desirable to carefully design the preprocessing function so that $\hat{\boldsymbol{x}}$ is as close as possible to $\boldsymbol{x}^\star$ when it is calculated using a finite number of measurements. Thus, we wish to carefully design preprocessing functions that leverage our knowledge of the measurement model (2.3) in order to obtain a desirable initialization.

We study the regime where $m \to \infty$ and $n \to \infty$, but their ratio tends to a positive constant, i.e. $m/n \to \alpha$. The two metrics we will consider to determine the success of a preprocessing function are the *cosine-squared similarity* between the ground truth vector $\boldsymbol{x}^\star$ and the top eigenvector $\hat{\boldsymbol{x}}$ of the data matrix $\boldsymbol{D}$:

$$\rho(\hat{\boldsymbol{x}}, \boldsymbol{x}^\star) = \frac{|\langle \hat{\boldsymbol{x}}, \boldsymbol{x}^\star\rangle|^2}{\|\hat{\boldsymbol{x}}\|_2^2 \, \|\boldsymbol{x}^\star\|_2^2}, \tag{2.5}$$

and the *sampling threshold*,

$$\alpha_u = \operatorname*{argmin}_{\alpha^*} \left\{ \forall \alpha > \alpha^*, \ \liminf_{m\to\infty} \mathbb{E}_y \left\{ \rho(\hat{\boldsymbol{x}}, \boldsymbol{x}^\star)\right\} > 0 \right\}, \tag{2.6}$$

the minimum sampling threshold required to have a non-zero cosine-squared similarity.

By analyzing the asymptotic characterization of the spectral method, one can derive the behavior of the cosine-squared similarity as a function of the sampling ratio. The cosine-squared similarity $0 \leq \rho(\hat{\boldsymbol{x}}, \boldsymbol{x}^\star) \leq 1$ gives us a measure of how aligned our estimate is with the ground truth, by specifying the correlations between these two vectors. In [66], Lu and Li provided precise asymptotic predictions of the cosine-squared similarity as a function of the sampling threshold $\alpha$ and the distribution $p(\cdot)$. These predictions highlight the existence of a phase transition phenomenon, such that there exists a sampling threshold $\alpha_u$ that determines the effectiveness of

[1] The norm $\|\boldsymbol{x}^\star\|_2$ can be estimated easily, for example using the average of the measurements, which is not the focus here.

the spectral method. When the sampling ratio $\alpha$ is below the threshold $\alpha < \alpha_u$, we have $\rho = 0$ and the spectral method is uninformative; and when $\alpha > \alpha_u$, we have $\rho$ bounded away from 0, and the spectral estimate can be computed efficiently via the power method. The precise form of $\rho(\hat{\boldsymbol{x}}, \boldsymbol{x}^\star)$ with respect to $\alpha$ is derived in [66] for generalized linear models including phase retrieval.

### 2.2.2 Optimal Preprocessing Functions

The results in [66] suggest that the performance of spectral initialization can be drastically different using different preprocessing functions. In [76], an *optimal* preprocessing function was proposed to optimize the sampling threshold, so that it obtains the minimum sampling threshold and the best cosine-squared similarity for all sampling ratios. In [67], Luo et. al. constructed an optimal design of spectral methods that is *uniformly optimal* for all sampling ratios. We present Theorem 1 from [67] below.

**Theorem 1** (Theorem 1 from [67])**.** *Let $\boldsymbol{x}^\star \in \mathbb{C}^n$, $\{\boldsymbol{a}_i\}_{i=1}^m$ be a known sensing vector ensemble, and $y_i \sim p(y \mid |\langle \boldsymbol{a}_i, \boldsymbol{x}^\star \rangle|^2)$. Define $s = \langle \boldsymbol{a}_i, \boldsymbol{x}^\star \rangle$. Then the optimal preprocessing function for a pair of sensing vectors and noise distribution is given by*

$$\mathcal{T}(y) = 1 - \frac{\mathbb{E}_s\{p(y \mid |s|)\}}{\mathbb{E}_s\{|s|^2 p(y \mid |s|)\}}. \tag{2.7}$$

*Furthermore, the sampling threshold can be derived as*

$$\alpha_u = \left( \int_{\mathbb{R}} \frac{\mathbb{E}_s \left\{ p(y \mid |s|)(1 - |s|^2) \right\}}{\mathbb{E}_s \left\{ p(y \mid |s|) \right\}} \right)^{-1}. \tag{2.8}$$

This theorem is very useful for obtaining optimal preprocessing functions for a given measurement ensemble (2.3). In [76], an optimal preprocessing function was derived for the noiseless phase retrieval problem, given as $\mathcal{T}(y) = 1 - 1/y$, with a corresponding sampling threshold $\alpha_u = 1$.

## 2.3 Optimal Preprocessing Functions for Noisy Phase Retrieval

We proceed to present the optimal preprocessing functions for the cases where the measurements are contaminated by AWGN and Poisson noise in phase retrieval (2.3), as well as the corresponding theoretical sampling threshold, which were not considered in [76]. All proofs and derivations can be found in Appendix A. In Propositions 1 and 2, the preprocessing functions and sampling thresholds are obtained using (2.7) and (2.8), respectively, by plugging in the relevant probability distribution and sensing vector model.

**Proposition 1** (AWGN, Complex Gaussian Sensing Vectors). *Consider the set up described in Theorem 1, where* $\boldsymbol{a}_i \overset{i.i.d.}{\sim} \mathcal{N}(\mathbf{0}, \frac{1}{2}\boldsymbol{I}_n) + j\mathcal{N}(\mathbf{0}, \frac{1}{2}\boldsymbol{I}_n)$ *and* $p(y \mid |\langle \boldsymbol{a}_i, \boldsymbol{x}^\star\rangle|^2) \sim \mathcal{N}(|\langle \boldsymbol{a}_i, \boldsymbol{x}^\star\rangle|^2, \sigma^2)$. *Then the optimal preprocessing function is given as*

$$\mathcal{T}_\sigma(y) = 1 - \left( y - \sigma^2 + \sqrt{\frac{\sigma^2}{2\pi}} \frac{\exp\left(-\frac{(y-\sigma^2)^2}{2\sigma^2}\right)}{\mathcal{Q}\left(-\frac{(y-\sigma^2)}{\sigma}\right)} \right)^{-1}, \tag{2.9}$$

*and the corresponding sampling threshold is*

$$\alpha_u = \left( 1 - \sigma^2 - \sigma^4 + \frac{\sigma^3 e^{-\sigma^2/2}}{2\pi} \int_{-\infty}^{\infty} \frac{\exp\left(-\sigma u - u^2\right)}{\mathcal{Q}(-u)} du \right)^{-1}. \tag{2.10}$$

It is evident that both the optimal preprocessing function and the sampling threshold depends on the noise level $\sigma^2$ in a nonlinear manner. Figure 2.1 shows the optimal preprocessing function $\mathcal{T}_\sigma(y)$ with respect to $y$ for varied noise levels. For smaller measurements, i.e. $0 < y < 1$, the preprocessing function maps the measurements to a wide range of negative values; as the value of the measurement continues to increase $y_i > 1$, the preprocessing functions maps the measurements to the range $(0, 1)$. When there is no noise $\sigma^2 = 0$, this recovers the optimal preprocessing function introduced in [76], that is $\mathcal{T}_0(y) = 1 - 1/y$. At we increase the noise level, the dynamic range of the $\mathcal{T}_\sigma(y)$ also decreases.

**Proposition 2** (Poisson Noise, Complex Gaussian Sensing). *Consider the set up described in Theorem 1, where* $\boldsymbol{a}_i \overset{i.i.d.}{\sim} \mathcal{N}(\mathbf{0}, \frac{1}{2}\boldsymbol{I}_n) + j\mathcal{N}(\mathbf{0}, \frac{1}{2}\boldsymbol{I}_n)$ *and* $p(y \mid |\langle \boldsymbol{a}_i, \boldsymbol{x}^\star\rangle|^2)$ *follows a Poisson*

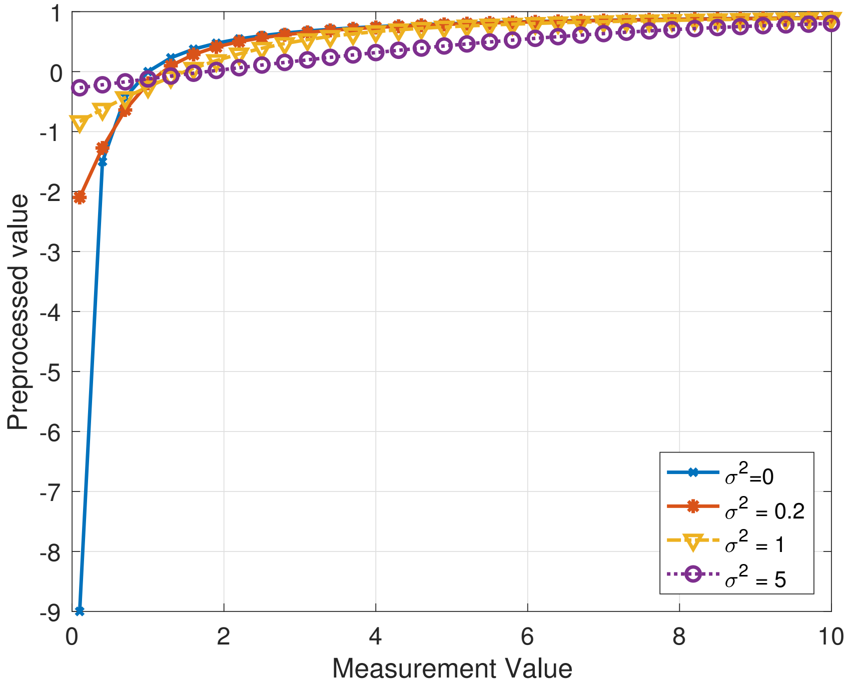


Figure 2.1: The optimal preprocessing function $\mathcal{T}_\sigma(y)$ with respect to the measurement value $y$ for noisy phase retrieval with AWGN under different noise levels $\sigma^2 = 0, 0.2, 1$, and $5$.

*distribution with rate $|\langle \boldsymbol{a}_i, \boldsymbol{x}^\star \rangle|^2$. Then the optimal preprocessing function is given as*

$$\mathcal{T}(y) = \frac{y-1}{y+1}. \tag{2.11}$$

*The sampling threshold is $\alpha_u = 2$.*

Similar to the AWGN case, the optimal preprocessing function for Poisson noise maps smaller measurements to a wider range of values (in a relative sense), while mapping larger measurements to values approaching $1$ as $y \to \infty$, similar to a simple truncation scheme. For example, $y \in \{0, 1, ..., 5\}$ maps to a range of $-1 \le \mathcal{T}(y) \le \frac{2}{3}$, while for $y_i \ge 6$, $\frac{5}{7} \le \mathcal{T}(y) < 1$. Additionally, notice that the sampling threshold for Poisson noise is $2$, in contrast to the sampling threshold in the AWGN case, which can approach $1$ when the noise goes to zero. In comparison, to achieve the same sampling threshold, processing the Poisson noise is about the same as AWGN at $\sigma^2 \approx 1.34$.

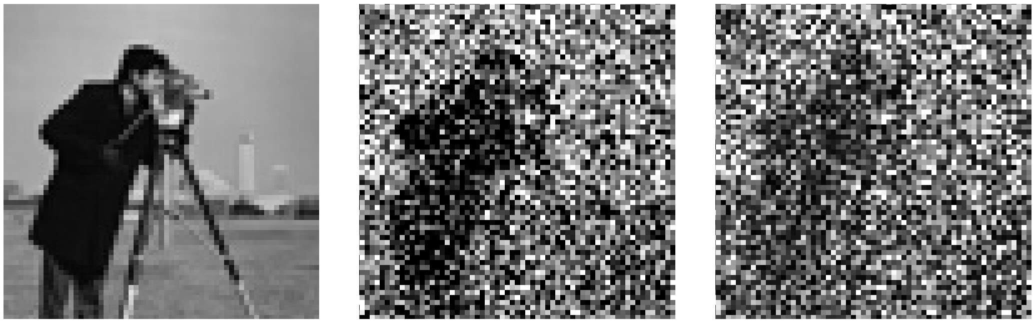


Figure 2.2: We visualize the effect that noisy measurements have on the quality of the spectral estimate when preprocessing functions are not used. On the far left is the original image. In the middle is the spectral estimate when there is no noise added to the phaseless measurements. In the last column is the spectral estimate when additive white Gaussian noise with variance $\sigma^2 = 5$ is added to each phaseless measurement.

## 2.4 Qualitative Effect of Preprocessing Functions

In this section, we visualize the effects of utilizing preprocessing functions in generating spectral estimates of a $64 \times 64$ pixel image from phaseless measurements. For each experiment, let the total dimensions $n = 4096$, and the number of measurements $m = 10n$. Measurements are then drawn according to (2.3). When preprocessing functions are used, they are derived with respect to the definition presented in (2.7).

In the first experiment, we analyze the effect that additive white Gaussian noise has on the fidelity of the spectral estimate in the case when no preprocessing functions are used. Referring to Figure 2.2, we visualize the accuracy of the spectral estimate in the case where no additive noise is present in the phaseless measurements, and the case when additive white Gaussian noise with variance $\sigma^2 = 5$ has been added to each of the $m$ measurements. Since this experiment is in a regime where the number of measurements is relatively small, one should not expect the spectral estimate to be particularly accurate–it is meant to be used as an initial point or as an anchor, after all. However, one can make out basic features within the image such as the outline of the cameraman. These details within the image are less prevalent when noise is added to

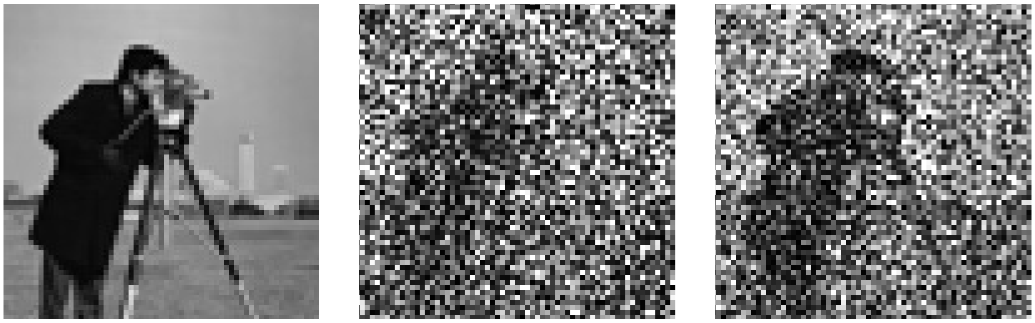


Figure 2.3: We visualize the effect that preprocessing functions have on improving the reconstruction of the image in case of noisy, phaseless measurements. On the far left is the original image. In the middle is the spectral estimate when a preprocessing fucntion is not used. In the last column is the spectral estimate when the optimal preprocessing function, $\mathcal{T}_5(\cdot)$, is used given the noise model.

the measurements. This demonstrates the necessity of using preprocessing functions when the phaseless measurements are corrupted by noise.

As shown in Figure 2.3, we demonstrate the effectiveness that utilizing preprocessing functions has in the case when the phaseless measurements are noisy. The middle image of Figure 2.3 is the same as the final image in Figure 2.2, i.e. the spectral estimate without the use of preprocessing functions. In comparison with the final image of Figure 2.3, one can see that when the optimal preprocessing function with respect to the noise model, i.e., $\mathcal{T}_5(\cdot)$, is used, we are once again able to recover some key features of the cameraman within the image. Indeed, this visual improvement is reflected in the spectral estimate having higher correlation with the ground truth image.

Finally, we visualize the scenario where there is a mismatch between the true noise model and the preprocessing function used. In Figure 2.4 we compare the quality of the spectral estimate when the matching optimal preprocessing function, $\mathcal{T}_5(\cdot)$, is used, to the quality when using $\mathcal{T}_1(\cdot)$. In this case, the noise level has been underestimated, leading to a model-mismatch. Evidentally, utilizing the incorrect preprocessing function can lead to more harm than good. When comparing the final images in Figure 2.2 and Figure 2.4, the spectral estimate when using the incorrect preprocesing function as poor as the spectral estimate when no preprocessing function is used.

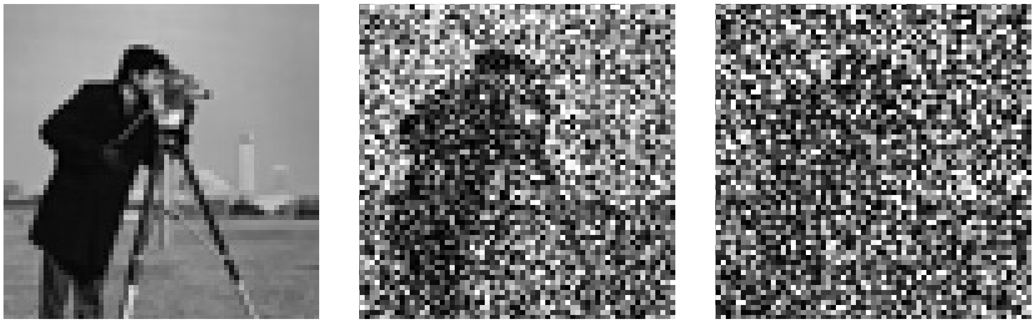


Figure 2.4: We visualize the effect of using the wrong preprocessing function. On the far left is the original image. In the middle is the spectral estimate when the optimal preprocessing function, $\mathcal{T}_5(\cdot)$, is used, while the last column is the spectral estimate when incorrect preprocessing function, $\mathcal{T}_1(\cdot)$, is used.

## 2.5 Sensitivity Studies via Numerical Experiments

In this section, we provide numerical experiments that demonstrate the sensitivity to model mismatch of the preprocessing function performance, focusing on the cases of AWGN and Poisson noise for noisy phase retrieval. Imagine one applies the optimal preprocessing function designed for a postulated measurement model, while the actual measurement noise is different from the postulated one. This situation is highly relevant in practice, since we either do not have perfect knowledge of the model. We will examine the sensitivity of both the sampling threshold and the cosine-squared similarity under model mismatch.

For all experiments, the sensing vectors $\boldsymbol{a}_i$ are identically and independently distributed as complex Gaussian vectors, i.e. $\boldsymbol{a}_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\mathbf{0}, \frac{1}{2}\boldsymbol{I}_n) + j\mathcal{N}(\mathbf{0}, \frac{1}{2}\boldsymbol{I}_n)$, for $i = 1, ..., m$. For the cosine-squared similarity curves, we refer to "trimming" as the preprocessing function where $\mathcal{T}(y) = \min(y, K)$ for some constant $K$. For testing the empirical performance of each preprocessing function, we run $8$ Monte Carlo experiments, where the signal dimensions are fixed to $n = 1024$. We note that as the signal dimension increases, the empirical curve becomes closer to the theoretical curve. The theoretical curves for the cosine-squared similarity are obtained following the derivations from [66].

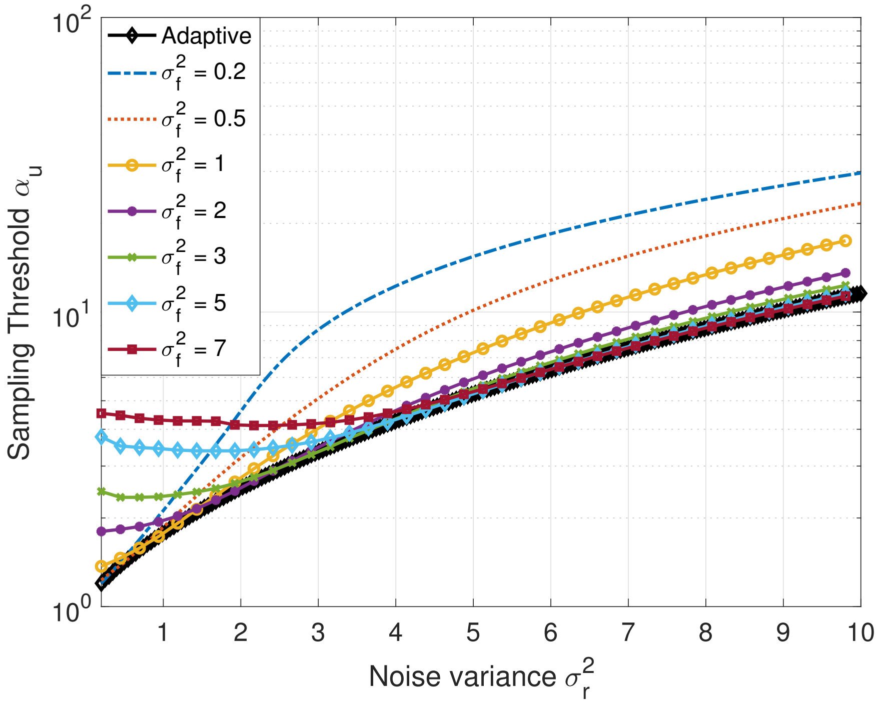


Figure 2.5: The sampling threshold $\alpha_u$ with respect to the true AWGN noise level $\sigma_r^2$ for a preprocessing function designed to be optimal for a postulated noise level $\sigma_f^2 = 0.2, 0.5, 1, 2, 3, 5, 7$, respectively. The sampling threshold of the adaptive preprocessing function corresponds to (2.10) when $\sigma_f^2 = \sigma_r^2$.

### 2.5.1 Sensitivity of Sampling Thresholds

We start by investigating the sampling threshold of the optimal preprocessing functions in AWGN for noisy phase retrieval. We assume that a fixed preprocessing function is used to process the measurements, designed for a postulated noise level $\sigma_f^2$, while the true noise level is set at $\sigma_r^2$. Figure 2.5 plots the sampling threshold of the preprocessing function with respect to the true noise level, when it is designed with respect to different postulated noise level $\sigma_f^2 = 0.2, 0.5, 1, 2, 3, 5,$ and 7. In addition, Figure 2.5 also plots the sampling threshold in (2.10) corresponds to the case when the postulated noise level in the preprocessing function matches with the true noise level, dubbed as the *adaptive* AWGN preprocessing function. Clearly, the adaptive AWGN preprocessing function where $\sigma_f^2 = \sigma_r^2$ serves as a lower bound of the minimal sampling threshold.

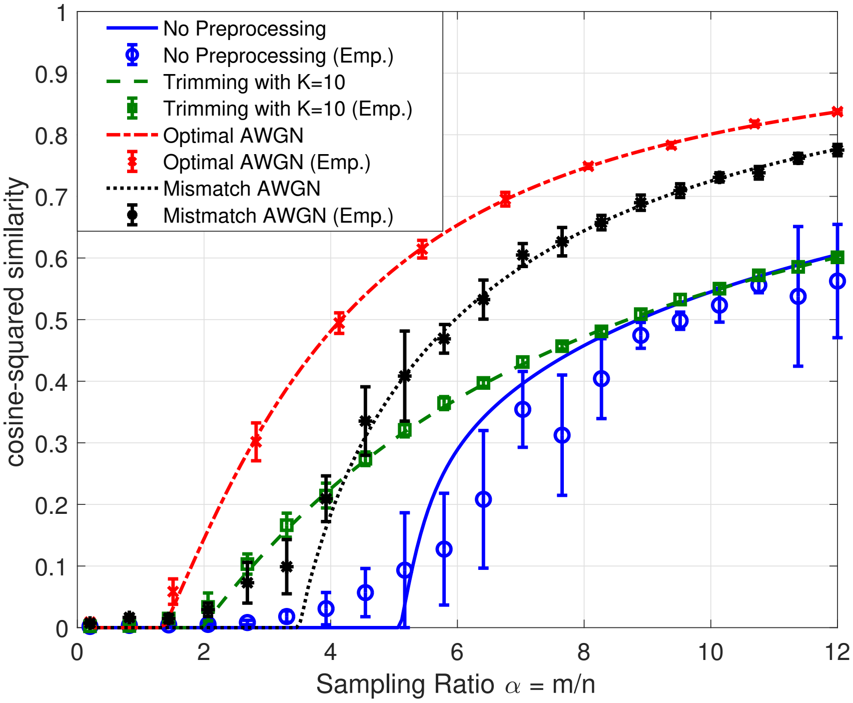


Figure 2.6: The theoretical prediction and the empirical realizations of the cosine-squared similarity of each preprocessing function with respect to the sampling ratio under AWGN with $\sigma_r^2 = 0.5$. The Optimal AWGN curve postulates $\sigma_f^2 = \sigma_r^2 = 0.5$, whereas the Mismatch AWGN curve assume $\sigma_f^2 = 5$.

It is interesting to observe that when the mismatch level is small, the deviation in the sampling threshold is also relatively small. However, the performance can be drastically worse when the mismatch level is high.

### 2.5.2 Sensitivity of Cosine-Squared Similarities

Next, we examine the sensitivity of cosine-squared similarities over a wide range of the sampling ratio in both AWGN and Poisson noise. In Figure 2.6, we examine the empirical and theoretical performance of the optimal AWGN preprocessing function in the presence of AWGN with true noise level $\sigma_r^2 = 0.5$ as the sampling ratio increases. The AWGN preprocessing function where $\sigma_f^2 = \sigma_r^2$ outperforms the other in terms of achieving the theoretical sampling ratio threshold

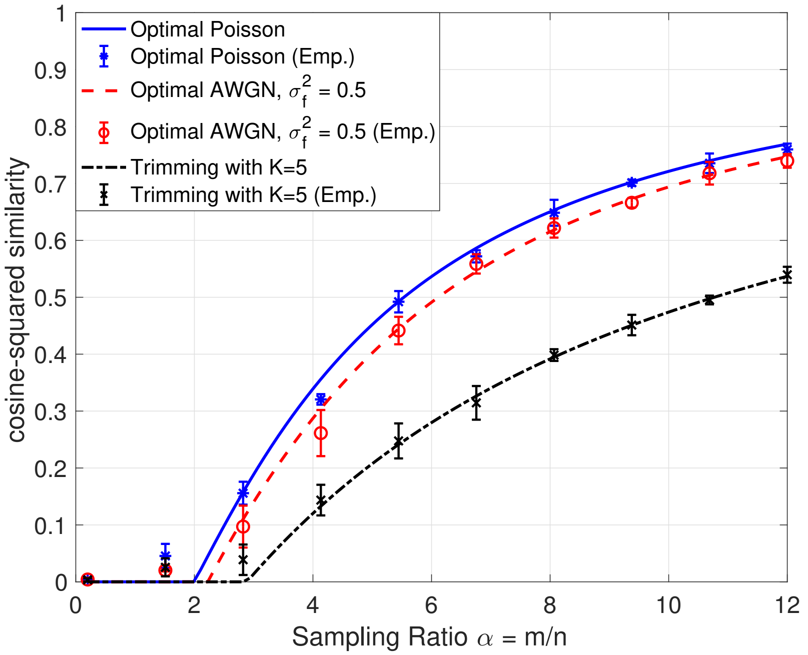


Figure 2.7: The theoretical prediction and the empirical realizations of the cosine-squared similarity of each preprocessing function with respect to the sampling ratio under Poisson noise. The Optimal AWGN curve postulates $\sigma_f^2 = 0.5$ and applies (2.9).

as well as the overall cosine-squared similarity performance. Compared with the mismatched AWGN preprocessing function where $\sigma_f^2 = 5$, the trimming preprocessing function obtains a smaller sampling ratio threshold, but the overall cosine-squared similarity does not increase at the same rate as the mismatched AWGN preprocessing function. Therefore, one may not determine the performance of the spectral method based only on the sampling threshold. The performance of the vanilla spectral initialization without preprocessing is much worse with more variability.

Lastly, we test the performance of preprocessing functions derived for different noise models in the presence of Poisson noise in Figure 2.7. The optimal preprocessing function for Poisson noise obtains the optimal sampling ratio threshold and outperforms the other preprocessing functions also in terms of the cosine-squared similarity. The AWGN preprocessing function with $\sigma_f^2 = 0.5$ performs nearly as well as the sampling ratio increases, but does not obtain the optimal sampling ratio threshold.

## 2.6 Conclusion

We studied the sensitivity of optimal preprocessing functions in spectral methods when there is a mismatch between the theoretical model and the practical model. Using noisy phase retrieval as a case study, we derived the optimal preprocessing functions under both Gaussian noise and Poisson noise, and further compared their performances under model mismatch. In the future, it would be interesting to analyze the benefits of preprocessing functions for more noise models and for other sensing vector formulations beyond the random Gaussian design. Additionally, investigating the generalization of preprocessing functions to vector signals to matrix signals would be beneficial.

# Chapter 3

# Solving Quadratic Equations via Amplitude-Based Nonconvex Optimization

In many signal processing tasks, one seeks to recover an $r$-column matrix object $\boldsymbol{X} \in \mathbb{C}^{n\times r}$ from a set of nonnegative quadratic measurements up to orthonormal transforms. Example applications include coherence retrieval in optical imaging and covariance sketching for high-dimensional streaming data. To this end, efficient nonconvex optimization methods are quite appealing, due to their computational efficiency and scalability to large-scale problems. There is a recent surge of activities in designing nonconvex methods for the special case $r = 1$, known as phase retrieval; however, very little work has studied the general rank-$r$ setting. Motivated by the success of phase retrieval, in this chapte we derive several algorithms which utilize the quadratic loss function based on amplitude measurements, including (stochastic) gradient descent and alternating minimization. Numerical experiments demonstrate their computational and statistical performances, highlighting the superior performance of stochastic gradient descent with appropriate mini-batch sizes.

## 3.1 Introduction

In this chapter, we are interested in recovering an $r$-column matrix object $\boldsymbol{X} \in \mathbb{C}^{n\times r}$ from a set of non-negative quadratic measurements, given as

$$y_i = \|\boldsymbol{a}_i^H \boldsymbol{X}\|_2^2, \quad i = 1, \dots, m, \tag{3.1}$$

where $\boldsymbol{a}_i \in \mathbb{C}^n$ is the $i$th sensing vector, and $m$ is the number of measurements. In the Gaussian design, the sensing vectors are generated i.i.d. from a complex-valued Gaussian distribution, i.e. $\boldsymbol{a}_i \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\boldsymbol{0}, \frac{1}{2}\boldsymbol{I}_n) + j\mathcal{N}(\boldsymbol{0}, \frac{1}{2}\boldsymbol{I}_n)$. Equivalently, this problem amounts to recovering a rank-$r$ positive semi-definite matrix $\boldsymbol{M} = \boldsymbol{X}\boldsymbol{X}^H \in \mathbb{C}^{n\times n}$ from a set of linear measurements

$$y_i = \boldsymbol{a}_i^H \boldsymbol{M} \boldsymbol{a}_i = \langle \boldsymbol{M}, \boldsymbol{a}_i \boldsymbol{a}_i^H \rangle, \tag{3.2}$$

where the sensing matrix $\boldsymbol{a}_i\boldsymbol{a}_i^H$ is rank-one. Since the measurements $\boldsymbol{y} = \{y_i\}_{i=1}^m$ are non-negative, they are dubbed *phaseless* measurements. The goal is to recover $\boldsymbol{M}$, or equivalently the factor $\boldsymbol{X}$ up to orthonormal transforms from as few measurements $m \ll n^2$ as possible. This problem arises in many applications, ranging from coherence retrieval in optical imaging [5], covariance sketching of high-dimensional streaming data [13, 23] for the general rank-$r$ case, to phase retrieval [16, 39, 87, 104] for the rank-1 case. The measurements $\boldsymbol{y}$ are quadratic in both $\boldsymbol{a}_i$'s and $\boldsymbol{X}$, and we therefore also refer to this sensing model as *quadratic sensing*.

There are two lines of approaches to solve this problem. The first one is based on convex relaxations to solve for the low-rank matrix $\boldsymbol{M}$ [16, 65]. These algorithms perfectly recover the underlying matrix $\boldsymbol{M}$ at a near-optimal sample complexity under the Gaussian design. However, the computational complexity of the resulting semi-definite programs scale at least cubically with the dimension of $\boldsymbol{M}$, and therefore is prohibitive when the problem size is large.

This leads to the second line of approaches, which are iterative algorithms based on non-convex optimization that directly estimate the factor $\boldsymbol{X}$ [15, 20, 28, 68]. In [64, 86], it is proposed to recover $\boldsymbol{X}$ directly by minimizing the following loss function that is the squared error of the

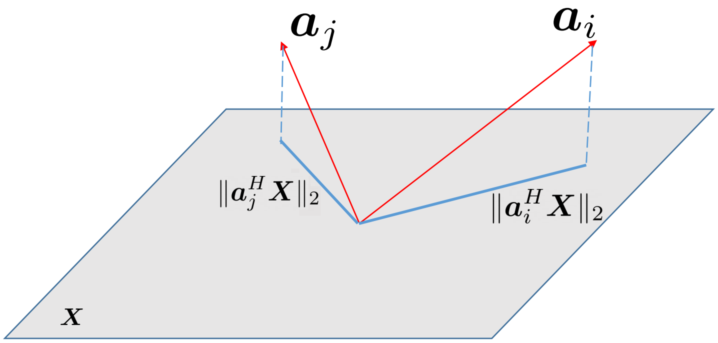

$\boldsymbol{a}_j$
$\boldsymbol{a}_i$
$\|\boldsymbol{a}_j^H\boldsymbol{X}\|_2$
$\|\boldsymbol{a}_i^H\boldsymbol{X}\|_2$
$\boldsymbol{X}$

Figure 3.1: A geometric interpretation of estimating a matrix, $\boldsymbol{X}$, from phaseless measurements. The measurements can be understood as back-projections formed by querying $\boldsymbol{X}$ with sensing vectors $\{\boldsymbol{a}_i\}_{i=1}^m$. Evidently, the problem at hand is related to the subspace retrieval problem.

*intensity* measurements $y_i$:

$$\ell_{\mathsf{in}}(\boldsymbol{U}) := \frac{1}{m}\sum_{i=1}^{m}\left(y_i - \|\boldsymbol{a}_i^H\boldsymbol{U}\|_2^2\right)^2, \tag{3.3}$$

where $\boldsymbol{U} \in \mathbb{C}^{n\times r}$. The algorithm proposed in [64, 86] is gradient descent with spectral initialization, and its efficiency is proved in [64] for the Gaussian design.

The work presented in Chapter 3 was published in part at IEEE ICASSP 2019 [75]. This work proposes to solve for $\boldsymbol{X}$ by minimizing the squared error of the *amplitude* measurements, that is, $z_i = \sqrt{y_i}$ for $i = 1, ..., m$. The goal is to minimize the following loss function:

$$\ell(\boldsymbol{U}) = \frac{1}{m}\sum_{i=1}^{m}\left(z_i - \|\boldsymbol{a}_i^H\boldsymbol{U}\|_2\right)^2, \tag{3.4}$$

which is both non-convex and non-smooth. To the best of our knowledge, this loss function has not been considered to solve the general rank-$r$ quadratic sensing problem considered here. Compared with the intensity-based loss function (3.3), the amplitude-based loss function (3.4) is a lower-order polynomial with respect to $\boldsymbol{a}_i^H\boldsymbol{U}$, and therefore is expected to have better curvatures around the global optimum, and more amenable to fast computation. We developed three algorithms to optimize (3.4): gradient descent (GD), mini-batch stochastic gradient descent (SGD), and alternating minimization, which are initialized by the spectral method based on amplitude measurements. All of these algorithms converge to a critical point of $\ell(\boldsymbol{U})$, and empirically achieve stronger statistical and computational performances than optimizing the intensity-based loss function (3.3) via gradient descent [64].

### 3.1.1 Related Work

This work is motivated by the successful adoption of the amplitude-based loss function in phase retrieval [69, 105, 110], where it achieves near-optimal computational and statistical complexities without requiring sophisticated truncation or regularization procedures as using the intensity-based loss function [20]. Besides [64, 86], an exponential-type gradient descent algorithm is proposed in [52] to minimize (3.3), which is similar to the truncation rule in [20] to suppress samples that heavily influence the search direction. A few papers proposed other algorithms to solve the quadratic sensing problem, including but not limited to [88, 113], but they are applied to either the lifted formulation (3.2) or the intensity-based based loss function (3.3).

### 3.1.2 Chapter Organization

The rest of this chapter is organized as follows. Section 3.2 presents the proposed algorithms using the amplitude-based loss function, including (stochastic) gradient descent and alternating minimization. Section 3.3 examines and provides numerical comparisons of the proposed algorithms with existing approaches. Section 3.4 provides additional insight on the benefits of using Scaled gradient descent (ScaledGD) methods for estimating ill-conditioned matrices. Finally, we conclude in Section 3.5.

## 3.2 Amplitude-Based Non-convex Optimization

We start by providing the intuition behind using the amplitude-based loss function (3.4) and then introduce three different algorithms to minimize it: gradient descent, stochastic gradient descent which also utilizes mini-batches, and alternating minimization. Finally, we propose a spectral initialization based on the amplitude measurements.

### 3.2.1 Making Sense of the Amplitude-Based Loss Function

We start by defining a generalized "phase" vector

$$\boldsymbol{b}_i^H = \frac{\boldsymbol{a}_i^H \boldsymbol{X}}{\|\boldsymbol{a}_i^H \boldsymbol{X}\|_2} \in \mathbb{C}^r \tag{3.5}$$

corresponding to each phaseless measurement. Then with the amplitude measurements $z_i = \|\boldsymbol{a}_i^H \boldsymbol{X}\|_2$, we can write a set of linear measurements of $\boldsymbol{X}$ as

$$\boldsymbol{a}_i^H \boldsymbol{X} = \boldsymbol{b}_i^H z_i, \qquad i = 1, ..., m.$$

Furthermore, define

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{a}_1^H \\ \boldsymbol{a}_2^H \\ \vdots \\ \boldsymbol{a}_m^H \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} \boldsymbol{b}_1^H \\ \boldsymbol{b}_2^H \\ \vdots \\ \boldsymbol{b}_m^H \end{bmatrix}, \quad \boldsymbol{z} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_m \end{bmatrix}. \tag{3.6}$$

Then we can compactly write

$$\boldsymbol{A}\boldsymbol{X} = \text{diag}(\boldsymbol{z})\boldsymbol{B}. \tag{3.7}$$

Indeed, if the matrix $\boldsymbol{B}$ is known, then $\boldsymbol{X}$ can be solved via standard least-squares. The challenge is that the phase term $\boldsymbol{B}$ is unknown, and we cannot apply least-squares directly. To this end, we aim to find the phase matrix $\boldsymbol{B}$ and $\boldsymbol{X}$ that minimize the loss function:

$$\min_{\boldsymbol{U}, \|\boldsymbol{p}_i\|_2 = 1} \ell(\boldsymbol{U}, \boldsymbol{P}) = \frac{1}{m} \|\boldsymbol{A}\boldsymbol{U} - \text{diag}(\boldsymbol{z})\boldsymbol{P}\|_F^2, \tag{3.8}$$

under the constraint that the rows of $\boldsymbol{P}$ are unit-norm. Interestingly, when fixing $\boldsymbol{U}$ (the estimate of $\boldsymbol{X}$), the phase vector that minimizes the right-hand side of (3.8) can be found in a closed-form as

$$\boldsymbol{p}_i^H = \boldsymbol{a}_i^H \boldsymbol{U} / \|\boldsymbol{a}_i^H \boldsymbol{U}\|_2. \tag{3.9}$$

Plugging (3.9) into $\ell(\boldsymbol{U}, \boldsymbol{P})$ lead to the amplitude-based loss function in (3.4):

$$\ell(\boldsymbol{U}) = \min_{\|\boldsymbol{p}_i\|_2 = 1} \ell(\boldsymbol{U}, \boldsymbol{P}).$$

In words, the amplitude-based loss function can be regarded as an attempt to approximate the least-squares loss in the absence of the phase information.

### 3.2.2 Gradient Descent

A first approach is to apply gradient descent to minimize (3.4), which may proceed at each iteration $k \geq 0$ as

$$\boldsymbol{U}_{k+1} = \boldsymbol{U}_k - \mu_k \nabla \ell(\boldsymbol{U}_k) \tag{3.10}$$

for some step size $\mu_k$, and $\boldsymbol{U}_0$ is a properly chosen initialization that will be discussed later. Due to the non-smoothness of the loss function, the generalized gradient [57] of $\ell(\boldsymbol{U})$ with respect to $\boldsymbol{U}$ is used:

$$\nabla \ell(\boldsymbol{U}) = \frac{2}{m} \sum_{i=1}^{m} \left( \|\boldsymbol{a}_i^H \boldsymbol{U}\|_2 - z_i \right) \frac{\boldsymbol{a}_i \boldsymbol{a}_i^H \boldsymbol{U}}{\|\boldsymbol{a}_i^H \boldsymbol{U}\|_2}$$

The details of gradient descent with the amplitude loss function is given in Algorithm 3.1. Comparing with the least-squares case when the generalized phase matrix $\boldsymbol{B}$ is known, here at each iteration, the phase vector is estimated via the current iterate.

### 3.2.3 Mini-Batch Stochastic Gradient Descent

Next, we implement a stochastic version of gradient descent using mini-batches, which is found in practice to be compelling both statistically and computationally. By utilizing a stochastic method

**Algorithm 3.1** Gradient Descent with Amplitude Loss

```
Input: $\{z_i\}_{i=1}^m$, $\{a_i\}_{i=1}^m$, step size $\mu_k$.
  Initialization:
  Obtain $U_0$ from Spectral Initialization (Algorithm 3.4).
  Gradient Updates:
  for $k = 0, 1, 2, ..., K-1$ do
    $W_k = \mathrm{diag}\left(\left[\frac{\|a_1^H U_k\|_2 - z_1}{\|a_1^H U_k\|_2}, ..., \frac{\|a_m^H U_k\|_2 - z_m}{\|a_m^H U_k\|_2}\right]\right).$
    $U_{k+1} = U_k - \frac{2\mu_k}{m} A^H W_k A U_k.$
  end for
  return $U_K$.
```

with an appropriate mini-batch size, we significantly decrease the computation cost per iteration while still converging at a moderate number of iterations. The details of the mini-batch stochastic gradient descent (SGD) is given in Algorithm 3.2.

### 3.2.4 Alternating Minimization

Last but not least, we propose alternating minimization (AltMin) to update the phase matrix and the signal sequentially to minimize $\ell(\boldsymbol{U}, \boldsymbol{P})$ in (3.8), which leads to a direct generalization of the well-known Gerchberg-Saxton algorithm for phase retrieval [45] to the general quadratic sensing problem. In words, at each iteration $k \geq 0$, we update each row of the phase matrix $\boldsymbol{P}_k = [\boldsymbol{p}_1^k, \dots, \boldsymbol{p}_m^k]^H$ as

$$(\boldsymbol{p}_i^k)^H = \boldsymbol{a}_i^H \boldsymbol{U}_k / \|\boldsymbol{a}_i^H \boldsymbol{U}_k\|_2, \quad i = 1, \dots, m, \tag{3.11}$$

Then, we update $\boldsymbol{U}_{k+1}$ by fixing $\boldsymbol{P}_k$ and solving a least-squares problem:

$$\begin{aligned} \boldsymbol{U}_{k+1} &= \mathrm{argmin}_{\boldsymbol{U} \in \mathbb{C}^{n \times r}} \|\boldsymbol{A}\boldsymbol{U} - \mathrm{diag}(\boldsymbol{z})\boldsymbol{P}_k\|_2^2 \\ &= \boldsymbol{A}^\dagger \,\mathrm{diag}(\boldsymbol{z})\boldsymbol{P}_k \\ &= \boldsymbol{A}^\dagger \,\mathrm{diag}\left(\left[\frac{z_1}{\|\boldsymbol{a}_1^H \boldsymbol{U}_k\|_2}, ..., \frac{z_m}{\|\boldsymbol{a}_m^H \boldsymbol{U}_k\|_2}\right]\right) \boldsymbol{A}\boldsymbol{U}_k. \end{aligned} \tag{3.12}$$

**Algorithm 3.2** Mini-Batch SGD with Amplitude Loss

```
Input: $\{z_i\}_{i=1}^m$, $\{\boldsymbol{a}_i\}_{i=1}^m$, mini-batch size $B$, step size $\mu_k$.
  Initialization:
  Obtain $\boldsymbol{U}_0$ from Spectral Initialization (Algorithm 3.4).
  Gradient Updates:
  for $k = 0, 1, 2, ..., K-1$ do
    Choose $\Gamma_k$ uniformly at random from $\{1, 2, ..., m\}$ with cardinality $B$ and update.
    $\boldsymbol{W}_{\Gamma_k} = \mathrm{diag}\left(\left\{ \frac{\|\boldsymbol{a}_i^H \boldsymbol{U}_k\|_2 - z_i}{\|\boldsymbol{a}_i^H \boldsymbol{U}_k\|_2} \,\middle|\, i \in \Gamma_k \right\}\right)$,
    $\boldsymbol{U}_{k+1} = \boldsymbol{U}_k - \frac{2\mu_k}{m} \boldsymbol{A}_{\Gamma_k}^H \boldsymbol{W}_{\Gamma_k} \boldsymbol{A}_{\Gamma_k} \boldsymbol{U}_k$.
    where $\boldsymbol{A}_{\Gamma_k}$ is a matrix stacking of $\boldsymbol{a}_i^H$ for $i \in \Gamma_k$ as its rows.
  end for
  return $\boldsymbol{U}_K$.
```

The details of the alternating minimization algorithm are shown in Algorithm 3.3. It is easy to check that this update rule is guaranteed to not increase the amplitude loss function at every iteration:

$$\begin{aligned}
\ell(\boldsymbol{U}_{k+1}) &= \min_{\|\boldsymbol{p}_i\|=1} \frac{1}{m} \sum_{i=1}^{m} \|\boldsymbol{a}_i^H \boldsymbol{U}_{k+1} - y_i \boldsymbol{p}_i\|_2^2 \\
&\le \frac{1}{m} \|\boldsymbol{A}\boldsymbol{U}_{k+1} - \mathrm{diag}(\boldsymbol{y})\boldsymbol{P}_k\|_2^2 \\
&\le \frac{1}{m} \|\boldsymbol{A}\boldsymbol{U}_{k} - \mathrm{diag}(\boldsymbol{y})\boldsymbol{P}_k\|_2^2 \\
&= \ell(\boldsymbol{U}_k),
\end{aligned}$$

where the second inequality follows from (3.12).

### 3.2.5 Spectral Initialization

So far, all the algorithms require an initialization $\boldsymbol{U}_0$, which hopefully is close to the ground truth $\boldsymbol{X}$ that we wish to recover. The spectral method is a popular method to provide a high-quality initial guess in non-convex optimization, where we construct a data matrix based on the

**Algorithm 3.3** Alternating Minimization

```
Input: ${z_i}_{i=1}^m$, and ${\boldsymbol{a}_i}_{i=1}^m$.
  Initialization:
  Obtain $\boldsymbol{U}_0$ from Spectral Initialization (Algorithm 3.4).
  Alternating Updates:
  for $k = 0, 1, 2, ..., K-1$ do
    $\boldsymbol{U}_{k+1} = \boldsymbol{A}^{\dagger}\text{diag}\left(\left[\frac{z_1}{\|\boldsymbol{a}_1^H\boldsymbol{U}_k\|_2}, ..., \frac{z_m}{\|\boldsymbol{a}_m^H\boldsymbol{U}_k\|_2}\right]\right)\boldsymbol{A}\boldsymbol{U}_k$.
  end for
  return $\boldsymbol{U}_K$.
```

measurements and sensing vectors and use its principal subspace to provide an initial guess. In this chapter, we advocate the use of amplitude measurements to construct the data matrix, as detailed in Algorithm 3.4. The method is inspired by [64], except for the use of amplitude measurements. Numerical experiments in the later section will verify the advantage of this approach.

**Algorithm 3.4** Spectral Initialization with Amplitude Measurements

```
Input: ${z_i}_{i=1}^m$, and ${\boldsymbol{a}_i}_{i=1}^m$
  Define the data matrix $\boldsymbol{D} = \frac{1}{2m}\sum_{i=1}^m z_i\boldsymbol{a}_i\boldsymbol{a}_i^H$.
  Obtain the $r$ normalized eigenvectors $\boldsymbol{Z}_0 \in \mathbb{C}^{n\times r}$ corresponding to the $r$ largest eigenvalues of $\boldsymbol{D}$.
  Obtain the diagonal matrix $\boldsymbol{\Lambda}_0 \in \mathbb{C}^{r\times r}$, with entries on the diagonal given by

      $$[\boldsymbol{\Lambda}_0]_i = \lambda_i(\boldsymbol{D}) - \lambda,\ \ i = 1, ..., r,$$

  where $\lambda = \frac{1}{m}\sum_{i=1}^m z_i$ and $\lambda_i(\boldsymbol{D})$ is the $i^{\text{th}}$ largest eigenvalue of $\boldsymbol{D}$.
  return $\boldsymbol{U}_0 = \boldsymbol{Z}_0\boldsymbol{\Lambda}_0^{1/2}$.
```

## 3.3 Numerical Experiments

In this section, we provide the empirical performance of various amplitude-based algorithms that we consider, with comparisons to gradient descent based on the intensity-based loss function [64]. We will first compare their statistical performance in terms of sample complexities, and then compare their computational performance in terms of wall-clock time taken to achieve a desired accuracy. Extensive experiments are conducted over a wide range of settings of problem dimensions; here we report the most representative results that we find are consistent over different problem instances.

For each run, the entries of the sensing vectors $\boldsymbol{a}_i$ are generated i.i.d. using complex-valued Gaussian variables $\mathcal{N}(\mathbf{0}, \frac{1}{2}\boldsymbol{I}_n) + j\mathcal{N}(\mathbf{0}, \frac{1}{2}\boldsymbol{I}_n)$. The performance is measured using the Normalized Mean Squared Error (NMSE), defined as

$$\text{NMSE} = \frac{\|\boldsymbol{U}\boldsymbol{U}^H - \boldsymbol{X}\boldsymbol{X}^H\|_F}{\|\boldsymbol{X}\boldsymbol{X}^H\|_F}, \tag{3.13}$$

where $\boldsymbol{U}$ is the estimate of an algorithm. For GD and mini-batch SGD with the amplitude loss function, we use a constant step size of $\mu = 0.8$, whereas for GD with the intensity loss function, we use a constant step size of $\mu = 0.13$, which are optimized for convergence consistency. Additionally, we pick a mini-batch size of $B = \lceil \frac{n}{3} \rceil$ for SGD, which performs the best in terms of convergence time.

### 3.3.1 Comparisons of Spectral Initialization Methods

We start by comparing the performance of spectral initialization using amplitude and intensity measurements. In order to justify the use of amplitude measurements with spectral initialization, we ran an experiment of $10,000$ trials to compare the empirical distribution of NMSE using the amplitude-based and intensity-based spectral initialization, as shown in Figure 3.2. The amplitude-based spectral initialization provides a lower NMSE and has a much lower variance than the intensity-based spectral initialization. For the rest of the simulations, we will use the amplitude-based spectral initialization.

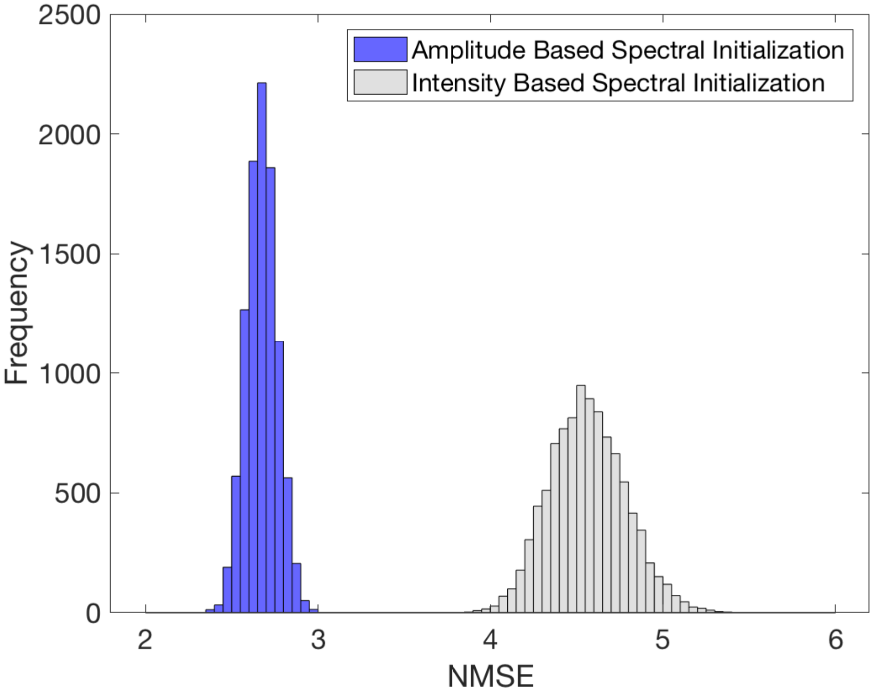


Figure 3.2: The empirical distribution of NMSE for spectral initialization constructed using amplitude and intensity measurements.

### 3.3.2 Comparisons of Statistical Performance

An important metric for the proposed algorithms is the minimum number of measurements required in order to recover the ground truth. Fix $n = 50$ and $r = 4$. We vary the number of measurements and measure the empirical success rate of each algorithm over $50$ Monte Carlo simulations. A trial is labeled as a success when NMSE $\leq 10^{-5}$. Figure 3.3 shows the empirical success rate with respect to the sampling ratio $m/(nr)$ using the same spectral initialization, for GD using the amplitude-based loss, GD using the intensity-based loss, mini-batch SGD using the amplitude-based loss, and alternating minimization. All the algorithms achieve perfect recovery as long as the sampling ratio is large enough; moreover, all of three proposed algorithms experience a phase transition using fewer measurements than GD using the intensity-based loss, indicating the benefit of the amplitude-based loss. Finally, the mini-batch SGD outperforms the rest of the algorithms, which is consistent with the observation in the phase retrieval case [110].

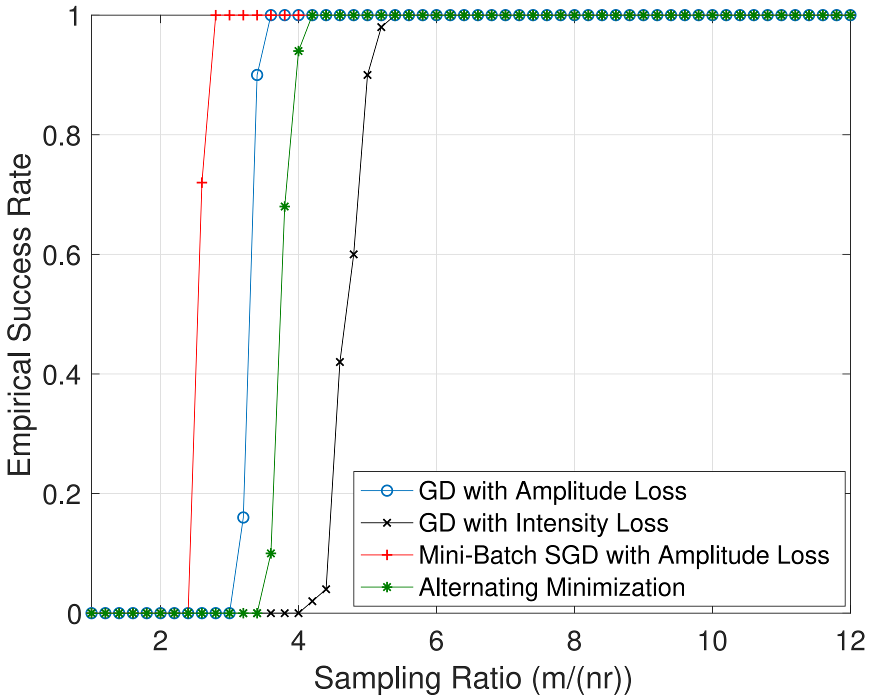


Figure 3.3: The empirical success rate with respect to the sampling ratio $m/(nr)$ for various algorithms, when $n = 50$ and $r = 4$.

### 3.3.3 Comparisons of Computational Performance

We next compare the computational performance of each algorithm, by demonstrating the decrease in NMSE as time passes. We track the convergence in terms of wall-clock time as opposed to the number of iterations to avoid the confounding variable of time-per-iteration. All experiments were run on a MacBook Air with a 2.2 GHz Intel Core i7 and 8 GB of 1600 MHz DDR3 RAM.

Figure 3.4 shows the NMSE with respect to the wall-clock time when $n = 50$, $r = 4$, and $m = 800$ using the same spectral initialization for the same set of algorithms as in Figure 3.3. Again, to achieve the same accuracy in terms of NMSE, GD using the intensity-based loss requires more time than all three algorithms using the amplitude-based loss. In particular, the mini-batch SGD requires significantly less time than GD or alternating minimization, due to its much faster execution per iteration.

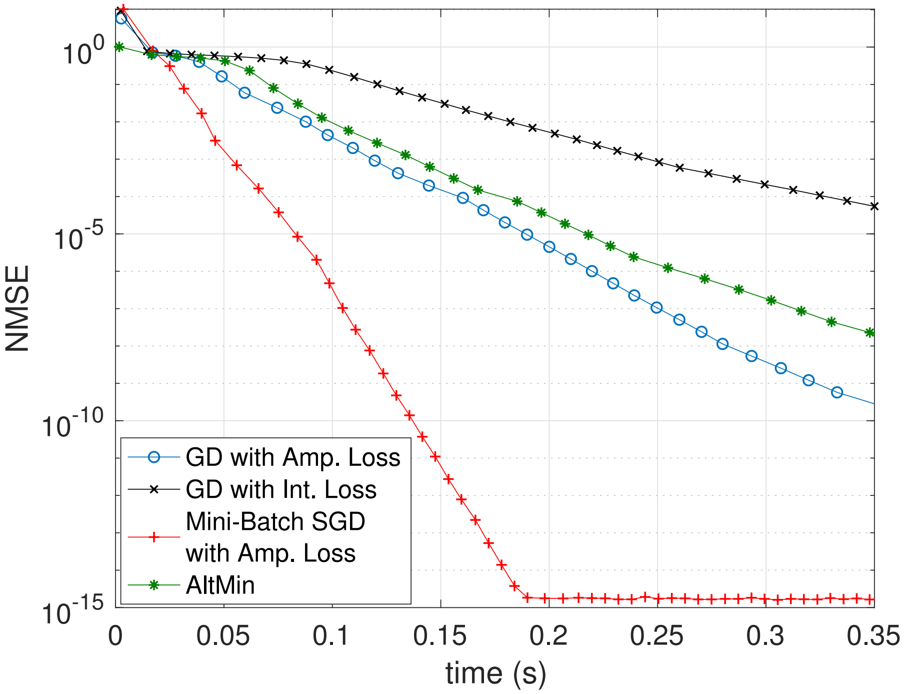


Figure 3.4: The NMSE with respect to wall-clock time for various algorithms, when $n = 50$, $r = 4$ and $m = 800$.

## 3.4 Scaled Gradient Methods for Estimating Ill-Conditioned Matrices

Not all $r$-column matrices are created equal. Indeed, recalling $\boldsymbol{M} = \boldsymbol{X}\boldsymbol{X}^H$, denote the eigenvalue decomposition of $\boldsymbol{M}$ as $\boldsymbol{M} = \boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{V}^H$, where $\boldsymbol{V} \in \mathbb{R}^{n\times r}$ is the matrix of eigenvectors of $\boldsymbol{M}$, and $\boldsymbol{\Sigma} \in \mathbb{R}^{r\times r}$ is a diagonal matrix consisting of the $r$ eigenvalues of $\boldsymbol{M}$ organized in non-increasing order, i.e., $\Sigma_{1,1} \geq \Sigma_{2,2} \geq \cdots \geq \Sigma_{r,r} > 0$. Let the condition number of the matrix $\boldsymbol{M}$ be defined as,

$$\kappa = \Sigma_{1,1}/\Sigma_{r,r}. \tag{3.14}$$

**Algorithm 3.5** Scaled Gradient Descent (ScaledGD) with Amplitude Loss

```
Input: ${z_i}_{i=1}^m$, ${\boldsymbol{a}_i}_{i=1}^m$, step size $\mu_k$
    Initialization:
    Obtain $\boldsymbol{U}_0$ from Spectral Initialization (Algorithm 3.4).
    Gradient Updates:
    for $k = 0, 1, 2, ..., K-1$ do
        $\boldsymbol{W}_k = \operatorname{diag}\left(\left[\frac{\|\boldsymbol{a}_1^H \boldsymbol{U}_k\|_2 - z_1}{\|\boldsymbol{a}_1^H \boldsymbol{U}_k\|_2}, ..., \frac{\|\boldsymbol{a}_m^H \boldsymbol{U}_k\|_2 - z_m}{\|\boldsymbol{a}_m^H \boldsymbol{U}_k\|_2}\right]\right).$
        $\boldsymbol{U}_{k+1} = \boldsymbol{U}_k - \frac{2\mu_k}{m} \boldsymbol{A}^H \boldsymbol{W}_k \boldsymbol{A} \boldsymbol{U}_k \left(\boldsymbol{U}_k^H \boldsymbol{U}_k\right)^{-1}.$
    end for
    return $\boldsymbol{U}_K$.
```

Generally speaking, the greater the value of $\kappa$, the more "ill-conditioned" a matrix is. It is well-known that the iteration complexity of an iterative algorithm such as GD grows proportionally with the condition number of a matrix. As it turns out, the algorithms proposed within this chapter suffer from the same relationship between the condition number and the convergence speed. Recently, a nascent idea in the matrix factorization literature called Scaled Gradient Methods have been proposed to accelerate ill-conditioned low-rank matrix estimation [95, 96, 97]. The authors introduce the notion of preconditioning the gradient update by scaling the update rule as in quasi-Newton type algorithms. In a nutshell, the preconditioners are computed by inverting an $r \times r$ matrix and back-multiplying the gradient by the preconditioner.

Algorithm 3.5 outlines how to incorporate pre-conditioning into the GD algorithm with the amplitude-based loss function. We demonstrate the improved performance of ScaledGD with amplitude-based loss function for estimating ill-conditioned matrices in Figure 3.5. The experimental set up mirrors that of Section 3.3.3. It is immediately apparent that the condition number has a large impact on the number of iterations required in order for the vanilla GD algorithm to converge. However, the ScaledGD algorithm converges in roughly the same number of iterations independent of the condition number of the matrix, in line with its performance in [95, 96, 97]. The computational overhead incurred from inverting an $r \times r$ matrix is dominated

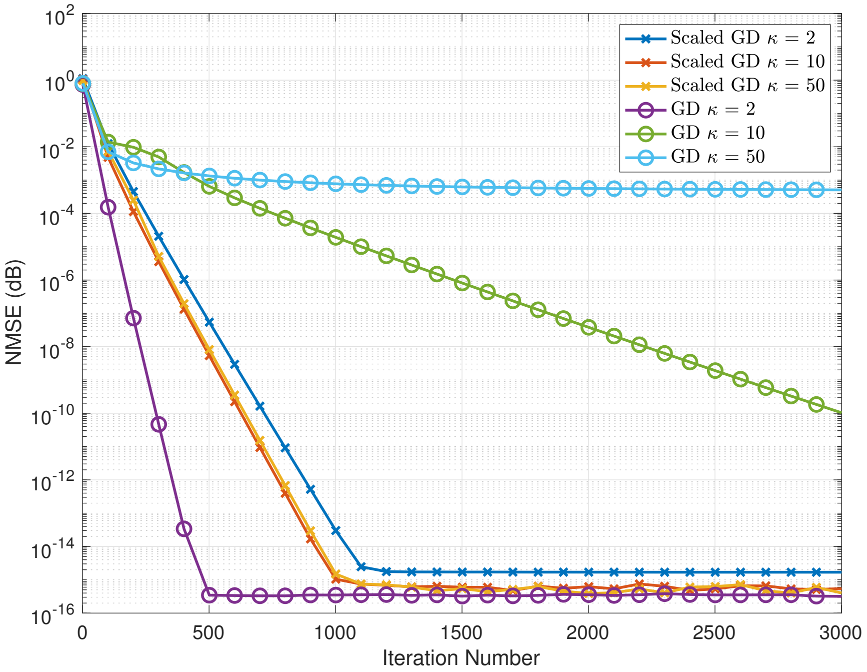


Figure 3.5: The NMSE with respect to iteration count for comparing GD (with Amp. Loss) with its Scaled version. Here, $n = 50$, $r = 4$ and $m = 800$. It can be seen that ScaledGD converges much faster than GD, especially for large condition numbers.

by the complexity of computing the gradient, and is easily outweighed by the improvement in the number of iterations required to converge. Generalization of the other algorithms proposed in this chapter to use preconditioning in the update rule in order to estimate ill-conditioned matrices is omitted for brevity.

## 3.5 Conclusion

We propose new non-convex approaches for solving quadratic sensing, or equivalently, low-rank matrix recovery from rank-one measurements, which all utilize an amplitude-based loss function as opposed to an intensity-based loss function. Additionally, we demonstrate how to include preconditioners to recover ill-conditioned matrices. Numerical experiments are provided to demonstrate their advantages over using the intensity-based loss function in terms of both sample complexity and computational complexity. In the future, we would like to provide theoretical analysis to guarantee the convergence of each of these algorithms to the global optimum solution as long as the sample size is sufficiently large. Another interesting venue of research would be to demonstrate the stability of these algorithms to noise and/or the presence of outliers in the measurements [109] using random initialization [21].

# Chapter 4

# Plug-and-Play Image Reconstruction Meets Stochastic Variance-Reduced Gradient Methods

Plug-and-play (PnP) methods have recently emerged as a powerful framework for image reconstruction that can flexibly combine different physics-based observation models with data-driven image priors in the form of denoisers, and achieve state-of-the-art image reconstruction quality in many applications. In this chapter, we aim to further improve the computational efficacy of PnP methods by designing a new algorithm that makes use of stochastic variance-reduced gradients (SVRG), a nascent idea to accelerate runtime in stochastic optimization. Compared with existing PnP methods using batch gradients or stochastic gradients, the new algorithm, called PnP-SVRG, achieves comparable or better accuracy of image reconstruction at a much faster computational speed. Extensive numerical experiments are provided to demonstrate the benefits of the proposed algorithm through the application of compressive imaging using partial Fourier measurements and phase retrieval, in conjunction with a wide variety of popular image denoisers.

## 4.1 Introduction

We consider the reconstruction of an image $\boldsymbol{x}^\star \in \mathbb{R}^n$ from a set of noisy measurements $\boldsymbol{y} \in \mathbb{C}^m$, formulated as

$$\boldsymbol{y} = \mathcal{A}\left(\boldsymbol{x}^\star\right) + \boldsymbol{\varepsilon}, \tag{4.1}$$

where $\mathcal{A} : \mathbb{R}^n \mapsto \mathbb{C}^m$ characterizes the *forward model* of the imaging system known *a priori*, $\boldsymbol{y} = \{y_i\}_{1\leq i\leq m}$ is the set of measurements, and $\boldsymbol{\varepsilon} = \{\varepsilon_i\}_{1\leq i\leq m}$ is additive (random) noise present in the system. In many applications, the forward model is ill-posed with more unknowns than the number of measurements, i.e. $m \ll n$, due to limited sampling budget or acquisition time. Without prior knowledge about the structure of the ground truth image, the problem is generally impossible to solve. Therefore, one would like to derive image reconstruction algorithms that exploit prior knowledge in the image in order to generate a proper estimate, $\hat{\boldsymbol{x}}$, of the image.

Image denoising and reconstruction algorithms have been thoroughly studied. While classically image priors are designed using hand-crafted features such as wavelets, many data-driven image priors based on deep learning and neural networks have been proposed in recent years [2, 18, 30, 50]. A number of techniques have been proposed, such as algorithm unrolling, using neural networks as generative models of images, end-to-end training, and so on; see [3, 77, 81] for recent overviews.

This chapter focuses on the plug-and-play (PnP) methods for image reconstruction, first proposed in [103]. PnP methods have recently emerged as a powerful framework for image reconstruction that can flexibly combine different physics-based observation models with data-driven image priors in the form of denoisers, and achieve state-of-the-art image reconstruction quality in many applications [89, 90]. To motivate the methodology of PnP, note that many classical approaches can be viewed as solving an optimization problem that is composed of a data-fidelity term that incorporates the observation model and a regularization term that incorporates image priors. Proximal algorithms such as proximal gradient methods and alternating direction method of multipliers are popular solvers, where the proximal map avoids the need to find derivatives of complicated image regularizers which may be non-differentiable. In [103], the authors made the keen observation that the proximal map is equivalent to finding the *maximum a posteriori* (MAP) image denoising estimate upon a suitable probability prior. Bearing this perspective, PnP methods replace this proximal update by "plugging in" an image denoiser of the user's choice, such as non-local means (NLM) [12], block matching 3D filtering (BM3D) [29], or of particular interest, a state-of-the-art convolutional neural network (CNN) based image denoiser [85].

Generally, the PnP framework allows the injection of sophisticated denoisers into iterative reconstruction algorithms, where first-order algorithms are particularly appealing due to their scalability to large-scale problem instances and amenability to computing advances. When combined with gradient descent (GD), this leads to the PnP-GD algorithm which alternates between taking a GD step with respect to the data-fidelity term, followed by a denoising step using the plugged-in denoiser of choice [55]. Moreover, the data-fidelity term is often given as a finite sum of sample losses over each observation, e.g. when the observations are collected in an independently and identically distributed manner. Consequently, one can leverage the finite sum structure to design PnP methods that take advantage of *stochastic* algorithms such as Stochastic Gradient Descent (SGD), which was done in [92] to derive the PnP-SGD algorithm for image reconstruction. In [92], the authors demonstrated that PnP-SGD achieves a considerable speed up in computation time, while achieving the same reconstruction quality as PnP-GD.

The work presented in Chapter 4 was published in part at IEEE ICIP 2021 [74]. The main contribution of this chapter is to introduce the use of stochastic variance-reduced gradient (SVRG) methods [53] into the PnP framework. It is well-known from the optimization literature that SGD tends to converge with a much larger number of iterations compared with GD, due to the higher variability in its search direction, while the per-iteration cost of GD is expensive when the data size is large. The crux of SVRG [53] is a carefully-designed stochastic gradient with reduced variance by using a reference batch gradient that is periodically updated to save computation. Therefore, it is of great interest to investigate whether SVRG brings additional benefits to the PnP framework. We propose a new algorithm called PnP-SVRG, which can be regarded as replacing the proximal map in the proximal SVRG algorithm [107] by a wide variety of popular image denoisers including NLM, CNN and BM3D. We demonstrate through extensive numerical experiments that compared with PnP-SGD and PnP-GD, PnP-SVRG achieves comparable or better accuracy of image reconstruction at a faster computational speed.

The rest of this chapter is organized as follows. First, we provide backgrounds on GD-based algorithms for PnP in Section 4.2. We develop the stochastic variance-reduction based PnP algorithm, called PnP-SVRG, in Section 4.3. We then evaluate the performance of the proposed PnP-SVRG algorithm in the context of compressive image reconstruction using partial 2D Fourier measurements as well as phase retrieval in Section 4.4, with comparisons to PnP-GD and PnP-SGD. In Section 4.5, we conclude our findings and discuss future directions.

## 4.2 Backgrounds

In this section, we first introduce the PnP-GD and PnP-SGD algorithms, and then discuss the use of stochastic variance-reduced gradient methods in finite-sum stochastic optimization.

### 4.2.1 Plug-and-Play with GD and SGD

To solve for the image from the observations (4.1), we start with a data-fidelity loss function that takes the following finite-sum form:

$$\ell(\boldsymbol{x}) = \frac{1}{m}\sum_{i=1}^{m} \ell_i(\boldsymbol{x}), \tag{4.2}$$

where $\ell_i(\boldsymbol{x}) = \ell_i(\boldsymbol{x}; y_i)$ depends on the $i$th measurement. A popular choice is minimizing the least-squares difference between the observed measurements and those produced by the current image estimate. A common approach to incorporate image priors is to solve the regularized optimization problem:

$$\min_{\boldsymbol{x}} \; \ell(\boldsymbol{x}) + \lambda g(\boldsymbol{x}), \tag{4.3}$$

where $g(\cdot)$ is the regularizer and $\lambda > 0$ is some regularization parameter. The above optimization problem can be solved by proximal gradient methods, which update the image estimate as

$$\boldsymbol{x}_t = \mathrm{prox}_{\lambda g}(\boldsymbol{x}_{t-1} - \eta \nabla \ell(\boldsymbol{x}_{t-1})), \tag{4.4}$$

where $\mathrm{prox}_g$ is the proximal map of $g(\cdot)$, $\nabla \ell(\boldsymbol{x})$ is the gradient of $\ell(\boldsymbol{x})$ and $\eta > 0$ is a proper learning rate. The key step of PnP methods is to replace the proximal map with respect to some choice of regularization function $g(\cdot)$ by an image denoising routine of choice. More specifically, we define an image denoiser

$$\mathrm{denoise}_{\hat{\sigma}}(\boldsymbol{x}) : \mathbb{R}^n \mapsto \mathbb{R}^n,$$

which is parameterized by some estimate of the noise level present in the image $\hat{\sigma} > 0$. Plugging the above denoiser into (4.4) leads directly to the PnP-GD algorithm in Algorithm 4.1. Similarly, leveraging the finite-sum form of the data-fidelity term leads to the PnP-SGD algorithm [92] in Algorithm 4.2, where a mini-batch of $B$ measurements can be used to construct the search direction to balance the convergence speed and per-iteration computational cost.

While the PnP method yields a simple form, its theoretical performance is still largely unknown and remains an active field of study. The authors of [90] proposed a set of conditions such that a denoising routine can be handled similarly to a Moreau proximal mapping. In [85], the authors borrowed from operator theory to prove convergence for strongly convex loss functions with properly trained CNN-based image denoisers. In a related approach, regularization by denoising [82] can be used to prove convergence of a slightly broader class of algorithms including PnP methods.

### 4.2.2 Stochastic Variance-Reduced Gradient Methods

SVRG [53], together with many variants and offsprings, has become a popular algorithmic approach in finite-sum optimization to achieve a better runtime complexity than both GD and SGD. In a nutshell, to optimize $\ell(\boldsymbol{x})$ in (4.2), it updates the parameter along the direction of the

**Algorithm 4.1** PnP-GD

```
Input: $\boldsymbol{x}_0, \eta, T, \hat{\sigma}$.
Initialize: $\boldsymbol{x}_0$.
for $t = 1, 2, \ldots, T$ do
    $\boldsymbol{z}_t = \boldsymbol{x}_{t-1} - \eta \nabla \ell(\boldsymbol{x}_{t-1})$;
    $\boldsymbol{x}_t = \text{denoise}_{\hat{\sigma}}(\boldsymbol{z}_t)$.
end for
Output: $\hat{\boldsymbol{x}} = \boldsymbol{x}_T$.
```

**Algorithm 4.2** PnP-SGD

```
Input: $\boldsymbol{x}_0, \eta, T, B, \hat{\sigma}$.
Initialize: $\boldsymbol{x}_0$.
for $t = 1, 2, \ldots, T$ do
    pick a set $\mathcal{I}_t \subset \{1, \ldots, m\}$ of cardinality $B$ uniformly at random;
    $\boldsymbol{z}_t = \boldsymbol{x}_{k-1} - \frac{\eta}{B} \sum_{i \in \mathcal{I}_t} \nabla \ell_i(\boldsymbol{x}_{t-1})$;
    $\boldsymbol{x}_t = \text{denoise}_{\hat{\sigma}}(\boldsymbol{z}_t)$.
end for
Output: $\hat{\boldsymbol{x}} = \boldsymbol{x}_T$.
```

stochastic gradient given below,

$$\boldsymbol{v}_t = \nabla \ell_{i_t}(\boldsymbol{x}_{t-1}) - \nabla \ell_{i_t}(\tilde{\boldsymbol{x}}) + \nabla \ell(\tilde{\boldsymbol{x}}), \tag{4.5}$$

where $\tilde{\boldsymbol{x}}$ is a *reference point* stored during the beginning of each epoch of the algorithm. On the one end, this ensures the stochastic gradient $\boldsymbol{v}_t$ is unbiased, since $\mathbb{E}[\boldsymbol{v}_t] = \nabla \ell(\boldsymbol{x}_{t-1})$, when $i_t$ is selected uniformly at random from $\{1, \ldots, m\}$. On the other end, the inclusion of the reference helps reduce the variance of $\boldsymbol{v}_t$ especially when the current iterate is not too far from the reference point. Thus, by periodically updating the reference point, SVRG accelerates the convergence speed; in fact, SVRG achieves linear convergence when $\ell(\boldsymbol{x})$ is strongly convex [53]. The SVRG algorithm can be extended to the proximal setting when minimizing the sum of a finite-sum

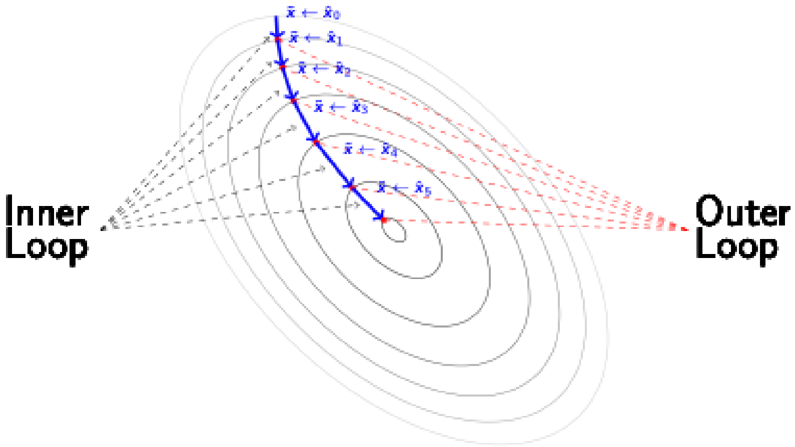

$\tilde{x} \leftarrow \tilde{x}_0$
$\tilde{x} \leftarrow \tilde{x}_1$
$\tilde{x} \leftarrow \tilde{x}_2$
$\tilde{x} \leftarrow \tilde{x}_3$
$\tilde{x} \leftarrow \tilde{x}_4$
$\tilde{x} \leftarrow \tilde{x}_5$
Inner Loop
Outer Loop

Figure 4.1: A visual summary of the SVRG algorithm. In each outer loop, a reference point is chosen, and one full gradient is computed with respect to the reference point. Then, within each inner loop, only stochastic gradients are computed to determine the descent direction. The full gradient computation is used within each inner loop calculation.

term and a regularization term [107]. The resulting algorithm alternates between a stochastic variance-reduced gradient step and a proximal map at each iteration, and achieves a similar runtime speed-up.

## 4.3 Plug-and-Play with Variance Reduction

Motivated by the success of SVRG in accelerating finite-sum optimization, we propose a novel algorithm that combines SVRG with PnP methods to solve image reconstruction. The proposed algorithm, dubbed PnP-SVRG, is presented in Algorithm 4.3. Specifically, the algorithm contains two loops. The outer loop, sometimes referred to as an epoch, is used to update the reference point $\tilde{\boldsymbol{x}}$ as well as its full gradient $\boldsymbol{w} = \ell(\tilde{\boldsymbol{x}})$ in a periodic manner. The inner loop, on the other end, performs stochastic gradient updates using the variance-reduced stochastic gradient $\boldsymbol{v}_t$, which can be computed in mini-batches to further reduce variance and improve performance.

The proposed PnP-SVRG algorithm can be regarded as replacing the proximal map in the proximal SVRG algorithm [107] by a wide variety of popular image denoisers such as CNN, BM3D, NLM, and so on, which will be examined in detail in Section 4.4.

**Algorithm 4.3** PnP-SVRG

```
Input: $\boldsymbol{x}_0, \eta, T_1, T_2, B, \hat{\sigma}$.
Initialize: $\boldsymbol{x}_0$.
for $s = 1, 2, \ldots, T_1$ do
    $\tilde{\boldsymbol{x}} = \boldsymbol{x}_{s-1}$;
    $\boldsymbol{w} = \nabla \ell(\tilde{\boldsymbol{x}})$;
    $\boldsymbol{z}_0 = \tilde{\boldsymbol{x}}$.
    for $t = 1, 2, \ldots, T_2$ do
        pick a set $\mathcal{I}_t \subset \{1, \ldots, m\}$ of cardinality $B$ uniformly at random;
        $\boldsymbol{v}_t = \frac{1}{B} \sum_{i \in \mathcal{I}_t} (\nabla \ell_i(\boldsymbol{z}_{t-1}) - \nabla \ell_i(\tilde{\boldsymbol{x}})) + \boldsymbol{w}$;
        $\boldsymbol{z}_t = \text{denoise}_{\hat{\sigma}}(\boldsymbol{z}_{t-1} - \eta \boldsymbol{v}_t)$.
    end for
    $\boldsymbol{x}_s = \boldsymbol{z}_{T_2}$.
end for
Output: $\hat{\boldsymbol{x}} = \boldsymbol{x}_{T_1}$.
```

It is worth emphasizing that Algorithm 4.3 is composed of $T_1$ epochs (outer loops) which each contains $T_2$ stochastic gradient and denoising updates. Within each epoch, we compute the full gradient of a reference point once which is used within the inner loop for stochastic gradient updates. Overall, we perform $T_1$ full gradient computations and $2T_1T_2$ stochastic gradient computations and $T_1T_2$ denoising operations. It is critical to leverage a parameter configuration such that the benefit of the SVRG methods outweighs the computation cost of calculating a full gradient in each epoch, particularly when the denoising operation is expensive. We include a visual summary of the SVRG algorithm in Figure 4.1.

## 4.4 Numerical Experiments

In this section, we examine the performance of the proposed PnP-SVRG algorithm for two different sensing models. First, we investigate compressive image reconstruction using partial 2D Fourier measurements, a common setting used in compressive magnetic resonance imaging (MRI). Second, we explore reconstructing an image from under-sampled, phaseless measurements. In each of these experiments, the number of measurements is less than the dimensions of the image; this leads to a regime where image priors are necessary.

We evaluate the reconstruction quality using the standard metric Peak Signal-to-Noise Ratio (PSNR), measured in decibels, between the original image $\boldsymbol{X}^\star$ and the recovered image $\hat{\boldsymbol{X}}$. Suppose that the images of interest are of size $n_1 \times n_2$ pixels. Then, the PSNR is calculated as,

$$\text{PSNR} = -10 \log_{10} \left( \frac{1}{n_1 n_2} \sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \left( X^\star_{i,j} - \hat{X}_{i,j} \right)^2 \right) \text{dB}, \tag{4.6}$$

assuming that each image is normalized such that values are within the range of $[0, 1]$.

We consider a few choices of image denoisers, including NLM, BM3D, TV-norm based wavelet denoising, and our own trained CNN. We implement a decay on the learning rate and the noise level estimate to improve stability. For CNN, we use a denoising CNN (DnCNN) [111], which follows a U-Net structure composed of a series of successive convolutions that first increase then decrease the input image's channels in order to separate the additive noise from the noisy image. The DnCNN denoiser we train is based off of the "realSN-DnCNN" introduced in [85], which uses a variation of layer-wise spectral norm to train a DnCNN. We train our denoiser on a truncated version of the Flickr30k dataset [108], using only 1000 images to train and 600 images to validate our results, where all RGB images are first converted to grayscale before training. We train the DnCNN on a variety of noise levels: $\sigma \in [\frac{10}{255}, \frac{20}{255}, \frac{30}{255}, \frac{40}{255}, \frac{50}{255}]$ and find that $\sigma = \frac{40}{255}$ produces the best results when denoising. Each network is trained on a Titan RTX for 20 epochs, taking no more than 5 minutes to train.

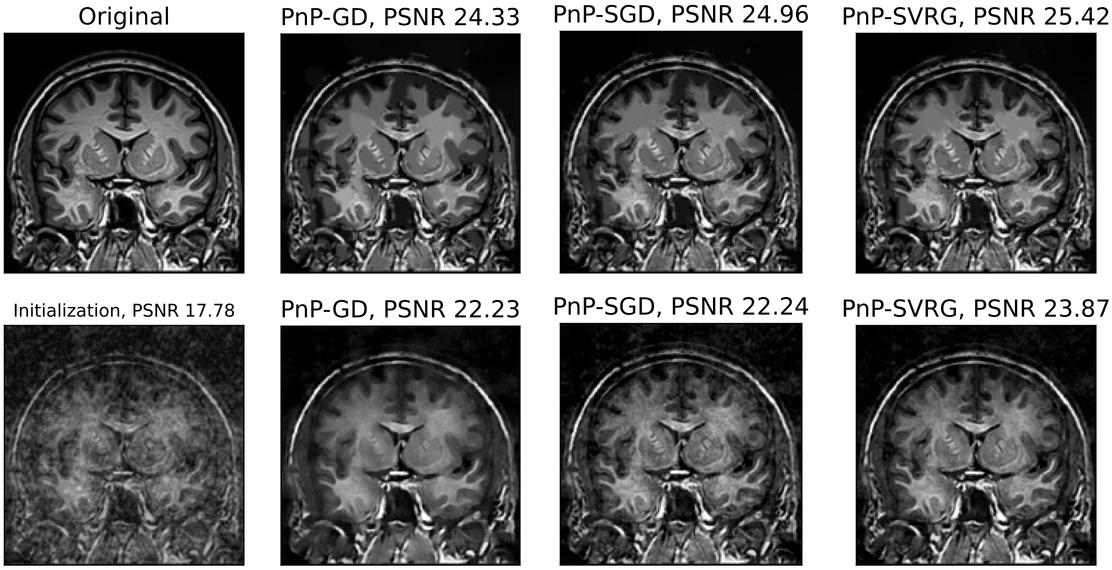
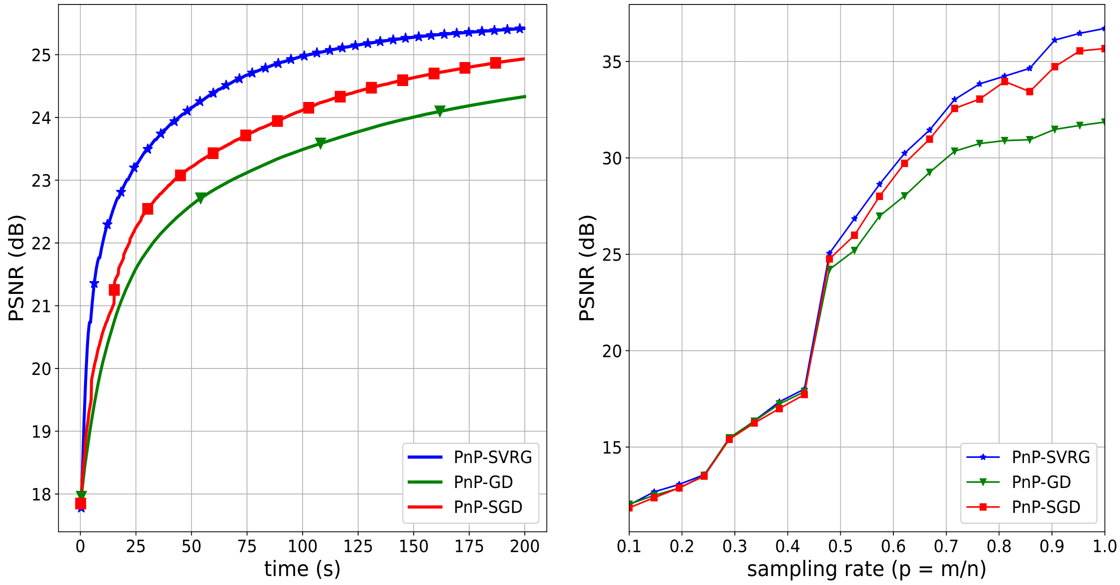


Figure 4.2: Performance of the proposed PnP-SVRG algorithm in comparison with PnP-GD and PnP-SGD for image reconstruction using partial 2D DFT measurements at a sampling rate $p = 0.5$. After the first column, the first row uses the NLM denoiser and the second row uses the CNN denoiser. The PSNR values are reported in dB.

### 4.4.1 Compressive Sensing Magnetic Resonance Imaging

Consider the problem where our goal is to recover an image from bilinear measurements of an orthonormal matrix and its matrix transpose. This captures examples using the discrete Fourier matrix, discrete cosine matrix, discrete wavelet matrix, and many more. We will assume that images are square, but the process can be easily generalized. Within the context of this section, the image $\boldsymbol{X}^\star \in \mathbb{R}^{n\times n}$ will be treated as a matrix, and the measurements $\boldsymbol{Y} \in \mathbb{R}^{n\times n}$ will be zero-padded to correspond with the image size. The matrix $\boldsymbol{F} \in \mathbb{C}^{n\times n}$ is the discrete Fourier transform (DFT) matrix, $\boldsymbol{M} \in \{0, 1\}^{n\times n}$ is a binary mask which determines the measurements that are sampled, and $\boldsymbol{\varepsilon}$ represents the noise in the system. We define the sampling rate $p$ to be equal to the ratio of the number of observed measurements to the size of the image. We arrive at the following measurement model:

$$\boldsymbol{Y} = \boldsymbol{M} \odot (\boldsymbol{F}\boldsymbol{X}^\star\boldsymbol{F}^\top + \boldsymbol{\varepsilon}). \tag{4.7}$$



(a) (b)

Figure 4.3: The PSNR of image reconstruction via the proposed PnP-SVRG algorithm with the NLM denoiser in comparison to PnP-GD and PnP-SGD. (a) The reconstructed PSNR with respect to runtime at a sampling rate of 50%. (b) The reconstructed PSNR of the final image estimate with respect to the sampling rate after running each algorithm for 200 seconds.

We formulate a natural squared loss function based on the Frobenius norm difference between the observed measurement matrix and the 2-D Fourier transform of the image, as

$$\ell(\boldsymbol{X}) = \frac{1}{2m}\|\boldsymbol{Y} - \boldsymbol{M} \odot \boldsymbol{F}\boldsymbol{X}\boldsymbol{F}^\top\|_F^2. \tag{4.8}$$

Clearly, this loss function can be written as a finite sum over the observed measurements, and thus fits the algorithmic framework of PnP-SVRG. Throughout the experiments, we take an image of size $256 \times 256$, and the noise in (4.7) is generated as additive white Gaussian noise with a standard deviation of $\sigma = 5$. For PnP-SVRG and PnP-SGD, we utilize a mini-batch size of $B = 2000$.

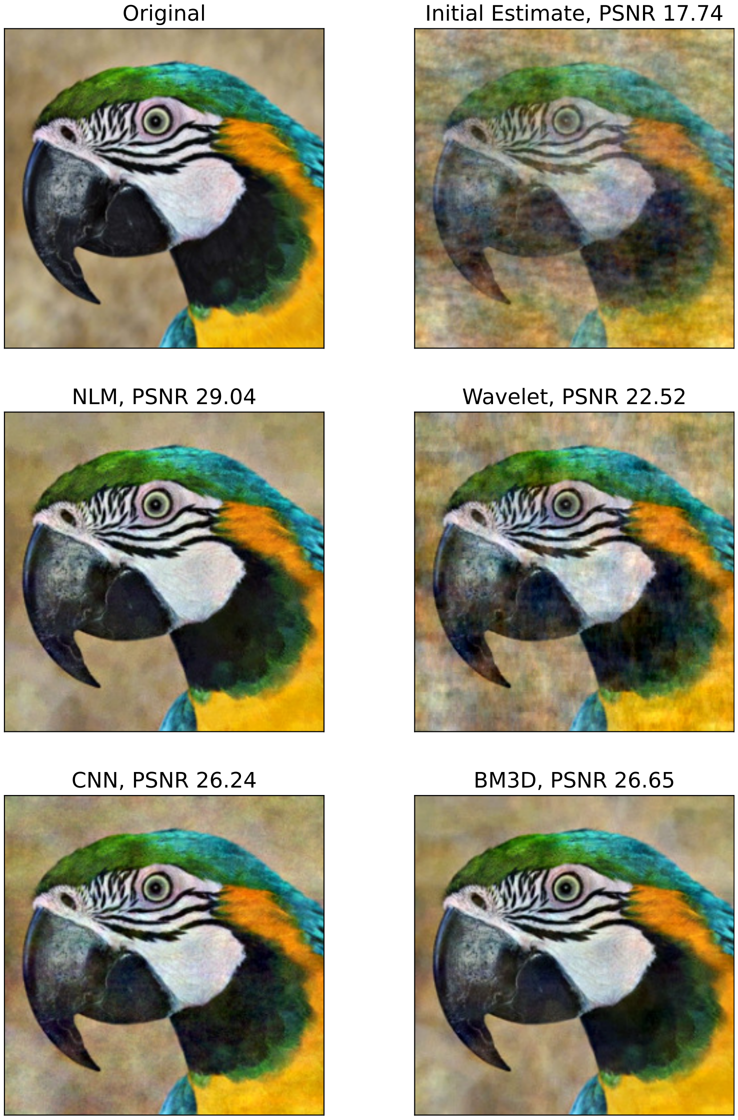


Figure 4.4: The reconstructed PSNR using the proposed PnP-SVRG algorithm with different denoising routines for an RGB image with a sampling rate $p = 0.5$.

In Figure 4.2, we evaluate the performance of the proposed PnP-SVRG algorithm in comparison with PnP-GD and PnP-SGD for image reconstruction at a sampling rate $p = 0.5$, using the NLM denoiser and the CNN denoiser. It can be seen that PnP-SVRG achieves slightly higher PSNR than PnP-GD and PnP-SGD. Figure 4.3 further examines the performance of image reconstruction via the proposed PnP- SVRG algorithm with the NLM denoiser in comparison to PnP-GD and PnP-SGD in terms of the runtime in (a), and sampling rate in (b). It can be seen that the proposed PnP-SVRG achieves a significantly improved runtime compared to its competitors, as well as slightly better reconstruction quality when the sampling rate is larger than $0.5$. Similar trends are also observed for other denoisers. Lastly, Figure 4.4 demonstrates the reconstruction quality of PnP-SVRG with various image denoisers for an RGB image when $p = 0.5$. Each RGB channel is processed separately and then combined for rendering. Here, the CNN, NLM, and BM3D based denoisers all achieve high visual quality. In conclusion, PnP-SVRG can be successfully implemented with a wide range of denoising routines.

### 4.4.2 Phase Retrieval

Consider the problem of reconstructing an $n$-pixel image, $\boldsymbol{x}^\star \in \mathbb{R}^n$, from $m$ phaseless measurements, $\{y_i\}_{i=1}^m$. The measurements are obtained through an ensemble of sensing vectors, $\{\boldsymbol{a}_i\}_{i=1}^m$, according to the model,

$$y_i = \left|\langle \boldsymbol{a}_i, \boldsymbol{x}^\star \rangle\right|^2 + \varepsilon_i,\ i = 1, \cdots, m, \tag{4.9}$$

where, $\{\varepsilon_i\}_{i=1}^m$, is additive white Gaussian noise with zero mean and standard deviation, $\sigma > 0$. Define $z_i = \sqrt{y_i},\ \forall i = 1, \cdots, m$. We adopt an amplitude-based loss function—see Chapter 3 for more details—based on the squared difference between the observed phaseless measurements and the image estimate as observed through the measurement model, i.e.,

$$\ell(\boldsymbol{x}) = \frac{1}{2m} \sum_{i=1}^{m} \left(z_i - |\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle|\right)^2. \tag{4.10}$$

Throughout these experiments, the image is made up of $128 \times 128$ pixels. The sensing vectors $\boldsymbol{a}_i$ are drawn independently, from a Gaussian distribution, $\boldsymbol{a}_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_n)$. The noise in (4.9) is generated as additive white Gaussian noise with a standard deviation of $\sigma = 0.5$. There is undersampling in the measurements, with $m = 0.5n$ measurements. For PnP-SVRG and PnP-SGD, we utilize a mini-batch size of $B = 512$. We employ a wavelet-based estimator of the Gaussian noise standard deviation [36], from the Scikit-image Python package [11], to be used as the noise-strength parameter of the BM3D image denoising routine [29]. The spectral method is used to generate an initial estimate; see Chapter 2 for more intuition. Since we are operating in

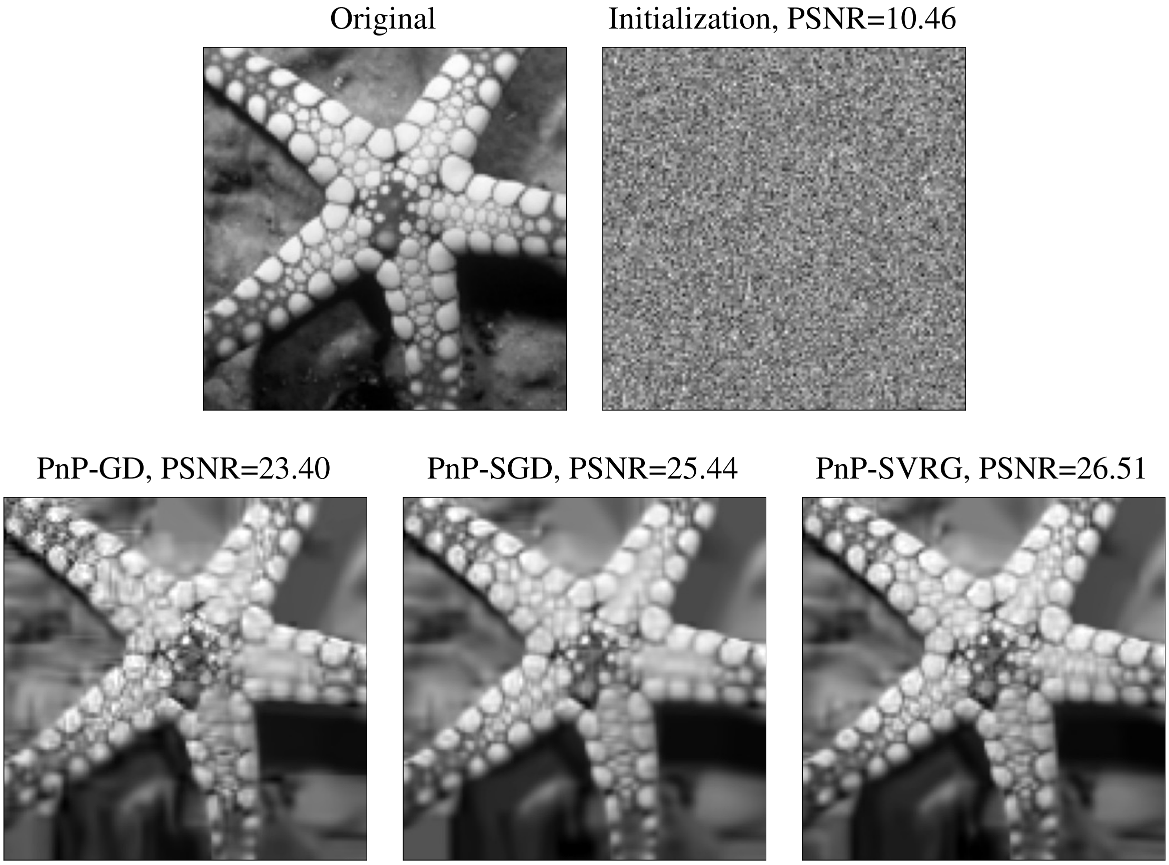


Figure 4.5: Performance of the proposed PnP-SVRG algorithm in comparison with PnP-GD and PnP-SGD for image reconstruction using phaseless measurements with $m = .5n$. All of the reconstructed images use the BM3D image denoiser.

an undersampled regime, the initial estimate is, visually, not particularly accurate. We run each algorithm for 30 seconds and report the results in terms of PSNR.

In Figure 4.5, we visualize the output of each of the algorithms using the BM3D image denoiser. With respect to the foreground, the PnP-SVRG algorithm is able to reconstruct finer details in the starfish in comparison to the other two algorithms. While the shadow of the starfish is generally recovered well, the rest of the background is not reconstructed particularly well. The improved computation speed of PnP-SVRG is again demonstrated when plotting the PSNR over runtime, in Figure 4.6.

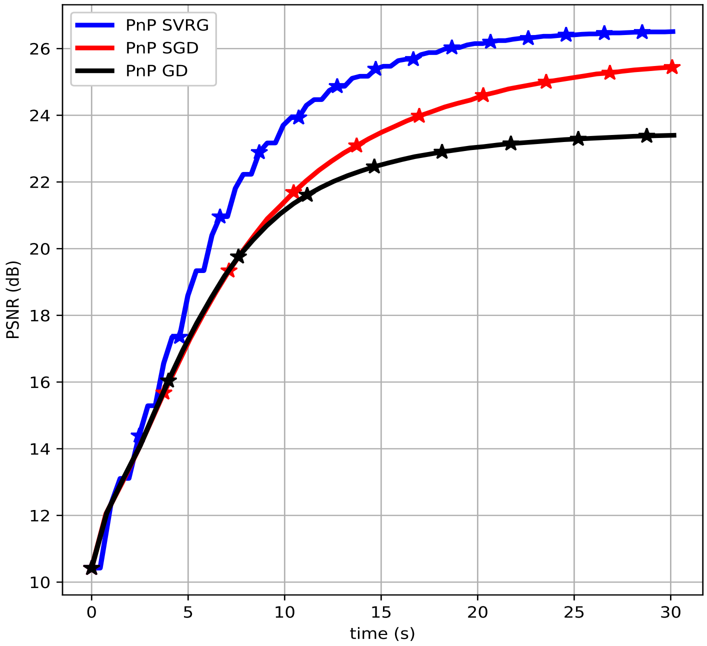


Figure 4.6: The PSNR of image reconstruction via the proposed PnP-SVRG algorithm with the BM3D denoiser in comparison to PnP-GD and PnP-SGD for phase retrieval. The reconstructed PSNR with respect to runtime at an undersampling rate of 50%.

## 4.5 Conclusion

This work proposes a new algorithm, dubbed PnP-SVRG, for image reconstruction, which carefully employs stochastic variance reduction techniques within the PnP framework. We demonstrated that substantially improved computational efficiency can be obtained without sacrificing image reconstruction quality. In the future, we plan to further investigate the performance of PnP-SVRG for other imaging tasks such as deblurring, super-resolution, and inpainting. In addition, other stochastic variance reduction algorithms such as SARAH [80] and SAGA [33] will also be examined within the PnP framework. Last but not least, we are interested in evaluating the performance of the proposed PnP-SVRG algorithm for real data applications such as electron microscopy imaging [90]. In the future we look to prove the convergence of PnP-SVRG to a fixed point, following similar lines of work such as [93].

# Chapter 5

# Joint Dictionary Indexing with Graph Regularization for Electron Back-Scattered Diffraction Patterns

Suppose that we are given a dictionary of known signals and a measured signal. Ideally, the observed measurements would be perfectly defined by a single signal in our dictionary. In practice, dictionaries are limited and there exists noise in the observation system, which leads to defining suboptimal measurements. In dictionary indexing, the goal is to reconstruct the coefficients of a linear combination of signals in our dictionary which accurately define the observed measurement.

Some applications require indexing a set of measurements by our dictionary. While one could index each measurement individually, we are interested in scenarios where the set of ground-truth signals which define the measurements share some relationship with each other that we wish to exploit. In pursuit of this goal, we demonstrate the applicability of leveraging graph structures for jointly dictionary indexing. We propose an ADMM-based algorithm for indexing the set of measurements in a joint fashion while exploiting both signal- and graph-based regularization. We apply our methodology to scanning electron microscopy and the process of indexing electron back-scattered diffraction patterns.

## 5.1 Introduction

In dictionary indexing (DI), measurements are constructed using a linear combination of basis vectors which make up the dictionary [112]. Let the set of $p$-dimensional signals $\{\boldsymbol{x}_i\}_{i=1}^{d}$ define our dictionary of $d$ entries. Then a (noiseless) measurement, $\boldsymbol{y} \in \mathbb{R}^p$, takes the form,

$$\boldsymbol{y} = \sum_{i=1}^{d} \beta_i \boldsymbol{x}_i, \tag{5.1}$$

where the set of coefficients, $\{\beta_i\}_{i=1}^{d}$, define the linear combination of dictionary elements. The underlying assumption in DI is that the number of dictionary elements necessary to accurately represent a measurement is small, i.e., any measurement can be represented by only a few elements in the dictionary. For this reason, DI is also often referred to as sparse coding in the signal processing literature.[1] This allows for regularization in the form of adding a sparsity-promoting penalty function to the optimization problem. For example, one popular choice of a sparsity-promoting penalty function is the $\ell_1$ norm of the estimated coefficients.

Often, we are not only interested in indexing a single measurement, but rather, a set of measurements. For example, if measurements are captured from a network of sensors, one may expect the captured measurements at two closely placed sensors to be related. We consider a framework where there exists an underlying relationship between the signals of interest within a set of DI instances. While it is reasonable to process each instance individually and then aggregate the results, we develop a method which indexes the entire set of DI instances in a joint manner which also leverages the assumed intra-measurement relationship. To this end, we pose the set of indexing problems on a graph structure and look towards graph regularization techniques for creating tractable solutions to the joint indexing problem.

Two main contributions are outlined. First, we demonstrate how to implement graph regularization for the joint DI problem. When the signal defined on the graph has a piece-wise constant structure, we demonstrate how to effectively exploit it with graph regularization during the signal estimation algorithm. In particular, we demonstrate the improvements gained

[1] We adopt the name, "dictionary indexing," to be consistent with the EBSD pattern indexing nomenclature.

from graph regularization in comparison to not leveraging graph structures. We demonstrate the effectiveness of the algorithm for multiple types of graph structures. Secondly, we apply the graph regularization framework to the problem of indexing a set of electron back-scattered diffraction patterns in scanning electron microscopy. We demonstrate that joint indexing with graph regularization can accurately index highly corrupted, low-resolution diffraction patterns. Additionally, we demonstrate that the proposed algorithm can index individual pattern orientations from a mixture of patterns, which may be useful in accurately indexing patterns that are measured at the border of two grains.

### 5.1.1 Related Work

Many algorithms have been proposed to solve the dictionary indexing problem. In computing the sparse coefficients in the linear combination as in (5.1), one seeks to find an accurate set of coefficients that remains sufficiently sparse. To name a few strategies, one can consider greedy strategies of approximation such as the matching pursuit algorithm [71], or the orthogonal matching pursuit algorithm [98]; constrained optimization strategies such as gradient projection sparse reconstruction [40]; or proximal algorithms such as iterative soft- or shrinkage-thresholding algorithms [7]. The latter of these strategies, iterative shrinkage-threshold algorithms, dubbed ISTA, has sparked a line of research interested in optimizing the computation speed of solving DI while maintaining a desired level of accuracy. Notably, ISTA has recently been used in tandem with deep learning to create a class of algorithms known as Learned ISTA, or LISTA [42, 43, 47].

Other works have studied the DI problem formulations within the context of exploiting intra-signal structures. For example, works have analyzed the sparse, multiple measurement vector model for multichannel signal processing [31, 37], as well as spectrum estimation and denoising [63]. One extension of the famous LASSO algorithm, group LASSO, groups the elements in the estimated signal to regularize each group individually [72]. Additionally, for graph structures specifically, a line of work known as Graph Trend Filtering (GTF) [106] has been fruitful in applying graph regularization in the context of vector-valued graph signal denoising [101] as well as recovering a graph signal from a subsampled set of the nodes [102]. The notion of graph structures in joint dictionary indexing was discussed in [83].

For Electron Back-Scattered Diffraction (EBSD) pattern indexing, a pure dictionary indexing based method has been proposed and is shown to be quite robust to system noise [24]. In the context of this existing dictionary indexing method, the observed measurement is assumed to be defined by a single dictionary element. In this chapter, we demonstrate how to generalize this notion to reconstructing a sparse, linear combination of dictionary elements. A follow-up using a technique known as Spherical Indexing (SphInx) has been shown to be much more computationally efficient for indexing patterns [61]. More on EBSD is discussed in Section 5.5.

### 5.1.2 Chapter Organization

The rest of this chapter is organized as follows. We provide further background on single-instance dictionary indexing and then define the joint dictionary indexing signal model and relevant graph structures in Section 5.2. Next, we formalize the optimization problem for solving joint dictionary indexing in Section 5.3 and then provide experimental results in Section 5.4. Then, we introduce electron back-scattered diffraction patterns and connect EBSD pattern indexing to the proposed joint dictionary indexing algorithm in Section 5.5. Some preliminary experimental results are provided in Section 5.6. Finally, we conclude and discuss our findings in Section 5.7.

## 5.2 Problem Formulation and Important Definitions

In this section, we introduce the signal models for both single instance and joint dictionary indexing. Along the way, we introduce important concepts from graph theory that will be referenced throughout the chapter.

### 5.2.1 Single Instance Dictionary Indexing

From the set of $d$ signals $\{\boldsymbol{x}_i\}_{i=1}^d$, construct the dictionary matrix $\boldsymbol{X} = [\boldsymbol{x}_1 \cdots \boldsymbol{x}_d] \in \mathbb{R}^{p\times d}$. Suppose that one wishes to estimate the coefficients of the linear combination of dictionary elements $\boldsymbol{\beta}^\star \in \mathbb{R}^d$ which define a given (noisy) measurement $\boldsymbol{y} \in \mathbb{R}^p$, according to,

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta}^\star + \boldsymbol{\varepsilon}. \tag{5.2}$$

Here, we assume that the measurements at each node are effected by additive white Gaussian noise, i.e., $\boldsymbol{\varepsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2\boldsymbol{I}),\ \sigma > 0$. As the dimension of the measurement is often much smaller than the number of elements in the dictionary, i.e., $p \ll d$, our goal is to solve an under-determined linear system of equations. In the context of dictionary indexing, one assumption that we are making is that the observed measurement is a linear combination of only a few dictionary entries. Such an assumption is familiarly known as sparsity, and a long range of work has been put forth to study this problem of estimating a sparse signal from an under-determined linear system [41]. From an optimization perspective, we can introduce regularization to our loss function formulation to penalize estimates which are not sparse.

One may calculate an estimate $\hat{\boldsymbol{\beta}}$ of $\boldsymbol{\beta}$ by solving the optimization problem,

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}\in\mathbb{R}^d} \left\{ \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + g(\boldsymbol{\beta};\boldsymbol{\zeta}) \right\}. \tag{5.3}$$

The function $g : \mathbb{R}^d \mapsto \mathbb{R}$ denotes our choice of regularization, and $\boldsymbol{\zeta}$ is the set parameter choices necessary for calculating certain penalty functions. Depending on the choice of $g$, a different set of parameters may be necessary for computation. Penalty functions take the form, $g(\boldsymbol{\beta};\boldsymbol{\zeta}) := \sum_{i=1}^d k(\beta_i;\boldsymbol{\zeta})$, where an element-wise operation is taken to promote an element in the vector to be zero. Two sparsity-promoting penalty functions are considered in this work: the $\ell_1$-norm and the Smoothly Clipped Absolute Deviation (SCAD) function [38].

- For the $\ell_1$-norm, only one parameter is necessary; assign $\boldsymbol{\zeta} = \lambda$, where $\lambda > 0$ is a non-negative, real scalar. Then, for any element $\beta \in \boldsymbol{\beta}$, define $k(\beta;\lambda) = \lambda|\beta|$, such that, $g_{\ell_1}(\boldsymbol{\beta};\lambda) = \lambda\sum_{i=1}^d |\beta_i| := \lambda\|\boldsymbol{\beta}\|_1$. With the $\ell_1$-norm as a penalty function, we arrive at the famous LASSO estimate [94].

- For SCAD, two parameters are necessary; assign $\boldsymbol{\zeta} = \{\lambda, \xi\}$, where both parameters are non-negative, real scalars. For any element $\beta \in \boldsymbol{\beta}$, the $k_{\text{SCAD}}(\beta; \lambda, \xi)$ penalty function is defined as,

$$k_{\text{SCAD}}(\beta; \lambda, \xi) = \lambda \int_0^{|x|} \min\left\{1, \frac{\max\{0, (\xi - u/\lambda)\}}{\xi - 1}\right\} du,\ \xi \geq 2, \tag{5.4}$$

and $g_{\text{SCAD}}(\boldsymbol{\beta}; \lambda, \xi) := \sum_{i=1}^{d} k_{\text{SCAD}}(\beta_i; \lambda, \xi)$. This non-convex, sparsity promoting penalty functions has been shown to avoid some implicit bias inherent to the $\ell_1$-norm [101].

### 5.2.2 Joint Dictionary Indexing

We aim to solve a set of $n$ related dictionary indexing problems, $\{\boldsymbol{y}_i, \boldsymbol{\beta}_i^\star\}_{i=1}^n$, as introduced in (5.2). We assume that the same dictionary $\boldsymbol{X}$ is used for each $(\boldsymbol{y}_i, \boldsymbol{\beta}_i^\star)$ pair. Each measurement is affected, independently, by noise $\boldsymbol{\varepsilon}_i \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I})$. Construct matrices $\boldsymbol{Y} = [\boldsymbol{y}_1 \cdots \boldsymbol{y}_n]$, $\boldsymbol{B}^\star = [\boldsymbol{\beta}_1^\star \cdots \boldsymbol{\beta}_n^\star]$, $\boldsymbol{E} = [\boldsymbol{\varepsilon}_1 \cdots \boldsymbol{\varepsilon}_n]$, i.e., each column $\boldsymbol{Y}_{:,i}, \boldsymbol{B}_{:,i}^\star$ of $\boldsymbol{Y}, \boldsymbol{B}^\star$ correspond, respectively, to their own dictionary indexing problem instance $\{\boldsymbol{y}_i, \boldsymbol{\beta}_i^\star\}$. We define the observation model as,

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{B}^\star + \boldsymbol{E}. \tag{5.5}$$

Indeed, each dictionary indexing instance can be solved individually using the estimate provided in (5.3). However, we are interested in the scenario where there exists some relationship between the different signals $\{\boldsymbol{\beta}_i^\star\}_{i=1}^n$. We formalize this relationship using graph structures. Let $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ be a graph such that each dictionary indexing instance lives on one of the $|\mathcal{V}| = n$ nodes. The set of edges, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, represents the set of $|\mathcal{E}| = m$ underlying relationships between any two ground truth signals. For simplicity, we will assume that the graph $\mathcal{G}$ is an undirected graph.

Let $\boldsymbol{\Delta} \in \mathbb{R}^{m \times n}$ be the oriented incidence matrix of $\mathcal{G}$ (see [101]), where each row in the oriented incidence matrix corresponds to an edge in the graph. We will assume that when the ground truth signal is defined on the graph, there exist sets of adjacent nodes which all have identical signals. Such a signal is said to have piece-wise constant structure. Note that we are not assuming that we know which nodes are identical—only that there exists such a relationship where the ground truth signal on adjacent nodes have a higher likelihood of being identical. With this structure, consider the quantity $\boldsymbol{\Delta}\boldsymbol{B}^{\star\top}$, which is the set of piece-wise differences between the signals at each node in the graph. Similar to our intuition about sparsity existing in the underlying signal, one can see that there exists underling sparsity in the $\boldsymbol{\Delta}\boldsymbol{B}^{\star\top}$ object: if there are large patches of piece-wise constant signal, then the pairwise differences between many pairs of nodes will be $\mathbf{0}$. Formally, if $\boldsymbol{B}^{\star}$ defines a piece-wise constant signal on the graph $\mathcal{G}$ with oriented incidence matrix $\boldsymbol{\Delta}$, then for each row of $\boldsymbol{B}^{\star}$, $\boldsymbol{B}^{\star}_{i,:}$, $i = 1, \cdots, n$, the quantity $\sum_{i=1}^{n} \|\boldsymbol{\Delta}\boldsymbol{B}^{\star\top}_{:,i}\|_2$ is small, as the number of non-zero elements pairwise differences will be small.

What remains is understanding how to properly apply regularization on both the signal structure and the graph structure. In either case, $k(\cdot)$ is the desired sparsity-promoting penalty function introduced in Section 5.2.1. For simplicity, we omit the additional parameters necessary (e.g. $\lambda, \xi$). At each node, the reconstructed signal should have a sparse structure, based on our domain knowledge of the dictionary indexing problem. For any matrix $\boldsymbol{V} \in \mathbb{R}^{n_1 \times n_2}$ the following matrix-valued penalty function, $h_1 : \mathbb{R}^{n_1 \times n_2} \mapsto \mathbb{R}$, is defined as,

$$h_1(\boldsymbol{V}) = \sum_{i=1}^{n_1} k(\|\boldsymbol{V}_{:,i}\|_1; \boldsymbol{\zeta}). \tag{5.6}$$

The signal regularizer promotes sparsity on each *column* of the matrix. For signal regularization, the input to the function $h_1$ is the current estimate, $\boldsymbol{B}$. This way, we can exploit our prior knowledge that the ground truth signal living at each node is sparse.

Similarly, we wish to couple the signals reconstructed on different nodes to have the same sparsity pattern if they are learned to be in the same region within the graph. For any matrix $\boldsymbol{V} \in \mathbb{R}^{n_1 \times n_2}$, define the function $h_2 : \mathbb{R}^{n_1 \times n_2} \mapsto \mathbb{R}$ as,

$$h_2(\boldsymbol{V}) = \sum_{j=1}^{n_2} k(\|\boldsymbol{V}_{j,:}\|_2; \boldsymbol{\zeta}). \tag{5.7}$$

For graph regularization, the input to $h_2$ is the quantity $\boldsymbol{\Delta}\boldsymbol{B}^{\top}$, for the current estimate, $\boldsymbol{B}$. The sparsity pattern in each row of $\boldsymbol{B}$ is enforced to be similar. In other words, the graph regularizer promotes sparsity on the differences of the rows of the matrix $\boldsymbol{B}$ according to the graph incidence matrix $\boldsymbol{\Delta}$. This regularization function is closely related to group selection models [51].

## 5.3 Joint Dictionary Indexing with Graph Regularization

We look to generalize the estimate presented in (5.3) to estimate matrix objects which also satisfy our underlying assumptions regarding graph structures. Consider solving the follow optimization problem,

$$\hat{\boldsymbol{B}} = \operatorname{argmin}_{\boldsymbol{B}\in\mathbb{R}^{d\times n}} \left\{ \frac{1}{2}\|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{B}\|_F^2 + h_1(\boldsymbol{B}) + h_2(\boldsymbol{Z}) \right\} \tag{5.8}$$
$$\text{s.t. } \boldsymbol{Z} = \boldsymbol{\Delta}\boldsymbol{B}^\top.$$

Here, $h_1, h_2$ are realizations of two forms of regularization following the definition presented in (5.6), and (5.7), respectively. Note that each penalty function choice will come with some set of real-valued, non-zero, scalar, parameter choices, which is omitted for simplicity. We introduce the variable $\boldsymbol{Z} = \boldsymbol{\Delta}\boldsymbol{B}^\top$ in order to derive an ADMM-based algorithm for obtaining the estimate as defined in (5.8). The augmented Lagrangian function is formulated as,

$$\begin{aligned}\mathcal{L}\left(\boldsymbol{B}, \boldsymbol{Z}, \boldsymbol{U}\right) = &\frac{1}{2}\|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{B}\|_F^2 + h_1(\boldsymbol{B}) + h_2(\boldsymbol{Z}) \\ &+ \frac{\tau}{2}\|\boldsymbol{\Delta}\boldsymbol{B}^\top - \boldsymbol{Z} + \boldsymbol{U}\|_F^2 - \frac{\tau}{2}\|\boldsymbol{U}\|_F^2.\end{aligned} \tag{5.9}$$

The dual variable $\boldsymbol{U}$ is introduced to quantify the difference between $\boldsymbol{\Delta}\boldsymbol{B}^\top$ and $\boldsymbol{Z}$. Additionally, $\tau$ is a parameter which regularizes the Frobenius norm of the $\boldsymbol{U}$ variable. Algorithm 5.1 summarizes the ADMM updates, and its derivation is included in Appendix B.

Recall that for any vector-valued function $f$, the proximal operator is defined as,

$$\operatorname{prox}_{\alpha f}(\boldsymbol{v}) = \arg\min_{\boldsymbol{x}} \left\{ \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{v}\|_2^2 + \alpha f(\boldsymbol{x}) \right\}. \tag{5.10}$$

The $\ell_1$-norm and SCAD penalty function both admit closed-form proximal operators [51]. In order to update the estimate of the ground truth signal during each pass, we need to solve the famous Sylvester's Equation. For any set of compatible matrices $\{\boldsymbol{A}^\circ, \boldsymbol{B}^\circ, \boldsymbol{Q}^\circ\}$, solving Sylvester's equation amounts to finding a matrix $\boldsymbol{R}$ such that,

$$\boldsymbol{A}^\circ\boldsymbol{R} + \boldsymbol{R}\boldsymbol{B}^\circ = \boldsymbol{Q}^\circ. \tag{5.11}$$

Many computationally efficient implementations for solving Sylvester's equation exist, and a unique solution is guaranteed to exist if and only if $\boldsymbol{A}^\circ$ and $-\boldsymbol{B}^\circ$ do not share any eigenvalue [6]. For our experiments written in Python, we utilize the Sylvester's equation solver provided by Scipy's linear algebra toolbox [54].

**Algorithm 5.1** ADMM for Graph-Regularized Joint Dictionary Indexing

```
Input: Data: measurements $\boldsymbol{Y}$, dictionary matrix $\boldsymbol{X}$, graph incidence matrix $\boldsymbol{\Delta}$.
       Parameters: $\tau, T$, and choice of $h_1, h_2$, (see discussion in 5.2.1).
Initialize:
    $\boldsymbol{B} = \boldsymbol{X}^\top \boldsymbol{Y}, \boldsymbol{Z} = \boldsymbol{\Delta}\boldsymbol{B}^\top, \boldsymbol{U} = \boldsymbol{0},$
    $\overline{\boldsymbol{D}} = \boldsymbol{\Delta}^\top\boldsymbol{\Delta}, \overline{\boldsymbol{V}} = \boldsymbol{X}^\top\boldsymbol{X}, \overline{\boldsymbol{W}} = \boldsymbol{X}^\top\boldsymbol{Y}.$
for $t = 1, \cdots, T$ do
    $\boldsymbol{Q} \leftarrow \overline{\boldsymbol{W}} + \tau\left(\boldsymbol{Z} - \boldsymbol{U}\right)^\top \boldsymbol{\Delta}.$
    $\boldsymbol{B} \leftarrow \text{SolveSylvesters}\left(\boldsymbol{A}^\circ \leftarrow \overline{\boldsymbol{V}}, \boldsymbol{B}^\circ \leftarrow \tau\overline{\boldsymbol{D}}, \boldsymbol{Q}^\circ \leftarrow \boldsymbol{Q}\right)$ {See (5.11)}.
    for $i = 1, \cdots, n$ do
        $\boldsymbol{B}_{:,i} \leftarrow \text{prox}_{h_1}(\boldsymbol{B}_{:.i})$
    end for
    for $j = 1, \cdots, m$ do
        $\boldsymbol{Z}_{j,:} \leftarrow \text{prox}_{h_2/\tau}(\boldsymbol{\Delta}_{j,:}\boldsymbol{B}^\top + \boldsymbol{U}_{j,:})$
    end for
    $\boldsymbol{U} \leftarrow \boldsymbol{U} + \left(\boldsymbol{\Delta}\boldsymbol{B}^\top - \boldsymbol{Z}\right).$
end for
Output: $\hat{\boldsymbol{B}} = \boldsymbol{B}.$
```

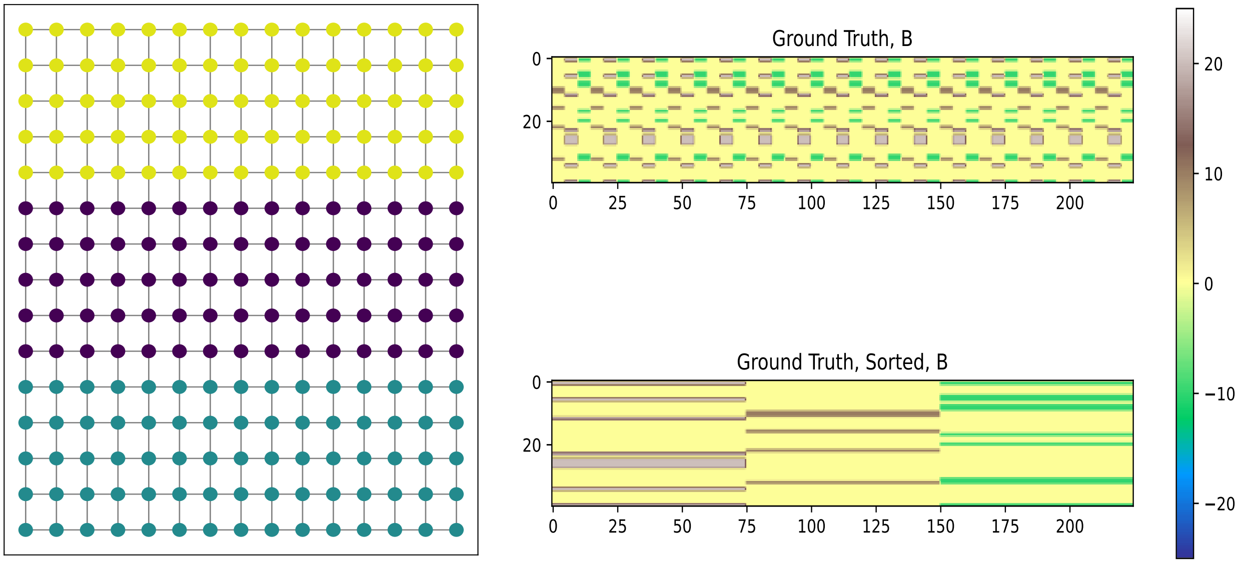


Figure 5.1: Left: The ground truth 2D-grid graph. The ground truth signal is identical within each of the regions with the same coloration. Right: The ground truth signal. At the top, we visualize it such that each column corresponds to one of the nodes in the 2D-grid graph. To see the correspondence, start at the top left, then go top to bottom, left to right. On the bottom, the columns have been grouped according to the class of the ground truth signal. The sorted ground truth is used for visualization only.

## 5.4 Joint Dictionary Indexing Experimental Results

In this section, we provide the empirical performance of ADMM for solving (5.8) as outlined in Algorithm 5.1. The performance is measured using the Normalized Mean Squared Error (NMSE),

$$\text{NMSE} = \frac{\|\hat{\boldsymbol{B}} - \boldsymbol{B}\|_F^2}{\|\boldsymbol{B}\|_F^2}, \tag{5.12}$$

where $\hat{\boldsymbol{B}}$ is the output estimate of the algorithm. For each experiment, we tuned the parameters using the Hyperopt toolbox [8]. The Python packages NetworkX [48] and PyGSP [34] were used to construct and plot graphs. Both $\ell_1$-norm and SCAD based penalty functions are considered in the 2D grid graph experiments. Following the grid graph experiments, only the SCAD penalty function is used with the random geometric graph experiments.

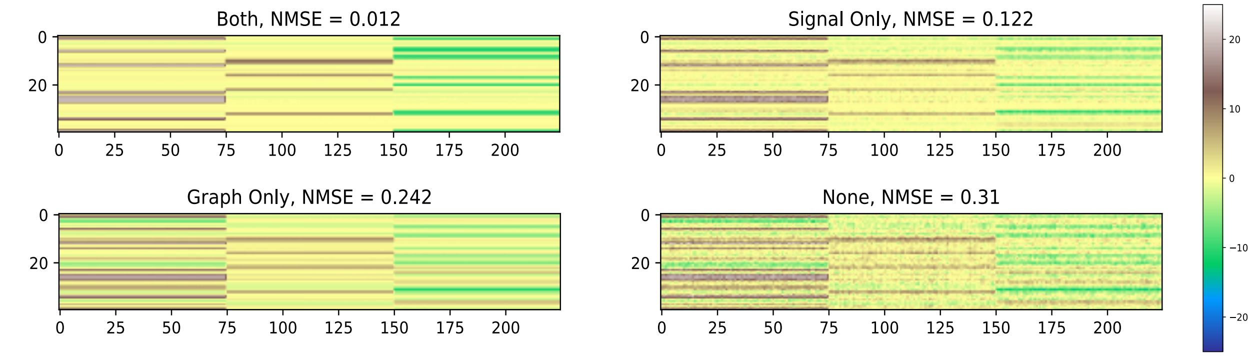


Figure 5.2: The output of the algorithm with four choices of regularization combinations. For these experiments, the SCAD penalty function has been used for both signal and graph regularization. Each of the recovered signals have their columns sorted according to the same order as presented in the bottom-right plot of Figure 5.1 for visualization only.

### 5.4.1 2D Grid Graph

For this set of experiments, a square, 2D-grid graph is generated with $n = 15 \times 15 = 225$ nodes. The ground truth graph is plotted in the left pane of Figure 5.1. Labels are assigned to each node according to this graph, i.e., there are three different signals and each $\boldsymbol{\beta}_i, \forall i \in [n]$ is determined according to the piece-wise constant structure. Each measurement $\boldsymbol{y}_i$ is obtained according to (5.2), where $d = 40$ and $p = 30$. Each element of the dictionary matrix, $\boldsymbol{X}$, is drawn i.i.d. at random according to, $X_{i,j} \sim \mathcal{N}(0, 1/d)$. The ground truth signal is plotted in the right pane of Figure 5.1. Each of the three ground truth signal representations have entries generated according to $\beta_k \cdot \eta_i, \forall k \in \{1, 2, 3\}, \forall i \in [n]$, where, $\beta_k \in \{20, 10, -10\}$ and $\eta_i \sim$ Bernoulli$(0.2)$, i.e., the probability of $\eta_i$ being 1 is 0.2.

In order to demonstrate the effectiveness of graph regularization for solving the joint dictionary indexing problem, we compare four different algorithm configurations. First, we consider the estimate where neither graph regularization nor signal sparsity are enforced, i.e., $h_1, h_2$ are absent, which is equivalent to $\hat{\boldsymbol{B}}_{LS} = \boldsymbol{X}^{\dagger}\boldsymbol{Y}$. Next, we compute the estimate when graph regularization is not enforced, and only signal regularization is used on the ground truth signal itself, i.e., $h_2$ is not considered. Then, we enforce graph regularization, but no signal regularization, i.e. $h_1$ is not

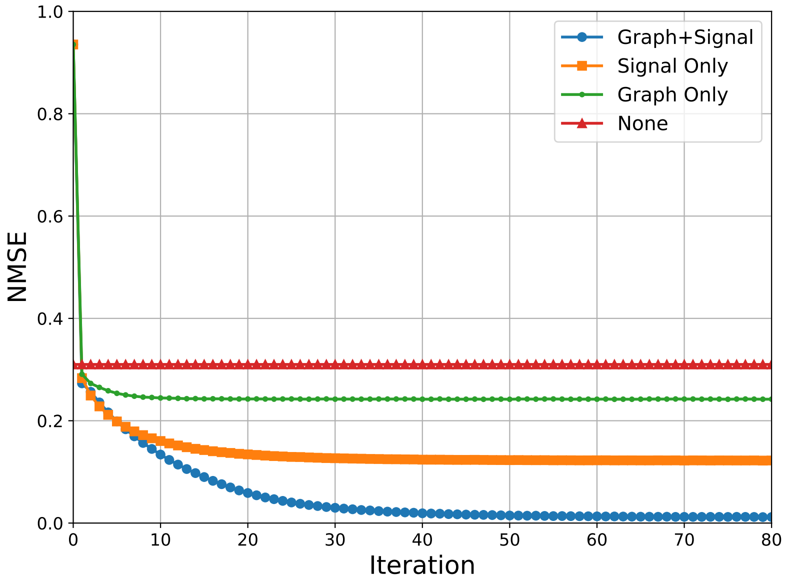


Figure 5.3: The NMSE of the estimate at each iteration for the four scenarios when the SCAD penalty function has been used for both regularizer functions. Here, zero-mean, additive white Gaussian noise with variance $\sigma^2 = 1.0$ has been added to each measurement.

included. Finally, we compute the estimate when both forms of regularization are enforced. In any case, all relevant parameters are tuned accordingly.

We first examine the accuracy of the final estimate provided from each algorithm scenario in Figure 5.2. For these experiments, the additive Gaussian noise is drawn according to a variance of $\sigma^2 = 1.0$. For a qualitative evaluation of each estimate, we refer to the colorbar—due to the choice of a relatively divergent colorbar, small errors are qualitatively emphasized. In Figure 5.2, the upper-left, upper-right, lower-left, and lower-right correspond to the reconstructed signal when both forms of regularization, only signal regularization, only graph regularization, and no regularization are used, respectively. These results corroborate the intuition that both forms of regularization would be the most effective for recovering a piece-wise constant, sparse signal. Next, we look towards the quantitative performance of the proposed algorithm. We plot the NMSE of the estimate at each iterate for each of the four configurations in Figure 5.3. As an ADMM algorithm is used with proper parameter choices, the algorithm is guaranteed to converge

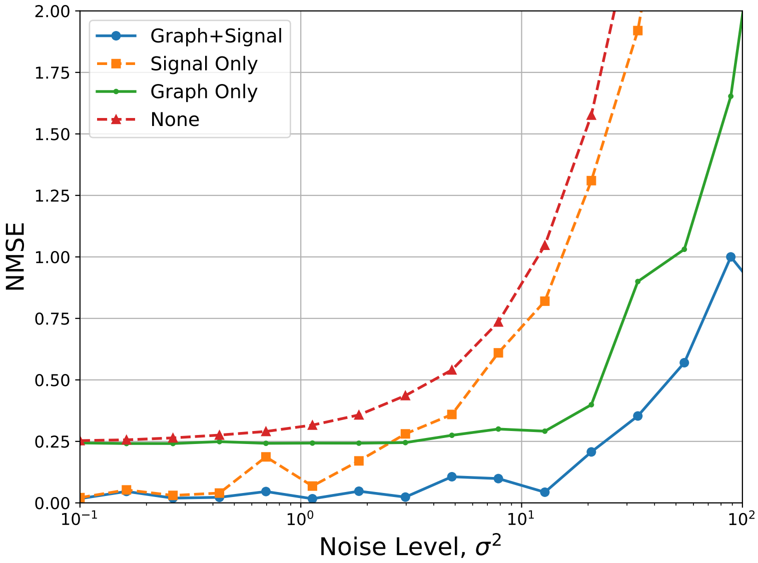


Figure 5.4: The NMSE of the output for the four scenarios with respect to different noise variance levels of additive Gaussian noise.

to some solution; see [70]. However, when both forms of regularization are used, the algorithm converges to a more accurate reconstructed signal. Finally, we analyze how robust the algorithm is to differing levels of noise in Figure 5.4. Hyperparameters are tuned for each noise level. Interestingly, both configurations which utilize graph regularization are more robust to noise.

Next, we mimic the previously presented experiments with $\ell_1$ regularizers. We visualize the output of each algorithm configuration in Figure 5.5 along with the NMSE of the output. The $\ell_1$ penalty function is unable to reconstruct as accurate of a signal as the SCAD penalty function. The same performance hierarchy exists with the $\ell_1$ regularizers as with the SCAD regularizers; both forms of regularization performs the best, while only signal regularization outperforms only graph regularization. We can additionally examine the convergence plots for the $\ell_1$ regularization choice in Figure 5.6. Again, similar conclusions can be made as with the SCAD penalty function.

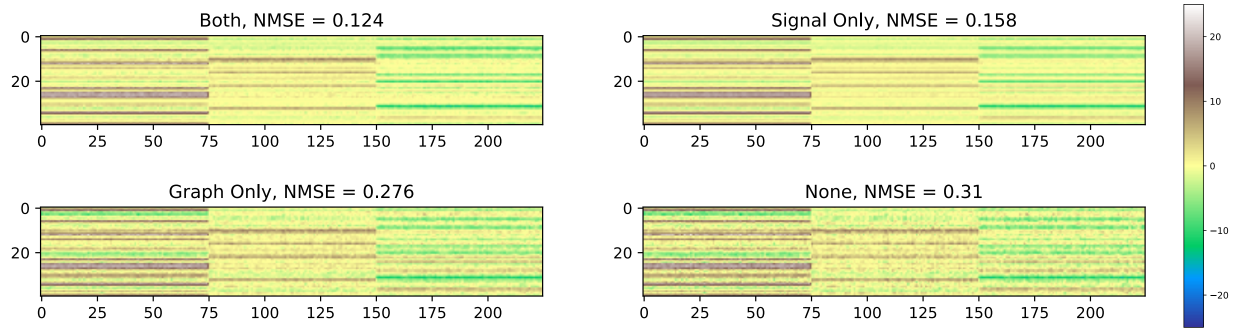


Figure 5.5: The output of the algorithm with four choices of regularization combinations. For these experiments, the $\ell_1$ penalty function has been used for both signal and graph regularization. Each of the recovered signals have their columns sorted according to the same order as presented in the bottom-right plot of Figure 5.1 for visualization only.

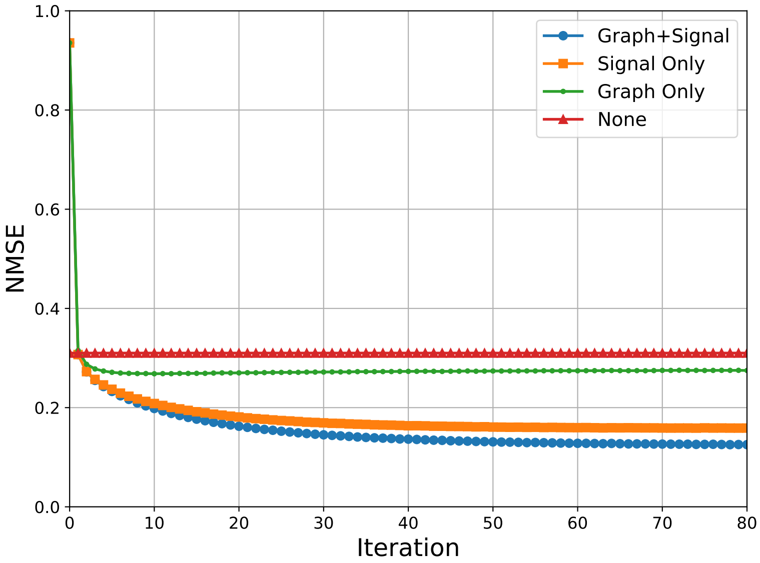


Figure 5.6: The NMSE of the estimate at each iteration for the four scenarios when the $\ell_1$ penalty function is used for both regularizer functions with $\sigma^2 = 1$.

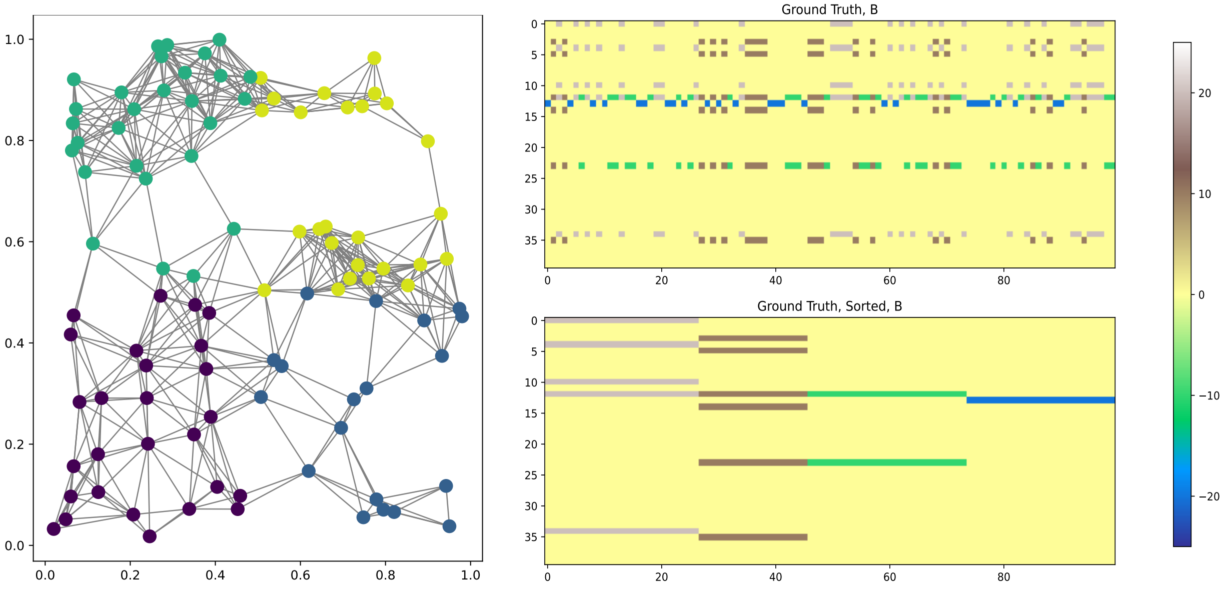


Figure 5.7: Left: The random geometric graph. The unit square has been divided into four quadrants, where the ground truth signal within each quadrant is identical. At the top, we visualize it such that each column corresponds to one of the nodes in the graph. On the bottom, the columns have been grouped according to the class of the ground truth signal. The sorted ground truth is used for visualization only.

### 5.4.2 Random Geometric Graph

For this set of experiments, a random geometric graph, which is often used to model distributed sensor networks, is generated with $n = 100$ nodes. The graph is plotted in the left pane of Figure 5.7. The coordinates of each sensor are distributed randomly in a unit square plane. Edges are drawn between two nodes when the distance between two nodes is less than or equal to $0.2$—in this graph, there are 526 edges. Labels are assigned to each node according to which quadrant the node is located in, i.e., there are four different signals and each $\boldsymbol{\beta}_i, \forall i \in [n]$ is determined according to the piece-wise constant structure. Each measurement $\boldsymbol{y}_i$ is obtained according to (5.2), where $d = 40$ and $p = 30$. Each element of the dictionary matrix, $\boldsymbol{X}$, is drawn i.i.d. at random according to, $X_{i,j} \sim \mathcal{N}(0, 1/d)$. The ground truth signal is shown in the right pane of Figure 5.7. Each of the four ground truth signal representations have entries generated according

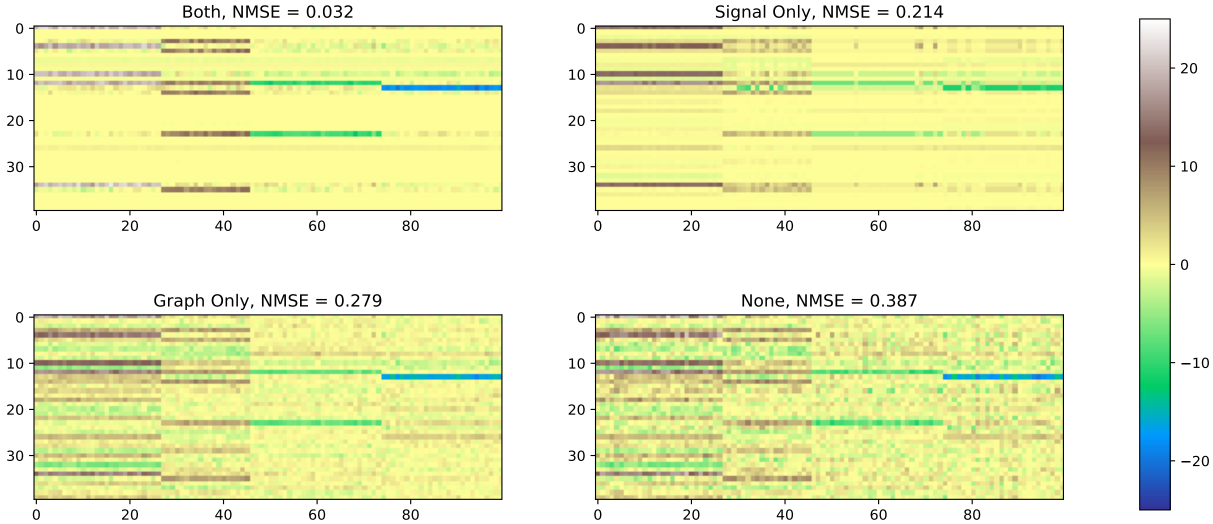


Figure 5.8: The output of the algorithm with four choices of regularization combinations. For these experiments, the SCAD penalty function has been used for both signal and graph regularization. Each of the recovered signals have their columns sorted according to the same order as presented in the bottom-right pane of Figure 5.7 for visualization only.

to $\beta_k \cdot \eta_i, \forall k \in \{1, 2, 3, 4\}, \forall i \in [n]$, where, $\beta_k \in \{20, 10, -10, -20\}$ and $\eta_i \sim \text{Bernoulli}(0.1)$. For each of these experiments, the SCAD penalty function is used with Algorithm 5.1.

A similar narrative is formed for the random geometric graph as with the 2D grid graph. While any form of regularization helps the ADMM-based algorithm to outperform the least squares estimate, exploiting the additional knowledge of the piece-wise constant signal structure improves the accuracy of the reconstructed signal. The reconstructed signals are shown in Figure 5.8. One noticeable difference in the reconstruction quality can be seen in the recovery of the first two signal classes, i.e., the light brown and dark brown. Values corresponding to the proper ground truth signal are recovered when both forms of regularization are used, but, this is not the case when only signal regularization is employed. Convergence for each algorithm configuration is plotted in Figure 5.9. One interesting difference between the convergence plot for the 2D-grid graph and the distributed sensor network graph is that the algorithm which only utilizes signal regularization converges marginally faster that using both forms of regularization.

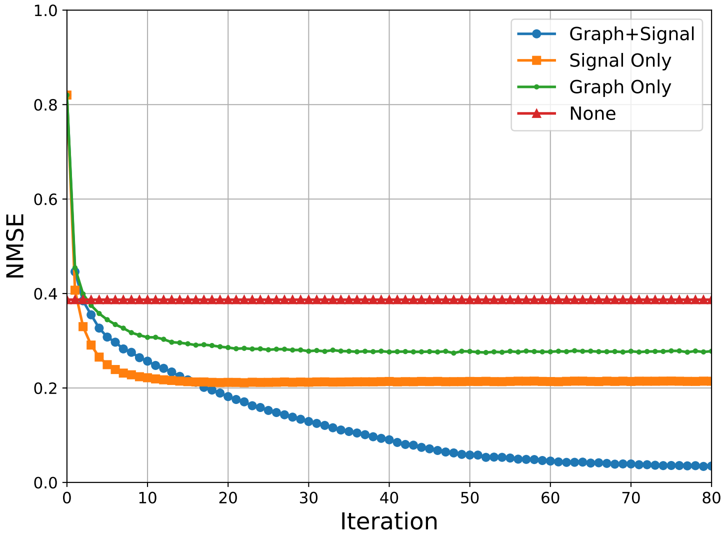


Figure 5.9: The NMSE of the estimate at each iteration for the four scenarios when the SCAD penalty function has been used for both regularizer functions.

## 5.5 Indexing Electron Back-Scattered Diffraction Patterns

In this section, we provide an overview of the Electron Back-Scattered Diffraction (EBSD) technique. The data created and figures visualized derive from the use of the EMSoft public repository [32].

We are interested in the problem of indexing a set of electron back-scattered diffraction (EBSD) patterns of some sample. In this context, indexing refers to estimating the orientation of the crystal structure in 3D space in terms of the Euler angles of the rotation. In a nutshell, a scanning electron microscope (SEM) is used to collect diffraction patterns of the sample in a raster-scan fashion. From knowledge of the microscope geometry, the physical and chemical makeup of the sample, and the time spent with the electron beam focused at a certain point, physics-based forward models have been derived which can accurately synthetically generate the diffraction patterns via Monte Carlo simulations [14]. The goal of this work is to—in post-processing—improve the accuracy of indexing each pattern individually as well as the set of diffraction patterns contained from the entire experiment.

Traditionally, diffraction patterns have been indexed using the Hough transform to extract the

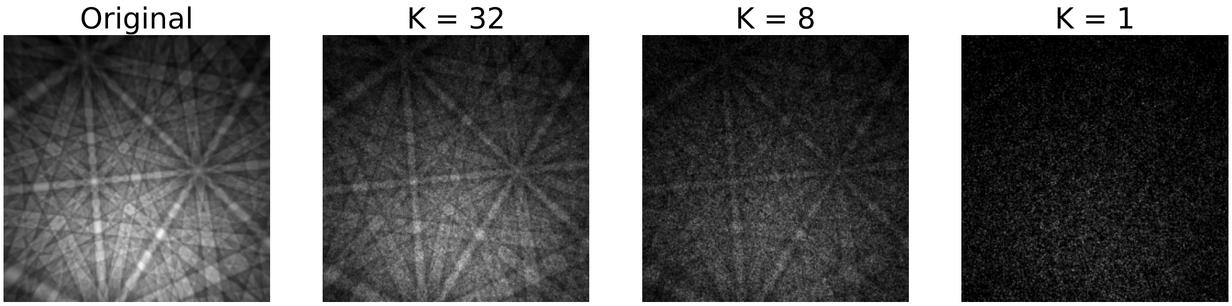


Figure 5.10: The effect of Poisson noise on the diffraction pattern quality.

positions and orientations of the characteristic Kikuchi bands [1, 58, 59]. Other approaches have been proposed, including a technique based on dictionary indexing [24]. While the dictionary indexing method was shown to be exceptionally robust to noisy diffraction patterns, its use is computationally heavy due to the incredibly large size of the dictionary. More recently, a technique known as Spherical Indexing (SphInx) has been proposed [61], which is based upon spherical harmonic transforms. In the following sections, we look to improve upon the dictionary indexing technique by enforcing global structures with graph regularization.

Diffraction patterns are visualizations of the number of electrons measured at the detector plane in the SEM. The quality of the diffraction patterns is dependent on the physical properties of the system. The combination of sources of noise in the observed diffraction pattern can be approximated by Poisson noise. Let $\boldsymbol{y}^\star = \boldsymbol{X}\boldsymbol{\beta}^\star$ be the (vectorized) ground truth diffraction pattern. Assume that each pixel values falls within the range $y_i^\star \in [0, 1]$. The noisy observed diffraction pattern $\boldsymbol{y}$ is then modeled by,

$$z_i \overset{\text{i.i.d.}}{\sim} \text{Poisson}(K y_i^\star), \tag{5.13}$$

$$y_i = \frac{z_i}{K}. \tag{5.14}$$

Here, $K > 0$ is a positive integer which determines the amount of noise present in the system. Example diffraction patterns at various noise levels for the same orientation are visualized in Figure 5.10. Larger values of $K$ correspond with less noise being present in the observed diffraction pattern.

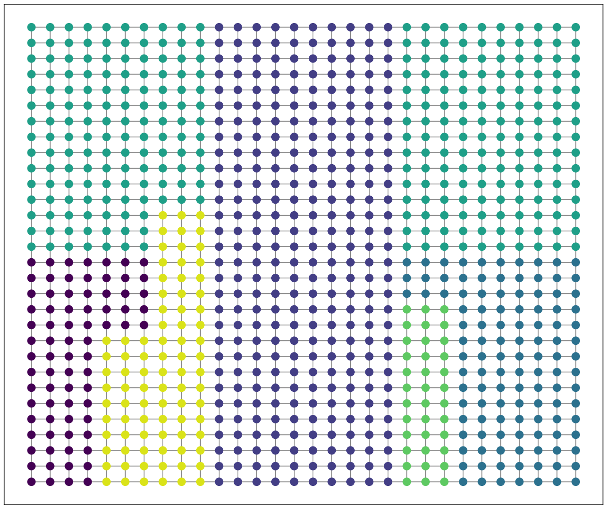
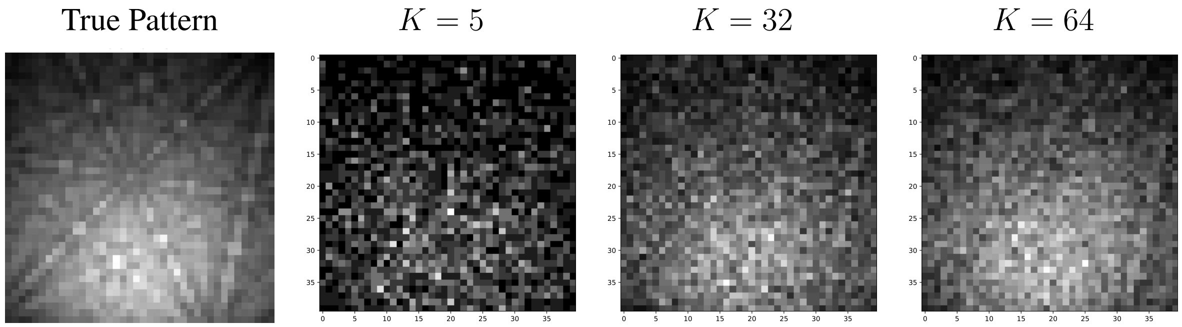


Figure 5.11: The graph structure used for the EBSD experiments, where the nodes have been labeled according to the different ground truth diffraction patterns.

## 5.6 EBSD Experimental Results

First, we must elucidate the process of performing joint dictionary indexing for EBSD diffraction patterns with graph regularization. We obtain $d$ diffraction patterns, $\{\boldsymbol{x}_i\}_{i=1}^{d}$, and construct the dictionary matrix $\boldsymbol{X} = [\boldsymbol{x}_1 \cdots \boldsymbol{x}_d]$. Each diffraction pattern is made up of $p$ pixels. This dictionary can be produced using the EMsoft public repository [32]. Let $\{\boldsymbol{y}_i\}_{i=1}^{n}$ be the set of $n$ diffraction patterns obtained in a raster-scan of the desired sample alloy, where each pattern is made up of $p$ pixels. Then for each diffraction pattern, the vector $\boldsymbol{\beta}_i$ is the (1-sparse) vector which determines the linear combination of dictionary items of $\boldsymbol{X}$ that define the observed pattern. In order to estimate the set of vectors $\{\boldsymbol{\beta}_i\}_{i=1}^{n}$, one can create matrix objects according to (5.5) and run Algorithm 5.1.

In this section, we provide preliminary results for joint pattern dictionary indexing as described in the previous section. The patterns are generated using the EMsoft software package [32]. The simulated dataset considered consists of EBSD scans of a Nickel sample at 20 keV. Individual 8-bit patterns were binned to $320 \times 320$ pixels, and then resized to be $40 \times 40$ pixels. After resizing, the experimental patterns are corrupted by various amounts of noise according to the


True Pattern

$K = 5$

$K = 32$

$K = 64$

Figure 5.12: Example noisy patterns are visualized, where the patterns are resized to lower dimensions. With the smaller patterns being used, the impact of noise is much more prevalent, in comparison to the patterns presented in Figure 5.10

.

noise model described in (5.13). Visualizations of the noisy patterns considered within the experiments are in Figure 5.12. We create a dictionary comprised of $d = 3000$ patterns and the observed, noisy experimental patterns exist within this (relatively small) dictionary. This differs from the traditional notion of dictionary indexing in the EBSD literature, where a more granular dictionary is created of ground truth patterns, and then the full experimental pattern (without resizing) is compared with every dictionary pattern to find the dictionary element which has the highest correlation with the experimental pattern. We leave the process of dictionary indexing a set of unknown experimental patterns against a full dictionary to future work. Results are quantified based on the classification rate, which is defined as the ratio of correctly indexed experimental patterns to the total number of experimental patterns. Algorithm 5.1 is run with both forms of regularization with the SCAD penalty function, where the relevant parameters have been properly tuned using the Hyperopt package.

### 5.6.1 Joint Diffraction Pattern Indexing

For this set of experiments, we vary the amount of corruption added to the experimental patterns and measure the classification rate. The graph configuration used for the experiments is shown in

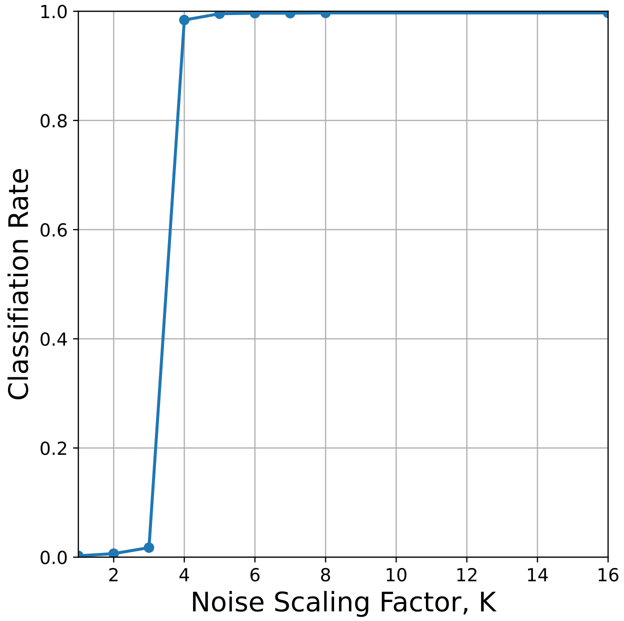


Figure 5.13: The classification rate as the scaling factor of Poisson noise, $K$, is modified. Recall that a smaller $K$ corresponds to more corruption; see Figure 5.12. 100% classification accuracy is obtained as soon as $K = 5$.

Figure 5.11, where each region corresponds to a different ground truth experimental diffraction pattern. From an optimization perspective, this amounts to recovering a 1-sparse signal from a noisy measurement, as the experimental pattern is in the dictionary of known patterns. As is expected, Algorithm 5.1 is able to successfully index patterns. Referring to Figure 5.12, one can conclude that it is possible to index incredibly corrupted experimental patterns with the usage of signal- and graph-regularization. As is seen in Figure 5.13, the algorithm is robust to the presence of Poisson noise.

### 5.6.2 Joint Pattern Demixing Model

In order to make the pattern indexing problem even more challenging, we propose a new experimental setting where experimental patterns have been added together, and now the goal is to recover the orientations of each of the summed experimental patterns. Figure 5.14 visualizes what the sum of 4 patterns looks like to the human eye. Each region in the graph in Figure 5.11

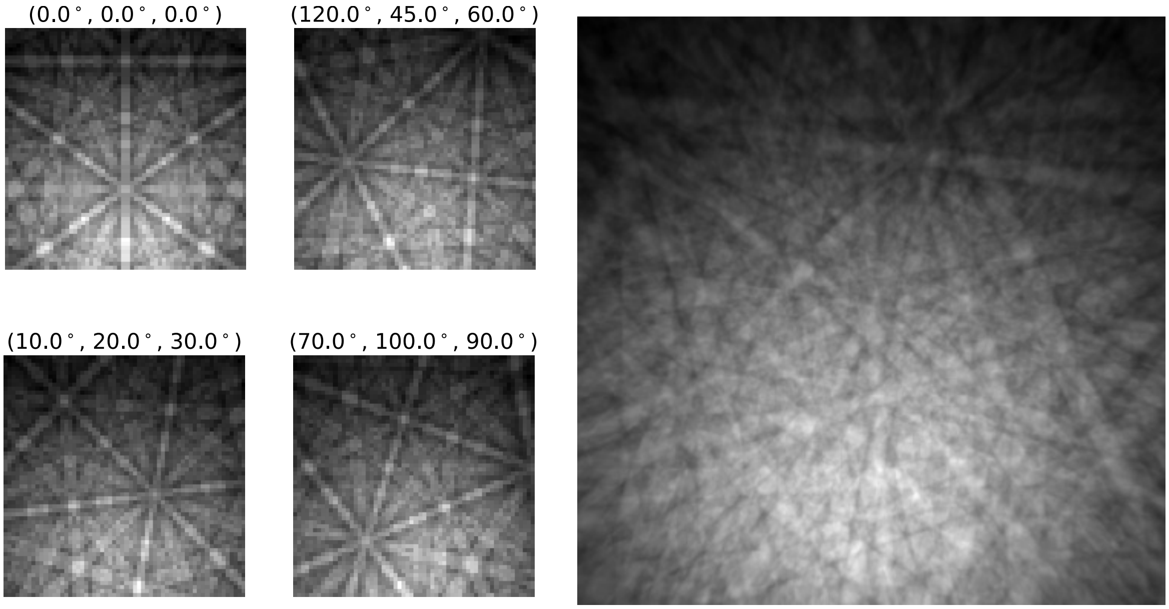


Figure 5.14: An example of four different diffraction patterns being added together to form a mixture of patterns. Algorithm 5.1 is then used to recover each of the ground truth orientations simultaneously.

corresponds to a different mixture of experimental patterns. For these experiments, Poisson noise has not been added to the observed mixture of experimental patterns. Algorithm 5.1 is again run with the SCAD penalty function used for both forms of regularization. Figure 5.15 catalogs the effectiveness of the algorithm to classifying a mixture of diffraction patterns. One can expect near-perfect classification rates for indexing a mixture of up to 40 patterns. Even at a mixture of 100 patterns, the algorithm is able to accurately classify over 94% of the experimental patterns included within the mixed pattern signal observed at each node of the graph.

## 5.7 Conclusion

We empirically demonstrate the utility of graph regularization in solving joint dictionary indexing problems. The proposed ADMM-based algorithm for signal- and graph-regularization is applied to the problem of indexing the orientations of electron back-scattered diffraction patterns from a

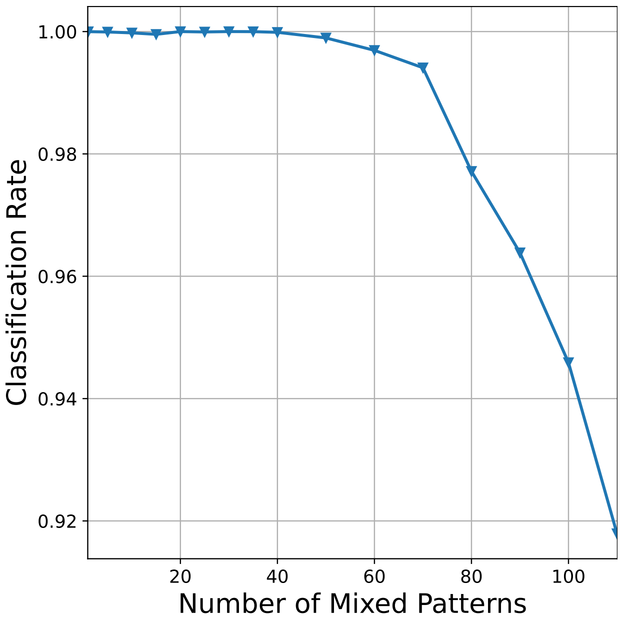


Figure 5.15: The classification rate as the number of diffraction patterns mixed together increases. The classification checks that all patterns are correctly indexed. A decline in performance is observed when more than 40 patterns are mixed together.

scanning electron microscope. In both cases, graph regularization is demonstrated to improve the global conformity with the structure assumed to exist *a priori* in the signal model. In both cases, we would like to investigate the performance of graph regularization in more scenarios: different noise levels, types of noise, forward models, etc. Furthermore, we are interested to see graph regularization can be applied to the spherical indexing process. Additionally, it would be interesting to prove the theoretical performance in both cases, similarly as in [101].

# Appendix A

# Derivations for Chapter 2

In the appendix, we provide the derivations of the optimal preprocessing functions in AWGN and Poisson noise using the formula in [76]. For the derivations, suppose that $\boldsymbol{x} \in \mathbb{C}^n$ is uniformly distributed over the $n$-dimensional complex sphere with radius $\sqrt{n}$, i.e., $\boldsymbol{x} \sim \text{Unif}(\sqrt{n}\mathbb{S}_{\mathbb{C}}^{n-1})$, and $\{\boldsymbol{a}_i\}_{i=1}^m \overset{\text{i.i.d.}}{\sim} \mathcal{N}(\boldsymbol{0}, \frac{1}{2n}\boldsymbol{I}_n) + j\mathcal{N}(\boldsymbol{0}, \frac{1}{2n}\boldsymbol{I}_n)$. Let $s_i = \langle \boldsymbol{x}, \boldsymbol{a}_i \rangle$, the optimal preprocessing function can be derived as

$$\mathcal{T}(y) = 1 - \frac{\mathbb{E}_s\{p(y \mid |s|)\}}{\mathbb{E}_s\{|s|^2 p(y \mid |s|)\}}. \tag{A.1}$$

Furthermore, the sampling threshold can be derived as

$$\alpha_u = \left( \int_{\mathbb{R}} \frac{\mathbb{E}_s\left\{p(y \mid |s|)(1 - |s|^2)\right\}}{\mathbb{E}_s\left\{p(y \mid |s|)\right\}} \right)^{-1}. \tag{A.2}$$

## A.1 Derivations for the AWGN case

In the presence of AWGN, the measurement $y_i$ follows the distribution $y_i \sim p(y \mid |s_i|)$, where

$$p(y \mid |s_i|) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left( -\frac{(y - |s_i|^2)^2}{2\sigma^2} \right). \tag{A.3}$$

Furthermore, we can rewrite $s_i = \Re(s_i) + j\Im(s_i)$, where $(\Re(s_i), \Im(s_i)) \sim \mathcal{N}(\boldsymbol{0}, \frac{1}{2}\boldsymbol{I}_2)$. Let $R = |s_i| = \sqrt{\Re(s_i)^2 + \Im(s_i)^2}$. Then $R$ follows a Rayleigh distribution with scale parameter $\frac{1}{\sqrt{2}}$.

### A.1.1 Optimal Preprocessing Function for AWGN

By the calculations in A.1.1, we have,

$$\mathbb{E}_s\{p(y \mid |s|)\} = \exp\left(-\frac{(2y-\sigma^2)}{2}\right)\mathcal{Q}(-(y-\sigma^2)/\sigma). \tag{A.4}$$

By the calculations in A.1.1, we have,

$$\begin{aligned}\mathbb{E}_s\{|s|^2 p(y \mid |s|)\} = (y-\sigma^2)\exp\left(-\frac{(2y-\sigma^2)}{2}\right)\mathcal{Q}(-(y-\sigma^2)/\sigma)+\\ \frac{\sigma}{\sqrt{2\pi}}\exp\left(-\frac{y^2}{2\sigma^2}\right).\end{aligned} \tag{A.5}$$

Therefore, plugging (A.4) and (A.5) into (A.1), we obtain the optimal preprocessing function as,

$$\begin{aligned}\mathcal{T}(y) &= 1 - \frac{\mathbb{E}_s\{p(y \mid |s|)\}}{\mathbb{E}_s\{|s|^2 p(y \mid |s|)\}} && \text{(A.6)}\\ &= 1 - \frac{\exp\left(-\frac{(2y-\sigma^2)}{2}\right)\mathcal{Q}(-(y-\sigma^2)/\sigma)}{(y-\sigma^2)\exp\left(-\frac{(2y-\sigma^2)}{2}\right)\mathcal{Q}(-(y-\sigma^2)/\sigma)+\frac{\sigma}{\sqrt{2\pi}}\exp\left(-\frac{y^2}{2\sigma^2}\right)} && \text{(A.7)}\\ &= 1 - \left(y-\sigma^2+\sqrt{\frac{\sigma^2}{2\pi}}\frac{\exp\left(-\frac{(y-\sigma^2)^2}{2\sigma^2}\right)}{\mathcal{Q}\left(-\frac{(y-\sigma^2)}{\sigma}\right)}\right)^{-1}. && \text{(A.8)}\end{aligned}$$

**Calculations for $\mathbb{E}_s\{p(y \mid |s|)\}$ in AWGN**

We have

$$\begin{aligned}\mathbb{E}_s\{p(y \mid |s|)\} &= \mathbb{E}_R\{p(y \mid R)\}\\ &= \int_0^\infty \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(y-z)^2}{2\sigma^2}\right)\exp(-z)dz\\ &= \int_0^\infty \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{y^2-2yz+z^2}{2\sigma^2}\right)\exp(-z)dz\\ &= \int_0^\infty \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{y^2-2yz+z^2+2\sigma^2 z}{2\sigma^2}\right)dz\\ &= \int_0^\infty \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{y^2-(2y-2\sigma^2)z+z^2}{2\sigma^2}\right)dz\\ &= \int_0^\infty \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(z-(y-\sigma^2))^2+\sigma^2(2y-\sigma^2)}{2\sigma^2}\right)dz\end{aligned}$$

$$
\begin{aligned}
&= \exp\left(-\frac{(2y-\sigma^2)}{2}\right)\int_0^{\infty}\frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(z-(y-\sigma^2))^2}{2\sigma^2}\right)dz \\
&= \exp\left(-\frac{(2y-\sigma^2)}{2}\right)\int_{-(y-\sigma^2)}^{\infty}\frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{z^2}{2\sigma^2}\right)dz \\
&= \exp\left(-\frac{(2y-\sigma^2)}{2}\right)\mathcal{Q}(-(y-\sigma^2)/\sigma).
\end{aligned}
$$

**Calculations for $\mathbb{E}_s\{|s|^2 p(y \mid |s|)\}$ in AWGN**

Similarly,

$$
\begin{aligned}
\mathbb{E}_s\{|s|^2 p(y \mid |s|)\} &= \int_0^{\infty}\frac{z}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(y-z)^2}{2\sigma^2}\right)\exp(-z)dz \\
&= \exp\left(-\frac{(2y-\sigma^2)}{2}\right)\int_0^{\infty}\frac{z}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(z-(y-\sigma^2))^2}{2\sigma^2}\right)dz \\
&= \exp\left(-\frac{(2y-\sigma^2)}{2}\right)\int_{-(y-\sigma^2)}^{\infty}\frac{z+(y-\sigma^2)}{\sigma\sqrt{2\pi}}\exp\left(-\frac{z^2}{2\sigma^2}\right)dz \\
&= \exp\left(-\frac{(2y-\sigma^2)}{2}\right)\left[\int_{-(y-\sigma^2)}^{\infty}\frac{z}{\sigma\sqrt{2\pi}}\exp\left(-\frac{z^2}{2\sigma^2}\right)dz\right. \\
&+ \left.\int_{-(y-\sigma^2)}^{\infty}\frac{(y-\sigma^2)}{\sigma\sqrt{2\pi}}\exp\left(-\frac{z^2}{2\sigma^2}\right)dz\right] \\
&= \exp\left(-\frac{(2y-\sigma^2)}{2}\right)\int_{-(y-\sigma^2)}^{\infty}\frac{z}{\sigma\sqrt{2\pi}}\exp\left(-\frac{z^2}{2\sigma^2}\right)dz \\
&+ (y-\sigma^2)\exp\left(-\frac{(2y-\sigma^2)}{2}\right)\mathcal{Q}(-(y-\sigma^2)/\sigma).
\end{aligned}
$$

Note that the first term can be further simplified as

$$
\begin{aligned}
\int_{-(y-\sigma^2)}^{\infty}\frac{z}{\sigma\sqrt{2\pi}}\exp\left(-\frac{z^2}{2\sigma^2}\right)dz &= \frac{1}{\sqrt{\pi}}\int_{-(y-\sigma^2)}^{\infty}\frac{z}{\sigma\sqrt{2}}\exp\left(-\left(\frac{z}{\sigma\sqrt{2}}\right)^2\right)dz \\
&= \frac{\sigma\sqrt{2}}{\sqrt{\pi}}\int_{-\frac{(y-\sigma^2)}{\sigma\sqrt{2}}}^{\infty}u\exp\left(-u^2\right)du \\
&= \frac{\sigma\sqrt{2}}{\sqrt{\pi}}\left[-\frac{1}{2}\exp(-u^2)\right]_{-\frac{(y-\sigma^2)}{\sigma\sqrt{2}}}^{\infty} \\
&= \frac{\sigma}{\sqrt{2\pi}}\exp\left(-\frac{(y-\sigma^2)^2}{2\sigma^2}\right),
\end{aligned}
$$

leading to the expression in (A.5).

### A.1.2 Sampling Ratio Threshold for AWGN

Plugging (A.4) and (A.5) into (A.2), we tackle the calculation of the sampling threshold as

$$\begin{aligned}
\alpha_u^{-1} &= \int_{\mathbb{R}} \frac{\left(\mathbb{E}_s\{p(y \mid |s|)(|s|^2-1)\}\right)^2}{\mathbb{E}_s\{p(y \mid |s|)\}} dy \\
&= \int_{\mathbb{R}} \frac{\left[\mathbb{E}_s\{|s|^2 p(y \mid |s|)\} - \mathbb{E}_s\{p(y \mid |s|)\}\right]^2}{\mathbb{E}_s\{p(y \mid |s|)\}} dy \\
&= \int_{\mathbb{R}} \frac{\left[(y-\sigma^2-1)\exp\left(-\frac{(2y-\sigma^2)}{2}\right)\mathcal{Q}(-(y-\sigma^2)/\sigma) + \frac{\sigma}{\sqrt{2\pi}}\exp\left(-\frac{y^2}{2\sigma^2}\right)\right]^2}{\exp\left(-\frac{(2y-\sigma^2)}{2}\right)\mathcal{Q}(-(y-\sigma^2)/\sigma)} dy \\
&= \underbrace{\int_{\mathbb{R}} (y-\sigma^2-1)^2 \exp\left(-\frac{(2y-\sigma^2)}{2}\right)\mathcal{Q}(-(y-\sigma^2)/\sigma) dy}_{\mathfrak{T}_1} && \text{(A.9)} \\
&\quad + \underbrace{\int_{\mathbb{R}} \frac{2\sigma(y-\sigma^2-1)}{\sqrt{2\pi}} \exp\left(-\frac{y^2}{2\sigma^2}\right) dy}_{\mathfrak{T}_2} && \text{(A.10)} \\
&\quad + \underbrace{\frac{\sigma^2}{2\pi}\int_{\mathbb{R}} \frac{\exp(-\frac{y^2}{2\sigma^2})\exp(-\frac{(y-\sigma^2)^2}{2\sigma^2})}{\mathcal{Q}(-(y-\sigma^2)/\sigma)} dy}_{\mathfrak{T}_3}\,. && \text{(A.11)}
\end{aligned}$$

We separately handle the calculation of the three terms, $\mathfrak{T}_1$, $\mathfrak{T}_2$, and $\mathfrak{T}_3$ in equations (A.9), (A.10), and (A.11), respectively.

**Calculation of $\mathfrak{T}_1$**

$$\begin{aligned}
\mathfrak{T}_1 &= \int_{\mathbb{R}} (y-\sigma^2-1)^2 \exp\left(-\frac{(2y-\sigma^2)}{2}\right)\mathcal{Q}(-(y-\sigma^2)/\sigma) dy \\
&= \int_{\mathbb{R}} (y-\sigma^2-1)^2 \exp\left(-\frac{(2y-\sigma^2)}{2}\right)\left[\int_{-(y-\sigma^2)}^{\infty} \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{z^2}{2\sigma^2}\right) dz\right] dy \\
&= \int_{\mathbb{R}} \left[\int_{-(y-\sigma^2)}^{\infty} (y-\sigma^2-1)^2 \exp\left(-\frac{(2y-\sigma^2)}{2}\right)\frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{z^2}{2\sigma^2}\right) dz\right] dy \\
&= \int_{-\infty}^{\infty} \left[\int_{-(z-\sigma^2)}^{\infty} (y-\sigma^2-1)^2 \exp\left(-\frac{(2y-\sigma^2)}{2}\right)\frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{z^2}{2\sigma^2}\right) dy\right] dz \\
&= \frac{1}{\sigma\sqrt{2\pi}}\exp\left(\frac{\sigma^2}{2}\right)\int_{-\infty}^{\infty}\exp\left(-\frac{z^2}{2\sigma^2}\right)\underbrace{\left[\int_{-(z-\sigma^2)}^{\infty}(y-\sigma^2-1)^2\exp(-y)dy\right]}_{\mathfrak{S}} dz.
\end{aligned}$$

We handle the integral within the brackets, $\mathfrak{S}$, first.

$$
\begin{aligned}
\mathfrak{S} &= \left[\int_{-(z-\sigma^2)}^{\infty} (y-\sigma^2-1)^2 \exp(-y)dy\right] \\
&= \int_{-(z-\sigma^2)}^{\infty} (y^2 - y(2\sigma^2+2) + (2\sigma^2+\sigma^4+1)) \exp(-y)dy \\
&= \underbrace{\int_{-(z-\sigma^2)}^{\infty} (y^2) \exp(-y)dy}_{\mathfrak{S}_1} \underbrace{- \int_{-(z-\sigma^2)}^{\infty} y(2\sigma^2+2) \exp(-y)dy}_{\mathfrak{S}_2} \\
&+ \underbrace{\int_{-(z-\sigma^2)}^{\infty} (2\sigma^2+\sigma^4+1) \exp(-y)dy}_{\mathfrak{S}_3} .
\end{aligned}
$$

The first of these integrals, $\mathfrak{S}_1$, is

$$
\begin{aligned}
\mathfrak{S}_1 &= \int_{-(z-\sigma^2)}^{\infty} (y^2) \exp(-y)dy = -(y^2+2y+2)\exp(-y)\Big|_{-(z-\sigma^2)}^{\infty} \\
&= [(z-\sigma^2)^2 - 2(z-\sigma^2) + 2] \exp(z-\sigma^2) \\
&= [(z-\sigma^2-2)(z-\sigma^2) + 2] \exp(z-\sigma^2) \\
&= [z^2 - z\sigma^2 - 2z - z\sigma^2 + \sigma^4 + 2\sigma^2 + 2] \exp(z-\sigma^2) \\
&= [z^2 - (2\sigma^2+2)z + \sigma^4 + 2\sigma^2 + 2] \exp(z-\sigma^2).
\end{aligned}
$$

The second of these integrals, $\mathfrak{S}_2$, is

$$
\begin{aligned}
\mathfrak{S}_2 &= -\int_{-(z-\sigma^2)}^{\infty} y(2\sigma^2+2) \exp(-y)dy \\
&= (2\sigma^2+2)\ (y+1)\exp(-y)\Big|_{-(z-\sigma^2)}^{\infty} \\
&= -(2\sigma^2+2)(-(z-\sigma^2)+1) \exp(z-\sigma^2) \\
&= (2\sigma^2+2)(z-\sigma^2-1) \exp(z-\sigma^2) \\
&= (2\sigma^2 z + 2z - 2\sigma^4 - 2\sigma^2 - 2\sigma^2 - 2) \exp(z-\sigma^2) \\
&= [z(2\sigma^2+2) - 2\sigma^4 - 4\sigma^2 - 2] \exp(z-\sigma^2).
\end{aligned}
$$

The third of these integrals, $\mathfrak{S}_3$, is

$$
\mathfrak{S}_3 = \int_{-(z-\sigma^2)}^{\infty} (2\sigma^2+\sigma^4+1) \exp(-y)dy
$$

$$
\begin{aligned}
&= -(2\sigma^2 + \sigma^4 + 1)\, \exp(-y)|_{-(z-\sigma^2)}^{\infty} \\
&= (2\sigma^2 + \sigma^4 + 1) \exp(z - \sigma^2)
\end{aligned}
$$

Therefore,

$$
\begin{aligned}
\mathfrak{S} &= \int_{-(z-\sigma^2)}^{\infty} (y^2) \exp(-y) dy - \int_{-(z-\sigma^2)}^{\infty} y(2\sigma^2 + 2) \exp(-y) dy \\
&+ \int_{-(z-\sigma^2)}^{\infty} (2\sigma^2 + \sigma^4 + 1) \exp(-y) dy \\
&= [z^2 - (2\sigma^2 + 2)z + \sigma^4 + 2\sigma^2 + 2] \exp(z - \sigma^2) + [z(2\sigma^2 + 2) - 2\sigma^4 - 4\sigma^2 - 2] \exp(z - \sigma^2) \\
&+ (2\sigma^2 + \sigma^4 + 1) \exp(z - \sigma^2) \\
&= (z^2 + 1) \exp(z - \sigma^2)
\end{aligned}
$$

Using this result above, we substitute $\mathfrak{S}$ back into the equation for $\mathfrak{T}_1$, and obtain

$$
\begin{aligned}
\mathfrak{T}_1 &= \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{\sigma^2}{2}\right) \int_{-\infty}^{\infty} \exp\left(-\frac{z^2}{2\sigma^2}\right) (z^2 + 1) \exp(z - \sigma^2) dz \\
&= \exp\left(\frac{\sigma^2}{2}\right) \int_{-\infty}^{\infty} \frac{z^2}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(z^2 - 2\sigma^2 z + 2\sigma^4)}{2\sigma^2}\right) dz \\
&+ \exp\left(\frac{\sigma^2}{2}\right) \int_{-\infty}^{\infty} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(z^2 - 2\sigma^2 z + 2\sigma^4)}{2\sigma^2}\right) dz \\
&= \int_{-\infty}^{\infty} \frac{z^2}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(z - \sigma^2)^2}{2\sigma^2}\right) dz + \int_{-\infty}^{\infty} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(z - \sigma^2)^2}{2\sigma^2}\right) dz \\
&= \sigma^4 + \sigma^2 + 1.
\end{aligned}
$$

**Calculation of $\mathfrak{T}_2$**

$$
\begin{aligned}
\mathfrak{T}_2 &= \int_{\mathbb{R}} \frac{2\sigma(y - \sigma^2 - 1)}{\sqrt{2\pi}} \exp\left(-\frac{y^2}{2\sigma^2}\right) dy \\
&= 2\sigma^2 \int_{\mathbb{R}} \frac{y}{\sigma\sqrt{2\pi}} \exp\left(-\frac{y^2}{2\sigma^2}\right) dy - \int_{\mathbb{R}} \frac{2\sigma^4 + 2\sigma^2}{\sigma\sqrt{2\pi}} \exp\left(-\frac{y^2}{2\sigma^2}\right) dy \\
&= -2\sigma^4 - 2\sigma^2.
\end{aligned}
$$

**Calculation of $\mathfrak{T}_3$**

$$
\mathfrak{T}_3 = \frac{\sigma^2}{2\pi} \int_{\mathbb{R}} \frac{\exp(-\frac{y^2}{2\sigma^2}) \exp(-\frac{(y-\sigma^2)^2}{2\sigma^2})}{\mathcal{Q}(-(y - \sigma^2)/\sigma)} dy
$$

$$
\begin{aligned}
&= \frac{\sigma^3}{2\pi} \int_{-\infty}^{\infty} \frac{\exp\left(-\frac{(u+\sigma)^2}{2}\right) \exp\left(-\frac{u^2}{2}\right)}{\mathcal{Q}(-u)} du \\
&= \frac{\sigma^3 \exp\left(-\sigma^2/2\right)}{2\pi} \int_{-\infty}^{\infty} \frac{\exp\left(-\sigma u\right) \exp\left(-u^2\right)}{\mathcal{Q}(-u)} du.
\end{aligned}
$$

Putting it all together, we finally obtain

$$
\begin{aligned}
\alpha_u &= (\mathfrak{T}_1 + \mathfrak{T}_2 + \mathfrak{T}_3)^{-1} \\
&= \left(1 - \sigma^2 - \sigma^4 + \frac{\sigma^3 \exp\left(-\sigma^2/2\right)}{2\pi} \int_{-\infty}^{\infty} \frac{\exp\left(-\sigma u\right) \exp\left(-u^2\right)}{\mathcal{Q}(-u)} du\right)^{-1}.
\end{aligned}
$$

## A.2 Derivations for the Poisson Case

Let $s_i = \langle \boldsymbol{x}, \boldsymbol{a}_i \rangle$. In the presence of Poisson noise with rate $|s_i|^2$, the measurement $y_i$ follows the distribution $y_i \sim p(y \mid |s_i|)$, where

$$
p(y \mid |s_i|) = \frac{|s_i|^{2y} e^{-|s_i|^2}}{y!}. \tag{A.12}
$$

### A.2.1 Optimal Preprocessing Function for Poisson noise

First, since $|s|$ follows a Rayleigh distribution with scale parameter $\frac{1}{\sqrt{2}}$, we have

$$
\begin{aligned}
\mathbb{E}_s\{p(y \mid |s|)\} &= \frac{1}{y!} \int_0^{+\infty} z^y \exp(-2z) dz \\
&= \frac{1}{y!} \int_0^{+\infty} \frac{1}{2} (\frac{1}{2} x)^y \exp(-x) dx \\
&= \frac{1}{y!} \frac{1}{2^{y+1}} \int_0^{\infty} x^y \exp(-x) dx \\
&= \frac{1}{y!} \frac{1}{2^{y+1}} \Gamma(y+1).
\end{aligned} \tag{A.13}
$$

Furthermore,

$$
\begin{aligned}
\mathbb{E}_s\{|s|^2 p(y \mid |s|)\} &= \int_0^{+\infty} \frac{z^{y+1}}{y!} \exp(-2z) dz \\
&= \frac{1}{y!} \int_0^{+\infty} z^{y+1} \exp(-2z) dz
\end{aligned}
$$

$$
\begin{aligned}
&= \frac{1}{y!}\int_0^{+\infty} \frac{1}{2}(\frac{1}{2}x)^{y+1}\exp(-x)dx \\
&= \frac{1}{y!}\frac{1}{2^{y+2}}\int_0^{\infty} x^{y+1}\exp(-x)dx \\
&= \frac{1}{y!}\frac{1}{2^{y+2}}\Gamma(y+2) \\
&= \frac{y+1}{y!2^{y+2}}\Gamma(y+1) \qquad (A.14)
\end{aligned}
$$

Then, plugging (A.13) and (A.14) into (A.1), we can obtain

$$\mathcal{T}(y) = 1 - \frac{\frac{1}{y!}\frac{1}{2^{y+1}}\Gamma(y+1)}{\frac{y+1}{y!2^{y+2}}\Gamma(y+1)} = 1 - \frac{2}{y+1} = \frac{y-1}{y+1}. \tag{A.15}$$

### A.2.2 Optimal Sampling Threshold for Poisson noise

Plugging (A.13) and (A.14) into (A.2), we have that

$$
\begin{aligned}
\alpha_u^{-1} &= \sum_{y\in\mathbb{R}} \frac{\left[\frac{y+1}{y!2^{y+2}}\Gamma(y+1) - \frac{1}{y!}\frac{1}{2^{y+1}}\Gamma(y+1)\right]^2}{\frac{1}{y!}\frac{1}{2^{y+1}}\Gamma(y+1)} \qquad (A.16) \\
&= \sum_{y\in\mathbb{R}} \frac{\left[\frac{y+1}{y!2^{y+2}}\Gamma(y+1) - \frac{1}{y!}\frac{1}{2^{y+1}}\Gamma(y+1)\right]^2}{\frac{1}{y!}\frac{1}{2^{y+1}}\Gamma(y+1)} \qquad (A.17) \\
&= \sum_{y\in\mathbb{R}} \frac{\left[\frac{y+1}{y!2^{y+2}}\Gamma(y+1) - \frac{1}{y!}\frac{2}{2^{y+2}}\Gamma(y+1)\right]^2}{\frac{1}{y!}\frac{1}{2^{y+1}}\Gamma(y+1)} \\
&= \sum_{y\in\mathbb{R}} \frac{\left[\frac{y-1}{y!2^{y+2}}\Gamma(y+1)\right]^2}{\frac{1}{y!}\frac{1}{2^{y+1}}\Gamma(y+1)} \\
&= \sum_{y\in\mathbb{R}} \frac{\left[\frac{y-1}{y!2^{y+2}}\right]^2\Gamma(y+1)}{\frac{1}{y!}\frac{1}{2^{y+1}}} \\
&= \sum_{y\in\mathbb{R}} \frac{(y-1)^2}{y!2^{2y+3}}\Gamma(y+1) \\
&= \frac{1}{8}\sum_{y\in\mathbb{R}} \frac{y^2-2y+1}{y!2^y}\left[\int_0^{+\infty} x^y\exp(-x)dx\right] \\
&= \frac{1}{8}\int_0^{+\infty}\sum_{y\in\mathbb{R}}\left[\left(\frac{x}{2}\right)^y\frac{\exp\left(-\frac{x}{2}\right)}{y!}(y^2-2y+1)\right]\exp\left(-\frac{x}{2}\right)dx
\end{aligned}
$$

$$
\begin{aligned}
&= \frac{1}{8}\int_0^{+\infty} \mathbb{E}_{y\sim\text{Poisson}(\frac{x}{2})}\left[y^2-2y+1\right]\exp\left(-\frac{x}{2}\right)dx \\
&= \frac{1}{8}\int_0^{+\infty} \left[\left(\frac{x}{2}\right)+\left(\frac{x}{2}\right)^2-2\left(\frac{x}{2}\right)+1\right]\exp\left(-\frac{x}{2}\right)dx \\
&= \frac{1}{8}\int_0^{+\infty} \left(\frac{x}{2}\right)^2\exp\left(-\frac{x}{2}\right)dx-\frac{1}{8}\int_0^{+\infty}\left(\frac{x}{2}\right)\exp\left(-\frac{x}{2}\right)dx+\frac{1}{8}\int_0^{+\infty}\exp\left(-\frac{x}{2}\right)dx \\
&= \frac{1}{4}\int_0^{+\infty} u^2\exp(-u)du-\frac{1}{4}\int_0^{+\infty}u\exp(-u)du+\frac{1}{4}\int_0^{+\infty}\exp(-u)du \\
&= \frac{1}{4}[2-1+1]=\frac{1}{2} \qquad \text{(A.18)}
\end{aligned}
$$

Therefore, $\alpha_u = 2$.

# Appendix B

# Derivations for Chapter 5

In this appendix, we provide the derivation of Algorithm 5.1. Recall that $\boldsymbol{B}^\star$ is the ground truth signal defined on the graph, to be estimated by $\hat{\boldsymbol{B}}$. The dictionary matrix is $\boldsymbol{X}$, and $\boldsymbol{Y}$ are the noisy measurements. The matrix $\boldsymbol{\Delta}$ is the oriented incidence matrix given from the graph structure. We look to solve an optimization problem of the form,

$$\hat{\boldsymbol{B}} = \mathrm{argmin}_{\boldsymbol{B}\in\mathbb{R}^{d\times n}} \quad \frac{1}{2}\|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{B}\|_F^2 + h_1(\boldsymbol{B}) + h_2(\boldsymbol{Z}), \text{ s.t. } \boldsymbol{Z} = \boldsymbol{\Delta}\boldsymbol{B}^\top. \tag{B.1}$$

Here, $h_1, h_2$ are the signal- and graph-regularization functions, respectively, where the parameters for computing the penalty functions are omitted here. The penalty function definitions can be found in Section 5.2.1.

We derive the augmented Lagrangian function for the above optimization problem as,

$$\begin{aligned}\mathcal{L}\left(\boldsymbol{B}, \boldsymbol{Z}, \boldsymbol{U}\right) = &\frac{1}{2}\|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{B}\|_F^2 + h_1(\boldsymbol{B}) + h_2(\boldsymbol{Z}) \\ &+ \frac{\tau}{2}\|\boldsymbol{\Delta}\boldsymbol{B}^\top - \boldsymbol{Z} + \boldsymbol{U}\|_F^2 - \frac{\tau}{2}\|\boldsymbol{U}\|_F^2,\end{aligned} \tag{B.2}$$

where $\tau$ is a tuning parameter, and the dual variable $\boldsymbol{U}$ is introduced to quantify the difference between $\boldsymbol{\Delta}\boldsymbol{B}^\top$ and $\boldsymbol{Z}$. Then, we can consider the following update rules according to alternating direction method of multipliers (ADMM):

$$\boldsymbol{B}^+ \leftarrow \arg\min_{\boldsymbol{B}} \mathcal{L}\left(\boldsymbol{B}, \boldsymbol{Z}, \boldsymbol{U}\right), \tag{B.3}$$

$$\boldsymbol{Z}^{+} \leftarrow \arg\min_{\boldsymbol{Z}} \mathcal{L}\left(\boldsymbol{B}^{+}, \boldsymbol{Z}, \boldsymbol{U}\right), \tag{B.4}$$

$$\boldsymbol{U}^{+} \leftarrow \boldsymbol{U} + (\boldsymbol{\Delta}\boldsymbol{B}^{+\top} - \boldsymbol{Z}^{+}). \tag{B.5}$$

With regard to updating $\boldsymbol{B}$, we aim to solve the problem,

$$\boldsymbol{B}^{+} = \arg\min_{\boldsymbol{B}} \left\{ \underbrace{\frac{1}{2}\|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{B}\|_F^2 + \frac{\tau}{2}\|\boldsymbol{\Delta}\boldsymbol{B}^\top - \boldsymbol{Z} + \boldsymbol{U}\|_F^2}_{:=f(\boldsymbol{B})} + h_1(\boldsymbol{B}) \right\}. \tag{B.6}$$

However, the above problem (B.6) does not admit a closed-form solution. One could use an iterative algorithm such as proximal gradient descent to obtain an exact solution. To reduce computational complexity, we use an inexact solver, which first finds the gradient calculation of $f(\boldsymbol{B})$ and then applies a proximal map to the minimizer. In this sense, Algorithm 5.1 is an ”inexact” ADMM algorithm [49]. Now, we return to deriving the inexact solution to (B.6) as follows. The gradient calculation is derived as,

$$\boldsymbol{0} \in \nabla_{\boldsymbol{B}} f\left(\boldsymbol{B}\right)\big|_{\boldsymbol{B}=\boldsymbol{B}^{+}} \tag{B.7}$$

$$\Rightarrow \boldsymbol{0} = \boldsymbol{X}^\top(\boldsymbol{X}\boldsymbol{B}^{+} - \boldsymbol{Y}) + \tau(\boldsymbol{B}^{+}\boldsymbol{\Delta}^\top + \boldsymbol{U}^\top - \boldsymbol{Z}^\top)\boldsymbol{\Delta} \tag{B.8}$$

$$\Rightarrow \boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{B}^{+} + \tau\boldsymbol{B}^{+}\boldsymbol{\Delta}^\top\boldsymbol{\Delta} = \boldsymbol{X}^\top\boldsymbol{Y} + \tau\boldsymbol{Z}^\top\boldsymbol{\Delta} - \tau\boldsymbol{U}^\top\boldsymbol{\Delta}. \tag{B.9}$$

The last line of the update rule for $\boldsymbol{B}$ shows that we need to solve Sylvester's equation, $\tilde{\boldsymbol{A}}\tilde{\boldsymbol{X}} + \tilde{\boldsymbol{X}}\tilde{\boldsymbol{B}} = \tilde{\boldsymbol{Q}}$, where

$$\tilde{\boldsymbol{X}} = \boldsymbol{B}^{+}, \tilde{\boldsymbol{A}} = \boldsymbol{X}^\top\boldsymbol{X}, \tilde{\boldsymbol{B}} = \tau\boldsymbol{\Delta}^\top\boldsymbol{\Delta}, \tag{B.10}$$

$$\tilde{\boldsymbol{Q}} = \boldsymbol{X}^\top\boldsymbol{Y} + \tau\boldsymbol{Z}^\top\boldsymbol{\Delta} - \tau\boldsymbol{U}^\top\boldsymbol{\Delta}. \tag{B.11}$$

Then, the proximal update is clear:

$$\boldsymbol{B}^{+} = \arg\min_{\boldsymbol{D}} \left\{ \frac{1}{2}\|\boldsymbol{D} - \boldsymbol{B}\|_F^2 + h_1(\boldsymbol{D}) \right\}, \tag{B.12}$$

$$:= \overline{\text{prox}}_{h_1}\left(\boldsymbol{B}\right), \tag{B.13}$$

where the $\overline{\text{prox}}_h(\cdot)$ is the matrix-valued definition of the proximal operator.

With regard to updating $\boldsymbol{Z}$,

$$\begin{aligned}
\boldsymbol{Z}^{+} &= \arg\min_{\boldsymbol{Z}} \mathcal{L}\left(\boldsymbol{B}^{+}, \boldsymbol{Z}, \boldsymbol{U}\right) && \text{(B.14)} \\
&= \arg\min_{\boldsymbol{Z}} \left\{ \frac{1}{2} \|\boldsymbol{\Delta} \boldsymbol{B}^{\top} - \boldsymbol{Z} + \boldsymbol{U}\|_F^2 + \frac{1}{\tau} h_2(\boldsymbol{Z}) \right\} && \text{(B.15)} \\
&= \overline{\text{prox}}_{h_2/\tau}\left(\boldsymbol{\Delta} \boldsymbol{B}^{+\top} + \boldsymbol{U}\right), && \text{(B.16)}
\end{aligned}$$

In the main text, each column is denoted to be updated in parallel to remain consistent with the typical vector-valued proximal operator that is presented.

# Bibliography

[1] B. L. Adams, S. I. Wright, and K. Kunze. Orientation imaging: the emergence of a new microscopy. *Metallurgical Transactions A*, 24(4):819–831, 1993. 5.5

[2] H. K. Aggarwal, M. P. Mani, and M. Jacob. MoDL: Model-based deep learning architecture for inverse problems. *IEEE Transactions on Medical Imaging*, 38(2):394–405, 2019. 4.1

[3] R. Ahmad, C. A. Bouman, G. T. Buzzard, S. Chan, S. Liu, E. T. Reehorst, and P. Schniter. Plug-and-play methods for magnetic resonance imaging: Using denoisers for image recovery. *IEEE Signal Processing Magazine*, 37(1):105–116, 2020. 4.1

[4] S. Bahmani and J. Romberg. Phase retrieval meets statistical learning theory: A flexible convex relaxation. In *Artificial Intelligence and Statistics*, pages 252–260, 2017. 2.1

[5] C. Bao, G. Barbastathis, H. Ji, Z. Shen, and Z. Zhang. Coherence retrieval using trace regularization. *SIAM Journal on Imaging Sciences*, 11(1):679–706, 2018. 3.1

[6] R. H. Bartels and G. W. Stewart. Solution of the matrix equation AX + XB = C. *Commun. ACM*, 15(9):820–826, sep 1972. 5.3

[7] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009. 1.3.2, 5.1.1

[8] J. Bergstra, D. Yamins, D. D. Cox, et al. Hyperopt: A python library for optimizing the hyperparameters of machine learning algorithms. In *Proceedings of the 12th Python in Science Conference*, volume 13, page 20. Citeseer, 2013. 5.4

[9] M. Bertero and P. Boccacci. *Introduction to Inverse Problems in Imaging*. CRC Press,

2020. 1.1

[10] S. Bhojanapalli, B. Neyshabur, and N. Srebro. Global optimality of local search for low rank matrix recovery. In *Advances in Neural Information Processing Systems*, pages 3873–3881, 2016. 2.1.1

[11] F. Boulogne, J. D. Warner, and E. Neil Yager. Scikit-image: Image processing in Python. *J. PeerJ*, 2:453, 2014. 4.4.2

[12] A. Buades, B. Coll, and J.-M. Morel. Non-Local Means Denoising. *Image Processing Online*, 1:208–212, 2011. 4.1

[13] T. Cai and A. Zhang. ROP: Matrix recovery via rank-one projections. *The Annals of Statistics*, 43(1):102–138, 2015. 3.1

[14] P. G. Callahan and M. De Graef. Dynamical electron backscatter diffraction patterns. part i: Pattern simulations. *Microscopy and Microanalysis*, 19(5):1255–1265, 2013. 5.5

[15] E. J. Candès, X. Li, and M. Soltanolkotabi. Phase retrieval via Wirtinger flow: Theory and algorithms. *IEEE Transactions on Information Theory*, 61(4):1985–2007, April 2015. 2.1, 2.1.1, 3.1

[16] E. J. Candès, T. Strohmer, and V. Voroninski. Phaselift: Exact and stable signal recovery from magnitude measurements via convex programming. *Communications on Pure and Applied Mathematics*, 66(8):1241–1274, 2013. 3.1

[17] E. J. Candès and T. Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE Transactions on Information Theory*, 52(12):5406–5425, Dec. 2006. 1.3.1

[18] J. R. Chang, C.-L. Li, B. Poczos, B. Vijaya Kumar, and A. C. Sankaranarayanan. One network to solve them all–solving linear inverse problems using deep projection models. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5888–5897, 2017. 4.1

[19] P. Chen, A. Fannjiang, and G.-R. Liu. Phase retrieval with one or two diffraction patterns

by alternating projections with the null initialization. *Journal of Fourier Analysis and Applications*, pages 1–40, 2015. 2.1.1

[20] Y. Chen and E. J. Candès. Solving random quadratic systems of equations is nearly as easy as solving linear systems. *Comm. Pure Appl. Math.*, 70(5):822–883, 2017. 2.1, 2.1.1, 3.1, 3.1.1

[21] Y. Chen, Y. Chi, J. Fan, and C. Ma. Gradient descent with random initialization: Fast global convergence for nonconvex phase retrieval. *Mathematical Programming*, 176(1):5–37, 2019. 3.5

[22] Y. Chen, Y. Chi, J. Fan, and C. Ma. Spectral methods for data science: A statistical perspective. *Foundations and Trends® in Machine Learning*, 14(5), 2021. 2.1

[23] Y. Chen, Y. Chi, and A. J. Goldsmith. Exact and stable covariance estimation from quadratic sampling via convex programming. *IEEE Transactions on Information Theory*, 61(7):4034–4059, 2015. 3.1

[24] Y. H. Chen, S. U. Park, D. Wei, G. Newstadt, M. A. Jackson, J. P. Simmons, M. De Graef, and A. O. Hero. A dictionary approach to electron backscatter diffraction indexing. *Microscopy and Microanalysis*, 21(3):739–752, 2015. 5.1.1, 5.5

[25] D. Chernoff and P. Stark. Principles of magnetic resonance imaging. *Waltham MA*, 2010. 1.1

[26] Y. Chi and H. Fu. Subspace learning from bits. *IEEE Transactions on Signal Processing*, 65(17):4429–4442, 2017. 2.1.1

[27] Y. Chi and Y. M. Lu. Kaczmarz method for solving quadratic equations. *IEEE Signal Processing Letters*, 23(9):1183–1187, 2016. 2.1.1

[28] Y. Chi, Y. M. Lu, and Y. Chen. Nonconvex optimization meets low-rank matrix factorization: An overview. *IEEE Transactions on Signal Processing*, 67(20):5239–5269, 2019. 2.1, 3.1

[29] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3-D transform-domain collaborative filtering. *IEEE Transactions on Image Processing*,

16(8):2080–2095, 2007. 4.1, 4.4.2

[30] A. Dave, A. K. Vadathya, R. Subramanyam, R. Baburajan, and K. Mitra. Solving inverse computational imaging problems using deep pixel-level prior. *IEEE Transactions on Computational Imaging*, 5(1):37–51, 2018. 4.1

[31] M. E. Davies and Y. C. Eldar. Rank awareness in joint sparse recovery. *IEEE Transactions on Information Theory*, 58(2):1135–1146, 2012. 5.1.1

[32] M. De Graef. Public EMsoft repository. `https://github.com/EMsoft-org/EMsoft`. Also, see: http://vbff.materials.cmu.edu/emsoft/. 5.5, 5.6, 5.6

[33] A. Defazio, F. Bach, and S. Lacoste-Julien. SAGA: a fast incremental gradient method with support for non-strongly convex composite objectives. In *Proceedings of the 27th International Conference on Neural Information Processing Systems*, pages 1646–1654, 2014. 4.5

[34] M. Defferrard, L. Martin, R. Pena, and N. Perraudin. PyGSP: Graph signal processing in python. 5.4

[35] D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006. 1.3.1

[36] D. L. Donoho and J. M. Johnstone. Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, 81(3):425–455, 1994. 4.4.2

[37] Y. C. Eldar and H. Rauhut. Average case analysis of multichannel sparse recovery using convex relaxation. *IEEE Transactions on Information Theory*, 56(1):505–519, 2009. 5.1.1

[38] J. Fan and R. Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, 96(456):1348–1360, 2001. 5.2.1

[39] J. R. Fienup. Phase retrieval algorithms: a comparison. *Applied Optics*, 21:2758–2769, 1982. 3.1

[40] M. A. Figueiredo, R. D. Nowak, and S. J. Wright. Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems. *IEEE Journal of Selected Topics in Signal Processing*, 1(4):586–597, 2007. 5.1.1

[41] S. Foucart and H. Rauhut. An invitation to compressive sensing. In *A mathematical introduction to compressive sensing*, pages 1–39. Springer, 2013. (document), 1.2, 5.2.1

[42] R. Fu, V. Monardo, T. Huang, and Y. Liu. Deep unfolding network for block-sparse signal recovery. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2880–2884. IEEE, 2021. 5.1.1

[43] R. Fu, V. Monardo, T. Huang, and Y. Liu. Theoretical linear convergence of deep unfolding network for block-sparse signal recovery. *arXiv preprint arXiv:2111.09801*, 2021. 5.1.1

[44] R. Ge, J. D. Lee, and T. Ma. Matrix completion has no spurious local minimum. In *Advances in Neural Information Processing Systems*, pages 2973–2981, 2016. 2.1.1

[45] R. W. Gerchberg. A practical algorithm for the determination of phase from image and diffraction plane pictures. *Optik*, 35:237, 1972. 3.2.4

[46] T. Goldstein and C. Studer. Phasemax: Convex phase retrieval via basis pursuit. *IEEE Transactions on Information Theory*, 64(4):2675–2689, 2018. 2.1

[47] K. Gregor and Y. Lecun. Learning fast approximations of sparse coding. In *International Conference on International Conference on Machine Learning*, pages 399–406, 2010. 5.1.1

[48] A. Hagberg, P. Swart, and D. S Chult. Exploring network structure, dynamics, and function using NetworkX. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008. 5.4

[49] W. W. Hager and H. Zhang. Convergence rates for an inexact admm applied to separable convex optimization. *Computational Optimization and Applications*, 77(3):729–754, 2020. B

[50] K. Hammernik, T. Klatzer, E. Kobler, M. P. Recht, D. K. Sodickson, T. Pock, and F. Knoll. Learning a variational network for reconstruction of accelerated MRI data. *Magnetic*

*Resonance in Medicine*, 79(6):3055–3071, 2018. 4.1

[51] J. Huang, P. Breheny, and S. Ma. A selective review of group selection in high-dimensional models. *Statistical science: a review journal of the Institute of Mathematical Statistics*, 27(4), 2012. 5.2.2, 5.3

[52] M. Huang and Z. Xu. Solving systems of quadratic equations via exponential-type gradient descent algorithm. *Journal of Computational Mathematics*, 38(4):638, 2020. 3.1.1

[53] R. Johnson and T. Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in Neural Information Processing Systems*, 1(3):1–9, 2013. 4.1, 4.2.2, 4.2.2

[54] E. Jones, T. Oliphant, P. Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. 5.3

[55] U. S. Kamilov, H. Mansour, and B. Wohlberg. A plug-and-play priors approach for solving nonlinear imaging inverse problems. *IEEE Signal Processing Letters*, 24(12):1872–1876, 2017. 4.1

[56] R. H. Keshavan, A. Montanari, and S. Oh. Matrix completion from a few entries. *IEEE Transactions on Information Theory*, 56(6):2980 –2998, June 2010. 2.1.1

[57] A. Y. Kruger. On Fréchet subdifferentials. *Journal of Mathematical Sciences*, 116(3):3325–3358, 2003. 3.2.2

[58] N. C. K. Lassen. Automated determination of crystal orientations from electron backscattering patterns. 1994. 5.5

[59] N. K. Lassen, D. J. Jensen, and K. Conradsen. Image processing procedures for analysis of electron back scattering patterns. *Scanning Microscopy*, 6:115–121, 1992. 5.5

[60] J. D. Lee, M. Simchowitz, M. I. Jordan, and B. Recht. Gradient descent only converges to minimizers. In *Conference on Learning theory*, pages 1246–1257. PMLR, 2016. 2.1.1

[61] W. Lenthe, S. Singh, and M. De Graef. A spherical harmonic transform approach to the

indexing of electron back-scattered diffraction patterns. *Ultramicroscopy*, 207:112841, 2019. 5.1.1, 5.5

[62] K.-C. Li. On principal Hessian directions for data visualization and dimension reduction: Another application of stein's lemma. *Journal of the American Statistical Association*, 87(420):1025–1039, 1992. 2.1

[63] Y. Li and Y. Chi. Off-the-grid line spectrum denoising and estimation with multiple measurement vectors. *IEEE Transactions on Signal Processing*, 64(5):1257–1269, 2015. 5.1.1

[64] Y. Li, C. Ma, Y. Chen, and Y. Chi. Nonconvex matrix factorization from rank-one measurements. *IEEE Transactions on Information Theory*, 67(3):1928–1950, 2021. 3.1, 3.1, 3.1, 3.1.1, 3.2.5, 3.3

[65] Y. Li, Y. Sun, and Y. Chi. Low-rank positive semidefinite matrix recovery from corrupted rank-one measurements. *IEEE Transactions on Signal Processing*, 65(2):397–408, 2017. 3.1

[66] Y. M. Lu and G. Li. Phase transitions of spectral initialization for high-dimensional nonconvex estimation. *Information and Inference: A Journal of the IMA*, 9(3):507–541, 2020. 2.1, 2.1, 2.1.1, 2.2, 2.2.1, 2.2.2, 2.5

[67] W. Luo, W. Alghamdi, and Y. M. Lu. Optimal spectral initialization for signal recovery with applications to phase retrieval. *IEEE Transactions on Signal Processing*, 67(9):2347–2356, 2019. 2.2.2, 1

[68] C. Ma, K. Wang, Y. Chi, and Y. Chen. Implicit regularization in nonconvex statistical estimation: Gradient descent converges linearly for phase retrieval, matrix completion, and blind deconvolution. *Foundations of Computational Mathematics*, 2019. 2.1, 2.1.1, 3.1

[69] J. Ma, J. Xu, and A. Maleki. Approximate message passing for amplitude based optimization. In *International Conference on Machine Learning*, pages 3365–3374. PMLR, 2018. 3.1.1

[70] S. Ma and J. Huang. A concave pairwise fusion approach to subgroup analysis. *Journal of the American Statistical Association*, 112(517):410–423, 2017. 5.4.1

[71] S. G. Mallat and Z. Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal processing*, 41(12):3397–3415, 1993. 5.1.1

[72] L. Meier, S. Van De Geer, and P. Bühlmann. The group lasso for logistic regression. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(1):53–71, 2008. 5.1.1

[73] V. Monardo and Y. Chi. On the sensitivity of spectral initialization for noisy phase retrieval. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5172–5176. IEEE, 2019. 2.1

[74] V. Monardo, A. Iyer, S. Donegan, M. De Graef, and Y. Chi. Plug-and-play image reconstruction meets stochastic variance-reduced gradient methods. In *2021 IEEE International Conference on Image Processing*, pages 2868–2872. IEEE, 2021. 4.1

[75] V. Monardo, Y. Li, and Y. Chi. Solving quadratic equations via amplitude-based nonconvex optimization. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5526–5530. IEEE, 2019. 3.1

[76] M. Mondelli and A. Montanari. Fundamental limits of weak recovery with applications to phase retrieval. *Foundations of Computational Mathematics*, 19(3):703–773, 2019. 2.1, 2.1, 2.1.1, 2.2, 2.2.2, 2.2.2, 2.3, 2.3, A

[77] V. Monga, Y. Li, and Y. C. Eldar. Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing. *IEEE Signal Processing Magazine*, 38(2):18–44, 2021. 4.1

[78] P. Netrapalli, P. Jain, and S. Sanghavi. Phase retrieval using alternating minimization. *Advances in Neural Information Processing Systems (NIPS)*, 2013. 2.1

[79] I. Newton. In [experimental] philosophy particular propositions are inferred from the phenomena and afterwards rendered general by induction. *Principia*, 2:392, 1729. 1.1

[80] L. M. Nguyen, J. Liu, K. Scheinberg, and M. Takáč. SARAH: A novel method for machine learning problems using stochastic recursive gradient. In *International Conference on Machine Learning*, pages 2613–2621. PMLR, 2017. 4.5

[81] G. Ongie, A. Jalal, C. A. Metzler, R. G. Baraniuk, A. G. Dimakis, and R. Willett. Deep learning techniques for inverse problems in imaging. *IEEE Journal on Selected Areas in Information Theory*, 1(1):39–56, 2020. 4.1

[82] E. T. Reehorst and P. Schniter. Regularization by denoising: Clarifications and new interpretations. *IEEE Transactions on Computational Imaging*, 5(1):52–67, 2018. 4.2.1

[83] D. Richards, S. N. Negahban, and P. Rebeschini. Decentralised sparse multi-task regression. *arXiv preprint arXiv:1912.01417*, 2019. 5.1.1

[84] S. Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016. 1.3.2

[85] E. Ryu, J. Liu, S. Wang, X. Chen, Z. Wang, and W. Yin. Plug-and-play methods provably converge with properly trained denoisers. In *International Conference on Machine Learning*, pages 5546–5557. PMLR, 2019. 4.1, 4.2.1, 4.4

[86] S. Sanghavi, R. Ward, and C. D. White. The local convexity of solving systems of quadratic equations. *Results in Mathematics*, pages 1–40, 2016. 3.1, 3.1, 3.1.1

[87] Y. Shechtman, Y. C. Eldar, O. Cohen, H. N. Chapman, J. Miao, and M. Segev. Phase retrieval with application to optical imaging: a contemporary overview. *IEEE Signal Processing Magazine*, 32(3):87–109, 2015. 3.1

[88] M. Soltani and C. Hegde. Improved algorithms for matrix recovery from rank-one projections. *arXiv preprint arXiv:1705.07469*, 2017. 3.1.1

[89] S. Sreehari, S. V. Venkatakrishnan, K. L. Bouman, J. P. Simmons, L. F. Drummy, and C. A. Bouman. Multi-Resolution Data Fusion for Super-Resolution Electron Microscopy. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, volume 2017-July, pages 1084–1092, 2017. 4.1

[90] S. Sreehari, S. V. Venkatakrishnan, B. Wohlberg, G. T. Buzzard, L. F. Drummy, J. P. Simmons, and C. A. Bouman. Plug-and-play priors for bright field electron tomography and sparse interpolation. *IEEE Transactions on Computational Imaging*, 2(4):408–423, 2016. 4.1, 4.2.1, 4.5

[91] J. Sun, Q. Qu, and J. Wright. A geometric analysis of phase retrieval. In *2016 IEEE International Symposium on Information Theory*, pages 2379–2383. IEEE, 2016. 2.1.1

[92] Y. Sun, B. Wohlberg, and U. S. Kamilov. An online plug-and-play algorithm for regularized image reconstruction. *IEEE Transactions on Computational Imaging*, 5(3):395–408, 2019. 4.1, 4.2.1

[93] Y. Sun, Z. Wu, X. Xu, B. Wohlberg, and U. S. Kamilov. Scalable plug-and-play admm with convergence guarantees. *IEEE Transactions on Computational Imaging*, 7:849–863, 2021. 4.5

[94] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996. 5.2.1

[95] T. Tong, C. Ma, and Y. Chi. Accelerating ill-conditioned low-rank matrix estimation via scaled gradient descent. *Journal of Machine Learning Research*, 22(150):1–63, 2021. 3.4

[96] T. Tong, C. Ma, and Y. Chi. Low-rank matrix recovery with scaled subgradient methods: Fast and robust convergence without the condition number. *IEEE Transactions on Signal Processing*, 69:2396–2409, 2021. 3.4

[97] T. Tong, C. Ma, A. Prater-Bennette, E. Tripp, and Y. Chi. Scaling and scalability: Provable nonconvex low-rank tensor estimation from incomplete measurements. *arXiv preprint arXiv:2104.14526*, 2021. 3.4

[98] J. A. Tropp and A. C. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on Information Theory*, 53(12):4655–4666, 2007. 5.1.1

[99] S. Tu, R. Boczar, M. Simchowitz, M. Soltanolkotabi, and B. Recht. Low-rank solutions

of linear matrix equations via Procrustes flow. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning-Volume 48*, pages 964–973. JMLR. org, 2016. 2.1.1

[100] N. H. S. (UK). MRI scan: How it is performed. `https://www.nhs.uk/conditions/mri-scan/what-happens/`, Aug. 2021. 1.2

[101] R. Varma, H. Lee, J. Kovačević, and Y. Chi. Vector-valued graph trend filtering with non-convex penalties. *IEEE Transactions on Signal and Information Processing over Networks*, 6:48–62, 2019. 5.1.1, 5.2.1, 5.2.2, 5.7

[102] R. A. Varma. *Exploiting Structure In Data: Sampling and Signal Processing on Graphs*. PhD thesis, Carnegie Mellon University, 2020. 5.1.1

[103] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg. Plug-and-play priors for model based reconstruction. In *2013 IEEE Global Conference on Signal and Information Processing*, pages 945–948. IEEE, 2013. 4.1

[104] I. Waldspurger, A. d'Aspremont, and S. Mallat. Phase recovery, maxcut and complex semidefinite programming. *Mathematical Programming*, 149(1-2):47–81, 2015. 3.1

[105] G. Wang, G. B. Giannakis, and Y. C. Eldar. Solving systems of random quadratic equations via truncated amplitude flow. *IEEE Transactions on Information Theory*, 64(2):773–794, 2018. 2.1, 3.1.1

[106] Y.-X. Wang, J. Sharpnack, A. Smola, and R. Tibshirani. Trend filtering on graphs. In *Artificial Intelligence and Statistics*, pages 1042–1050. PMLR, 2015. 5.1.1

[107] L. Xiao and T. Zhang. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075, 2014. 4.1, 4.2.2, 4.3

[108] P. Young, A. Lai, M. Hodosh, and J. Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *TACL*, 2:67–78, 2014. 4.4

[109] H. Zhang, Y. Chi, and Y. Liang. Provable non-convex phase retrieval with outliers: Median

truncated Wirtinger flow. In *International Conference on Machine Learning*, pages 1022–1031, 2016. 2.1, 3.5

[110] H. Zhang, Y. Zhou, Y. Liang, and Y. Chi. A nonconvex approach for phase retrieval: Reshaped Wirtinger flow and incremental algorithms. *The Journal of Machine Learning Research*, 18(1):5164–5198, 2017. 2.1, 3.1.1, 3.3.2

[111] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang. Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155, 2017. 4.4

[112] J. Zhao, Z. Sihao, and Z. Jing. Review of the sparse coding and the applications on image retrieval. In *2016 International Conference on Communication and Electronics Systems (ICCES)*, pages 1–5. IEEE, 2016. 5.1

[113] K. Zhong, P. Jain, and I. S. Dhillon. Efficient matrix sensing using rank-1 Gaussian measurements. In *International Conference on Algorithmic Learning Theory*, pages 3–18. Springer, 2015. 3.1.1