

© 2023 Sidharth Gupta

MEASUREMENTS AND OPERATORS: REWORKING THE ESSENTIAL
IMAGING ELEMENTS

BY

SIDHARTH GUPTA

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois Urbana-Champaign, 2023

Urbana, Illinois

Doctoral Committee:

Associate Professor Ivan Dokmanić, Chair
Professor Yoram Bresler
Professor Andrew C. Singer
Associate Professor Zhizhen Zhao

ABSTRACT

We solve imaging problems by recovering, transforming, and modifying what we have—the measurements and the operator.

In the first parts of this thesis, we formulate techniques for imaging problems where the phase of the measurements is lost. We begin by showing how to use distance geometry to recover the missing phase of these measurements. This enables us to rapidly perform matrix-vector optically. Knowing the measurement phase is also useful because it transforms the cumbersome quadratic problem of optical imaging operator calibration into a simple linear problem which, for typical sized operators, can be solved in minutes rather than hours. However, unfortunately calibrated operators have errors which can result in poor reconstructions when used for imaging. This motivates us to next create a total least squares framework to account of operator errors when imaging with phaseless measurements. The effectiveness of all these phase problem techniques is verified by experiments on optical hardware.

We then use deep learning for scenarios where measurements are scarce and training data is not available. Instead of learning a map from measurements to target reconstructions, we learn a map from measurements to low-dimensional projections of the target. The projections are viewed as measurements from a imaging operator built of multiple projection operators. Solving this new inverse problem gives reconstructions that are more robust to measurement noise.

Next, we solve imaging problems when the measurement coordinates such as sensor locations or projection angles are uncertain. Using implicit neural networks and differentiable splines, we learn a representation of the measurements which can be evaluated at any measurement coordinate. By optimizing the inputs at the same time as the measurement representation parameters, we learn the unknown measurement coordinates. To ensure consistency we also jointly reconstruct the final image during the optimization.

To my parents, Rachna and Sunil, for their love, encouragement and support.

ACKNOWLEDGMENTS

Thank you to my advisor, Professor Ivan Dokmanić. Ivan, you always advised me in the way that was best for me, and I greatly benefited from this. In particular, I am grateful for how you balanced nudging me when I needed it, with giving me the freedom to figure out my own path.

I would also like to thank my thesis committee members, Professors Yoram Bresler, Andrew C. Singer and Zhizhen Zhao for their teaching and feedback throughout my time at UIUC. Furthermore, a large part of the work in this thesis is the result of collaborations with Professors Rémi Gribonval and Laurent Daudet, and backing from the team at LightOn. With their help, I was able to explore a broader variety of areas than I originally envisioned. An important thank you also goes to Professor Richard Turner for introducing me to academic research and for always making time for me.

I am fortunate to have been surrounded by my dear friends at UIUC. Aditya, Ulaş, Jamila, Vikram, Varun, Puoya, Konik, Sam, Evan, Berk, Ufuk, Anadi, Akshayaa, Kayvon, Seiyun, Helmuth and Valentin (in approximate order of meeting), all of you made these years one of the best experiences of my life. To my friends from beyond UIUC, Harpal, Eno, Chad, Husam, Chris, Joaquín, Anuran and Gaurav, it has been wonderful to take breaks from work to catch up. Finally, Asmita, you have been my teammate and companion through this journey. Thank you for supporting me during the tough times and celebrating me during the highlights.

My family has always encouraged my interest in engineering. My mother, Rachna, was my first mathematics teacher. Sitting on the blue chair and being taught by her is one of my most vivid childhood memories. Sunil, my father, taught me to start questioning and probing deeper whenever I think I know the solution to a problem. I am also lucky to receive unlimited blessings from my grandmothers, Santosh and Usha. My younger siblings, Sahil and Sunena, have always been my fans for some reason. Both of them inspire me

with how they approach everything. Family, thank you, for everything.

TABLE OF CONTENTS

CHAPTER 1	INTRODUCTION	1
1.1	Imaging applications in this thesis	3
1.2	Thesis organization and contributions	7
CHAPTER 2	MEASUREMENT PHASE RETRIEVAL	10
2.1	Phasing optical random projections	10
2.2	The measurement phase retrieval problem	14
2.3	Experimental verification and application	20
2.4	Summary	25
CHAPTER 3	RAPID TM CALIBRATION	26
3.1	Imaging through scattering media	26
3.2	Rapid transmission matrix computation	29
3.3	Obtaining the transmission matrix	33
3.4	Experimental verification	35
3.5	Summary	37
CHAPTER 4	TOTAL LEAST SQUARES PHASE RETRIEVAL	39
4.1	Phase retrieval with sensing system errors	39
4.2	TLS for phase retrieval	43
4.3	TLS and LS solution reconstruction errors	52
4.4	TLS phase retrieval simulations	60
4.5	Experiments on real optical hardware	66
4.6	Summary	69
CHAPTER 5	RANDOM MESH PROJECTORS	70
5.1	Scarce measurements and limited training data	70
5.2	Regularization by random mesh projections	73
5.3	Numerical results	78
5.4	Summary	82
CHAPTER 6	DIFFERENTIABLE UNCALIBRATED IMAGING	84
6.1	Measurement coordinates	84
6.2	Differentiable framework	89
6.3	Experimental verification	93
6.4	Summary	103

CHAPTER 7	LOOKING FORWARD	106
7.1	Phase problems	106
7.2	Imaging inverse problems	107
REFERENCES	108
APPENDIX A	MEASUREMENT PHASE RETRIEVAL	124
A.1	Practical considerations with hardware	124
A.2	Randomized singular value decomposition (RSVD) details	126
A.3	Localization with known anchor positions	127
APPENDIX B	TOTAL LEAST SQUARES PHASE RETRIEVAL	128
B.1	Roots of cubic equations	128
B.2	Proof of Proposition 3	129
B.3	Taylor series expansions	129
B.4	Proof of Proposition 5	140
B.5	Experimental verification of Proposition 4	141
B.6	Handcrafted errors	142
B.7	Coded diffraction pattern (CDP) measurement model	145
B.8	Additional OPU experiment information	147
APPENDIX C	RANDOM MESH PROJECTORS	152
C.1	Oblique projection experiments	152
C.2	Proof of Proposition 6	152
C.3	Additional training information	154
C.4	Further reconstructions	155
C.5	Comparison of proposed method with ensemble of direct nets	157
APPENDIX D	DIFFERENTIABLE UNCALIBRATED IMAGING	159
D.1	2D CT experiments with 120 view angles	159
D.2	Additional spline information	159
D.3	Implementation details	162

CHAPTER 1

INTRODUCTION

If we have forgotten to label boxes when moving houses, how would we determine what is in a particular box without unpacking it? We might lift it to feel the weight or shake it to hear a sound. To help limit the possibilities, we will try to remember all the items that we have packed and already unpacked across all boxes. However, adverse factors such as multiple items of the same weight and extra protective padding that muffles sound makes our task harder. In many ways, this natural process of probing objects and using known information in the face of unfavorable factors is similar to what we as imaging scientists do when solving imaging inverse problems.

In a nutshell, to solve an imaging inverse problem means to recover an image of an object from its measurements. These measurements are obtained via a forward process that probes the object. As imaging scientists, our job is to work backwards in the inverse direction. Starting with the obtained measurements and what we know about the forward process, we set out to reconstruct an image of the object. To help better understand this broad description, we apply it to a range of applications:

- **X-ray computed tomography (CT):** In the forward process, parallel x-ray beams at different angles are shined through a human body and absorbed by body tissues. Attenuated beams that penetrate through are measured by detectors. Given the angles and these measurements, we seek to reconstruct an image of the body [1].
- **Crystallography:** X-ray beams are shined through a crystal specimen as it is rotated. At each rotation angle, a diffraction pattern that depends on the specimen's internal lattice structure is measured [2]. Using the patterns and corresponding rotation angles, we form an image of the specimen's internal structure.
- **Seismic traveltime tomography:** Sensors which send and receive

probing waves are placed on the surface of earth. Depending on the subsurface structure and material, the waves are received by the sensors with some delay time. Using these timings and the sensor locations, we reconstruct an image of the subsurface [3].

- **Deblurring:** We capture a blurry digital image due to an out of focus camera lens. Using the blurry image and what we know about the blurring process we deblur the image [4].
- **Super-resolution:** Due to the expense of taking high resolution scientific images, we measure a series of low resolution overlapping images. We use these images and knowledge of the low resolution image acquisition procedure to reconstruct a high resolution image [5].

Although there is a diverse assortment of imaging problems, there are common notions and challenges which need addressing. One such notion is understanding whether a reconstructed image is unique. For example, in super-resolution, if we only have one low resolution image measurement, multiple high resolution reconstructions can give the same measurements. Sensitivity of the solution is another criteria. If our measurements change due to noise, we would want the solution to remain relatively unchanged. To handle non-unique solutions and poor sensitivity, regularization strategies are typically employed [6]. This entails incorporating prior knowledge and assumptions about the image into the reconstruction method. Designing regularization strategies is another major consideration in itself.

Before we narrow down to the perspective that this thesis takes, it is helpful to build a general formulation of an imaging problem. If we denote the object whose image we want as $\mathbf{x} \in \mathcal{X}$, then we can model the forward process that gives us measurements as

$$\mathbf{y} = \mathcal{A}(\mathbf{x}) \tag{1.1}$$

where $\mathbf{y} \in \mathcal{Y}$ denotes the measurements, and $\mathcal{A} : \mathcal{X} \rightarrow \mathcal{Y}$ is an operator that models the forward process and maps from objects to measurements. The inverse problem is to work backward and estimate \mathbf{x} using what we have—the measurements and the operator. Denoting the estimate of \mathbf{x} as $\hat{\mathbf{x}}$, we

write the inverse problem solution as

$$\hat{\mathbf{x}} = \mathcal{S}(\mathbf{y}, \mathcal{A}) \quad (1.2)$$

where \mathcal{S} is an inverse problem solver that uses \mathbf{y} and \mathcal{A} to reconstruct $\hat{\mathbf{x}}$. The solver \mathcal{S} is general and methods based on signal processing, statistics, and machine learning can be derived from it. Crucially, this high level abstraction in Equation 1.2 stresses that the measurements and operator are essential elements when solving imaging problems. With this perspective on imaging in mind, in this thesis we recover, transform, and modify the essential imaging elements,

E1 Measurements \mathbf{y} :

- Retrieve the phase of complex-valued measurements when only their magnitude can be measured (Chapter 2).
- Transform measurements to a new set of measurements which are projections of the object being imaged (Chapter 5).
- Reevaluate measurements to correspond to a known operator when they originally correspond to an unknown operator (Chapter 6).

E2 Operators \mathcal{A} :

- Rapid calibration of the unknown operator when imaging through scattering media in minutes rather than hours (Chapter 3).
- Account for and correct calibration errors in the operator when imaging through scattering media (Chapter 4).
- Recover the parameters of an unknown operator which was used to acquire measurements (Chapter 6).

1.1 Imaging applications in this thesis

In this thesis, we explore **E1** and **E2** through a mixture of imaging applications. The applications that we consider can be divided into two categories: quadratic phase problems and linear imaging problems.

1.1.1 Quadratic phase problems

Phase problems are ubiquitous in the applied sciences [7]. We can understand the challenges in these problems by considering imaging through a scattering medium as an example. The prototypical setup is shown in Figure 1.1.

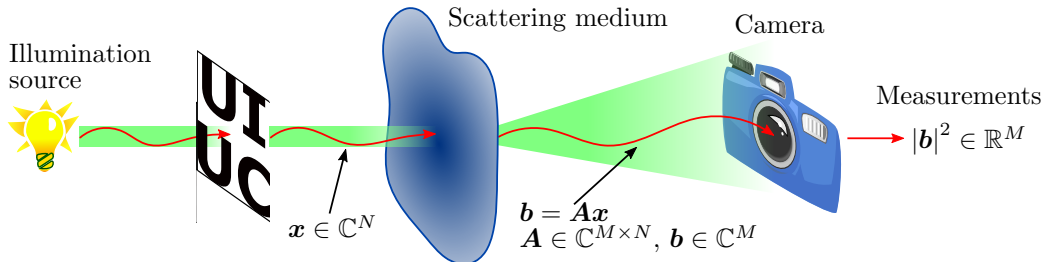


Figure 1.1: The setup for imaging through scattering media. The input optical field from illuminating an unknown pattern propagates through a scattering medium. The resulting output optical field is linearly related to the input field by an unknown transmission matrix, \mathbf{A} . A camera measures the intensity of the output field. The inverse problem is to reconstruct an image of the input pattern from the camera’s measurements.

An unknown pattern is displayed and illuminated, which results in a complex-valued input optical field, $\mathbf{x} \in \mathbb{C}^N$. This optical field propagates through a scattering medium and the resulting output optical field after scattering is denoted as $\mathbf{b} \in \mathbb{C}^M$. Because the medium is optically linear, the input field is linearly related to the output field by a complex-valued transmission matrix (TM), $\mathbf{A} \in \mathbb{C}^{M \times N}$,

$$\mathbf{b} = \mathbf{A}\mathbf{x}. \quad (1.3)$$

Our goal is to recover an image of the unknown pattern from the scattered light. This appears to be conceptually simple due to the linear relationship in (1.3). However, in practice this is not straightforward because cameras only measure intensity, $|\mathbf{b}|^2$, where $|\mathbf{b}|^2$ is the elementwise squared magnitude of the vector \mathbf{b} . As a result, the phase of the complex-valued measurements is lost. Furthermore, the TM is typically unknown. With $\mathcal{A} = |\mathbf{A}\cdot|^2$, we can frame the imaging problem using (1.1) as

$$\mathbf{y} = |\mathbf{b}|^2 = |\mathbf{A}\mathbf{x}|^2, \quad (1.4)$$

and the solution using (1.2) is $\hat{\mathbf{x}} = \mathcal{S}(|\mathbf{b}|^2, |\mathbf{A}\cdot|^2)$. Immediately we can see

that **E2** is relevant because **A** is unknown and needs to be calibrated to solve the imaging problem. Furthermore, the solver, **S**, needs to account for calibration errors.

Phaseless measurements are also obtained in crystallography [8] and ptychography [9], after the scattering of x-ray waves, and can be modeled similarly to (1.4). In astronomy, phase retrieval techniques were applied to estimate aberrations in the Hubble Space Telescope hardware from intensity measurements [10]. Accounting for the aberrations led to better reconstructions.

We can use our understanding of imaging through scattering media to solve phase problems in optical computing. Optical Processing Units (OPUs) are optics-based hardware that leverage the propagation of light in random media to perform random projections of data. Figure 1.2 shows an OPU randomly projecting data signal $\mathbf{x} \in \mathbb{R}^N$.¹ The data signal is ‘imprinted’ on laser light and shined through a random scattering medium to produce the output optical field **b**. As in Figure 1.1, the output and input are linearly related by an unknown TM, **A**, and a camera measures the intensity, $|\mathbf{b}|^2$.

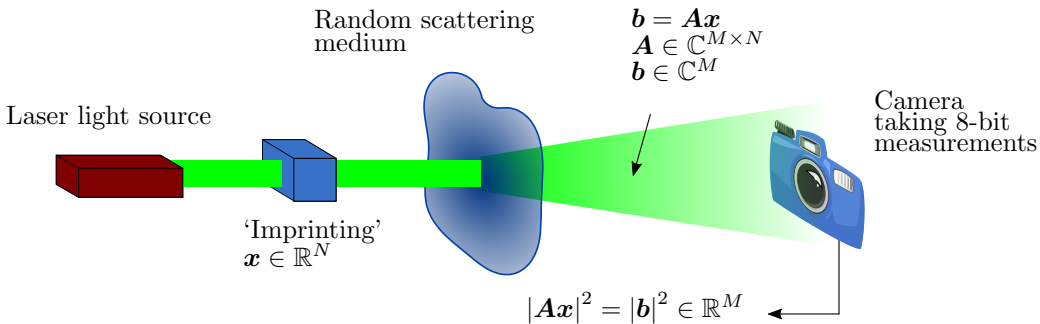


Figure 1.2: The operation of an OPU. A data signal is ‘imprinted’ on laser light and shined through a random scattering medium whose TM is iid standard complex Gaussian. Therefore, the output optical field is a random projection of the data signal. The output field’s intensity is captured by a camera. We wish to recover the random projection, **b**, from its intensity measurements.

While the elements of the TM are unknown, the random scattering medium has been designed so that the TM’s elements are independent and identically distributed (iid) standard complex Gaussian. Therefore, the output optical field **b** is a random projection of **x**. Such optics-based matrix multiplication

¹Commercially available OPUs use this setup (<https://lighton.ai/>).

can be much faster and energy-efficient than their CPU or GPU counterparts at large scales. However, to realize this, we need to retrieve the random projection, \mathbf{b} , from its intensity measurements, $|\mathbf{b}|^2$. This connects to **E1**. Improving the speed and scale of matrix multiplications is an ongoing area of research and recently received attention with DeepMind’s work on using machine learning to discover new algorithms for fundamental tasks [11].²

Physical explanation. We briefly explain the phase problem modeling in Figures 1.1 and 1.2. To begin, optical waves are physical quantities with amplitude and phase. As they have amplitude and phase, it is standard to model them as complex-valued quantities for convenience [12, 13, 14]. Importantly, the results from the complex-valued mathematical models that are used in optics and the quantities that we can experimentally measure, such as field intensities, are consistent with each other.

We model scattering media as matrices due to linear optics theory which shows that a complex-valued optical field that propagates through such media is linearly related to the input field. Well-known linear propagation examples are propagation through free-space and using lenses to perform Fourier transforms [12, 1]. The scattering media used in this thesis are the concatenation of several thin optically linear slices. Therefore, the entire scattering medium propagates the field linearly [15, 14].

The cameras in Figures 1.1 and 1.2 measure electrical current which is proportional to the optical intensity, I , that is incident on the camera’s semiconductor detector [12]. To compute I , we switch to the electromagnetic (EM) waves description of the incident optical waves so that Maxwell’s EM equations can be used. Then, with commonly used assumptions, I is proportional to the squared norm of the EM wave’s electric field amplitude vector, $\|\mathbf{E}\|_2^2$. Furthermore, due to the assumptions, these EM waves have associated optical waves with amplitude whose magnitude is proportional to $\|\mathbf{E}\|_2$ [13]. Hence, the cameras measure the squared-magnitude of the optical field.

²DeepMind research blog on machine learning for algorithm discovery: <https://www.deepmind.com/blog/discovering-novel-algorithms-with-alphatensor>

1.1.2 Linear imaging problems

A broad range of imaging problems can be modeled as linear inverse problems. This means that a linear transform of the object being imaged gives the measurements. If we denote the linear operator as the matrix $\mathbf{A} \in \mathbb{R}^{M \times N}$, the vectorized object whose image we want as $\mathbf{x} \in \mathbb{R}^N$, and the measurements as $\mathbf{y} \in \mathbb{R}^M$, linear problems can be framed using Equation 1.1 as

$$\mathbf{y} = \mathbf{A}\mathbf{x} \tag{1.5}$$

and the solution using (1.2) is $\hat{\mathbf{x}} = \mathcal{S}(\mathbf{y}, \mathbf{A})$.

Well-known linear imaging applications that fit Equation 1.5 are x-ray computed tomography (CT) [1], image deblurring [4] and compressed sensing imaging [16]. Imaging problems with nonlinear forward processes such as seismic travelttime tomography and photoacoustic tomography may also be linearized and modeled as linear problems [17, 18]. This can simplify computing the forward process and allow us to use linear algebra for analysis.

Typically when $M > N$, it is straightforward to obtain an accurate solution to the inverse problem. However, many imaging problems have $M \ll N$. For example, in seismic tomography, it is not practical to densely cover earth surfaces with sensors and so we receive few measurements. And, in x-ray CT imaging, we try to acquire fewer measurements in order to reduce a patient's radiation exposure. Consequently, \mathbf{A} has a possibly large null space and so there are many solutions, $\hat{\mathbf{x}}$, which give the same measurements, \mathbf{y} . This is known as an ill-posed inverse problem, and regularization strategies are used [6]. In Chapter 5, we explore regularization through projections (**E1**). We might also face issues due to \mathbf{A} not accurately modeling the true forward process due to calibration errors [19]. We design methods to account for this, and this work falls under **E2**.

1.2 Thesis organization and contributions

This thesis explores the essential imaging elements through **E1** and **E2**. Chapters 2, 3 and 4 look at quadratic phase problems, and Chapters 5 and 6 are general inverse problem methods. The effectiveness of all techniques developed in this thesis is demonstrated through numerical simulations. In

addition to this, we verify the applicability of all phase problems algorithms on real optical hardware.

Chapter 2 Measurement phase retrieval (E1): We tackle the problem of recovering the phase of complex linear measurements when only magnitude information is available and we control the input. This is motivated by the development of OPUs as explained in Figure 1.2. We show that even without knowing the TM, we can recover the unknown measurement phase and, hence, recover random projections from their magnitudes. Our proposed method casts measurement phase retrieval as a Euclidean distance geometry problem.

Chapter 3 Rapid TM calibration (E1 and E2): We propose a numerical interferometry method to calibrate unknown TMs when only intensity can be measured, as in Figure 1.1. Our method simplifies TM calibration from a quadratic to a linear problem by first recovering the phase of the measurements. We perform measurement phase retrieval by designing a rapid version of the method developed in Chapter 2. Where the previous state-of-the-art method reports hours to calibrate a TM, our method takes only a few minutes.

Chapter 4 Total least squares phase retrieval (E2): We address problems such as imaging through scattering media (Figure 1.1) when there are unknown TM errors. For example, the resulting TM from the calibration process in Chapter 3 may have errors. Most image recovery methods are based on least squares (LS) formulations which only assume errors in the measurements. We extend this approach to also handle TM errors by adopting the total least squares (TLS) framework that is used in linear inverse problems with operator errors. We show how the specific geometry of the phase retrieval problem can be used to obtain an efficient TLS solution. Additionally, we derive expressions for the solution error.

Chapter 5 Random mesh projectors (E1): We propose a deep learning-based approach to solve imaging problems when ground truth training samples are unavailable and we can only get few measurements. We show that the typical approach to learn a mapping from the measured data to the

solution is unstable. Instead, we first learn an ensemble of mappings from the measurements to random projections of the unknown image. The projections are then treated as measurements for a reformulated inverse problem whose solution is our final reconstruction.

Chapter 6 Differentiable uncalibrated imaging (E1 and E2): We solve imaging problems when the measurement coordinates such as sensor locations and projection angles are uncertain. From the observed discrete measurements, we learn a continuous representation of the measurements which is differentiable with respect to measurement coordinates. We use popular implicit neural networks and also develop differentiable splines as measurement representations. Differentiability allows us to learn the measurement representation and optimize over the uncertain measurement coordinates at the same time. To ensure consistency, the optimization is done jointly with reconstructing the final image.

Chapter 7 Looking forward: We summarize the key developments in this thesis and discuss future research opportunities that this thesis motivates.

CHAPTER 2

MEASUREMENT PHASE RETRIEVAL

In this chapter, we recover the phase of complex linear measurements from their intensity measurements when we control the input. We are motivated to do this by the development of OPUs to optically compute random projections as shown in Figure 1.2. This work falls under **E1** as we recover the phase information of phaseless measurements.

2.1 Phasing optical random projections

Random projections are at the heart of many algorithms in machine learning, signal processing, and numerical linear algebra. Recent developments ranging from classification with random features [21], kernel approximation [22], and sketching for matrix optimization [23, 24], to sublinear-complexity transforms [25] and randomized linear algebra are all enabled by random projections. Computing random projections for realistic signals such as images, videos, and modern big data streams is computation and memory intensive. Thus, from a practical point of view, any increase in the size and speed at which one can do the required processing is highly desirable.

This fact has motivated work on using dedicated hardware based on physics rather than traditional CPU and GPU computation to obtain random projections. As mentioned in Chapter 1 (Figure 1.2), a notable example is the scattering of light in random media (Figure 2.1 (left)) with an optical processing unit (OPU). The OPU enables rapid (20 kHz) projections of high-dimensional data such as images, with input dimension scaling up to one million and output dimension also in the million range. It works by “im-

This work was previously published in S. Gupta, R. Gribonval, L. Daudet, and I. Dokmanić, “Don’t take it lightly: Phasing optical random projections with unknown operators,” in *Advances in Neural Information Processing Systems*, vol. 32, 2019 [20] and is adapted here with permission.

printing” an input data signal, $\boldsymbol{\xi} \in \mathbb{R}^N$, onto a coherent light beam using a digital micro-mirror device (DMD) and shining the modulated light through a multiple scattering medium such as titanium dioxide white paint. The scattered lightfield in the sensor plane can then be written as

$$\mathbf{y} = \mathbf{A}\boldsymbol{\xi} \quad (2.1)$$

where $\mathbf{A} \in \mathbb{C}^{M \times N}$ is the transmission matrix of the random medium with desirable properties.

One of the major challenges associated with this approach is that \mathbf{A} is in general *unknown*. Though it could, in principle, be learned via calibration [26], such a procedure is slow and inconvenient, especially at large dimensions. On the other hand, the system can be designed so that the distribution of \mathbf{A} is approximately iid standard complex Gaussian. Luckily, this fact alone is sufficient for many algorithms and the actual values of \mathbf{A} are not required.

Another challenge is that common camera sensors are only sensitive to intensity, so we can only measure the intensity of scattered light, $|\mathbf{y}|^2$, where $|\cdot|$ is the elementwise absolute value. The phase information is thus lost. While the use of interferometric measurements with a reference could enable estimating the phase, the practical setup is more complex, sensitive, and does not share the convenience and simplicity of the setup illustrated in Figure 2.1 (left).

This motivates us to consider the *measurement phase retrieval (MPR) problem*. The measured sensor data is modeled as

$$\mathbf{b} = |\mathbf{y}|^2 + \boldsymbol{\eta} = |\mathbf{A}\boldsymbol{\xi}|^2 + \boldsymbol{\eta}, \quad (2.2)$$

where $\mathbf{b} \in \mathbb{R}^M$, $\boldsymbol{\xi} \in \mathbb{R}^N$, $\mathbf{A} \in \mathbb{C}^{M \times N}$, $\mathbf{y} \in \mathbb{C}^M$, and $\boldsymbol{\eta} \in \mathbb{R}^M$ is noise. The goal is to recover the phase of each complex-valued element of \mathbf{y} , y_i for $1 \leq i \leq M$, from its magnitude measurements \mathbf{b} when $\boldsymbol{\xi}$ is *known* and the entries of \mathbf{A} are *unknown*. The classical phase retrieval problem which has received much attention over the last decade [27, 28] has the same quadratic form as (2.2) but with a known \mathbf{A} and the task being to recover $\boldsymbol{\xi}$ instead of \mathbf{y} . While at a glance it might seem that not knowing \mathbf{A} precludes computing the phase of $\mathbf{A}\boldsymbol{\xi}$, we show in this chapter that it is in fact possible via an exercise in distance geometry.

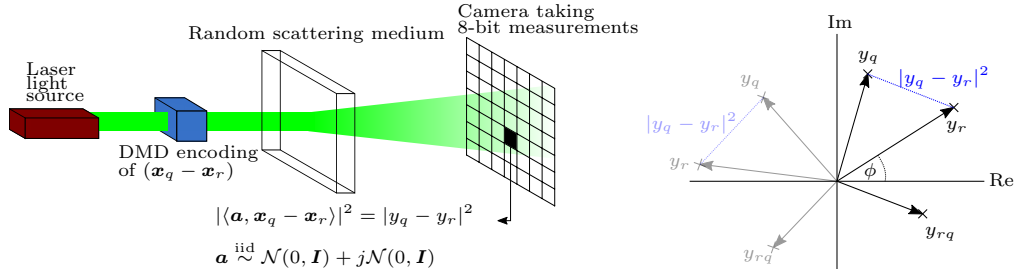


Figure 2.1: Left: The optical processing unit (OPU) is an example application of where the MPR problem appears. A coherent laser beam spatially encodes a signal $(\mathbf{x}_q - \mathbf{x}_r)$ via a digital micro-mirror device (DMD) which is then shined through a random medium. A camera measures the squared magnitude of the scattered light, which is equivalent to the Euclidean distance between complex numbers $y_q \in \mathbb{C}$ and $y_r \in \mathbb{C}$. Furthermore the camera takes quantized measurements; Right: y_q and y_r are points on the two-dimensional complex plane. We can optically measure the squared Euclidean distance between points and use these distances to localize points on the complex plane and obtain their phase. Note that transformations such as rotations and reflections do not change the distances.

The noise $\boldsymbol{\eta}$ is primarily due to quantization because standard camera sensors measure low precision values—8-bit in our case (integers between 0 and 255 inclusive). Furthermore, cameras may perform poorly at low intensities. This is another data-dependent noise source which is modelled in (2.3) by a binary mask vector $\mathbf{w} \in \mathbb{R}^M$ which is zero when the intensity is below some threshold and one otherwise; \odot denotes the elementwise product.

$$\mathbf{b} = \mathbf{w} \odot (|\mathbf{y}|^2 + \boldsymbol{\eta}) = \mathbf{w} \odot (|\mathbf{A}\boldsymbol{\xi}|^2 + \boldsymbol{\eta}) \quad (2.3)$$

The distribution of \mathbf{A} follows from the properties of random scattering media [29, 26]. It has iid standard complex Gaussian entries, $a_{mn} \sim \mathcal{N}(0, 1) + j\mathcal{N}(0, 1)$ for all $1 \leq m, n \leq M, N$.

The usefulness of phase is obvious. While, in some applications, having only the magnitude of the random projection is enough (see [30] for an example related to elliptic kernels), most applications require the phase. For example, with the phase one can implement a more diverse range of kernels as well as randomized linear algebra routines like randomized singular value decomposition (SVD). We report the results of the latter on a real hardware OPU in Section 2.3.1.

2.1.1 Chapter overview and contributions

We develop an algorithm based on distance geometry to solve the MPR problem (2.2). We exploit the fact that we control the input to the system, which allows us to mix ξ with arbitrary reference inputs. By interpreting each measurement as the magnitude of a point in the complex plane, this leads to a formulation of the MPR problem as a pure distance geometry problem (see Section 2.2.2 and Figure 2.1 (right)). With enough pairwise distances (corresponding to reference signals), we can localize the points on the complex plane via a variant of multidimensional scaling (MDS) [31, 32], and thus compute the missing phase.

As we demonstrate, the proposed algorithm not only accurately recovers the phase, but also improves the number of useful bits of the magnitude information thanks to the multiple views. Established Euclidean distance geometry bounds imply that even with many distances below the sensitivity threshold and coarse quantization, the proposed algorithm allows for accurate recovery. This fact, which we verify experimentally, could have bearing on the design of future random projectors by navigating the tradeoff between physics and computation.

2.1.2 Related work

The classical phase retrieval problem looks at the case where \mathbf{A} is known and ξ has to be recovered from \mathbf{b} in (2.2) [33, 7, 34]. A modified version of the classical problem known as holographic phase retrieval is related to our approach: a known reference signal is concatenated with ξ to facilitate the estimation of ξ [35]. Interference with known references for classical phase retrieval has also been studied for known (Fourier) operators [36, 37].

An optical random projection setup similar to the one we consider has been used for kernel-based classification [30], albeit using only magnitudes. A phaseless approach to classification with the measured magnitudes fed into a convolutional neural network was reported by Satat et al. [38].

An alternative to obtaining the measurement phase is to measure, or calibrate, the unknown transmission matrix \mathbf{A} from the measured intensities. This has been attempted in compressive imaging applications, but the process is impractical at even moderate pixel counts [26, 29]. Estimating \mathbf{A} can

take days and even the latest GPU-accelerated methods take hours for moderately sized \mathbf{A} [39]. Due to this, other approaches forego calibration and use the measured magnitudes to learn an inverse map of $\mathbf{x} \mapsto |\mathbf{Ax}|^2$ [40]. On the other hand, in Chapter 3 of this thesis, we will explore how retrieving the measurement phase reduces calibrating \mathbf{A} to a simple linear system.

Leaving hardware approaches aside, there have been multiple algorithmic efforts to improve the speed of random projections [41, 22] for machine learning and signal processing tasks. Still, efficiently handling high-dimensional input remains an ongoing challenge.

2.2 The measurement phase retrieval problem

We will denote the signal of interest by $\boldsymbol{\xi} \in \mathbb{R}^N$, and the K reference *anchor* signals by $\mathbf{r}_k \in \mathbb{R}^N$ for $1 \leq k \leq K$. To present the full algorithm we will need to use multiple signals of interest which we will then denote $\boldsymbol{\xi}_1, \dots, \boldsymbol{\xi}_S$; each $\boldsymbol{\xi}_s$ is called a frame. We set the last, K th anchor to be the origin, $\mathbf{r}_K = \mathbf{0}$. We ascribe $\boldsymbol{\xi}$ and the anchors to the columns of the matrix $\mathbf{X} \in \mathbb{R}^{N \times Q}$, so that $\mathbf{X} = [\boldsymbol{\xi}, \mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_K]$ and have $Q = K + 1$. The q th column of \mathbf{X} is denoted \mathbf{x}_q . For any $1 \leq q, r \leq Q$, we let $\mathbf{y}_q = \mathbf{Ax}_q$ and $\mathbf{y}_{qr} := \mathbf{A}(\mathbf{x}_q - \mathbf{x}_r)$, with $y_{qr,m}$ being its m th entry. Finally, the m th row of \mathbf{A} will be denoted by \mathbf{a}^m so that $y_{qr,m} = \langle \mathbf{a}^m, \mathbf{x}_q - \mathbf{x}_r \rangle$.

2.2.1 Problem statement and recovery up to a reference phase and conjugation

Since we do not know \mathbf{A} , it is clear that recovering the absolute phase of \mathbf{Ax} is impossible. On the other hand, many algorithms do not require any knowledge of \mathbf{A} except that it is iid standard complex Gaussian, and that it does not change throughout the computations.

Let \mathbf{R} be an operator which adds a constant phase to each row of its argument (multiplies it by $\text{diag}(e^{j\phi_1}, \dots, e^{j\phi_m})$ for some ϕ_1, \dots, ϕ_m) and conjugates a subset of its rows. Since a standard complex Gaussian is circularly symmetric, $\mathbf{R}(\mathbf{A})$ has the same distribution as \mathbf{A} . Therefore, since we do not know \mathbf{A} , it does not matter whether we work with \mathbf{A} itself or with $\mathbf{R}(\mathbf{A})$ for some possibly unknown \mathbf{R} . As long as the same *effective* \mathbf{R} is used for all

inputs during algorithm operation, the relative phases between the frames will be the same whether we use $\mathbf{R}(\mathbf{A})$ or \mathbf{A} .¹

Problem 1. *Given a collection of input frames $\boldsymbol{\xi}_1, \dots, \boldsymbol{\xi}_S$ to be randomly projected and the device illustrated in Figure 2.1 (left) with an unknown transmission matrix $\mathbf{A} \in \mathbb{C}^{M \times N}$ and a b -bit camera, compute the estimates of projections $\hat{\mathbf{y}}_1, \dots, \hat{\mathbf{y}}_S$ up to a global row-wise phase and conjugation; that is, so that there exists some \mathbf{R} such that $\hat{\mathbf{y}}_s \approx \mathbf{R}(\mathbf{y}_s)$ for all $1 \leq s \leq S$.*

2.2.2 MPR as a distance geometry problem

Since the rows of \mathbf{A} are statistically independent, we can explain our algorithm for a single row and then repeat the same steps for the remaining rows. We will therefore omit the row subscript/superscript m except where explicitly necessary.

Instead of randomly projecting $\boldsymbol{\xi}$ and measuring the corresponding projection magnitude $|\mathbf{A}\boldsymbol{\xi}|^2$, consider randomly projecting the difference between $\boldsymbol{\xi}$ and some reference vector, or more generally a difference between two columns in \mathbf{X} , thus measuring $|\langle \mathbf{a}, \mathbf{x}_q - \mathbf{x}_r \rangle|^2 = |y_q - y_r|^2$. Interpreting y_q and y_r as points in the complex plane, we see that the camera sensor measures exactly the squared Euclidean distance between them. Since we control the input to the OPU, we can indeed set it to $\mathbf{x}_q - \mathbf{x}_r$ and measure $|y_q - y_r|^2$ for all $1 \leq q, r \leq Q$.

This is the key point: as we can measure pairwise distances between a collection of two-dimensional vectors in the two-dimensional complex plane, we can use established distance geometry algorithms such as multidimensional scaling (MDS) to localize points and get their phase. This is illustrated in Figure 2.1 (right). The same figure also illustrates the well known fact that rigid transformations of a point set cannot be recovered from distance data. We need to worry about three things: translations, reflections, and rotations.

The translation ambiguity can be easily dealt with if one notes that for any column \mathbf{x}_q of \mathbf{X} , $|y_q| = |\langle \mathbf{a}, \mathbf{x}_q \rangle|$ gives us the distance of y_q to the origin which is a fixed point, ultimately resolving the translation ambiguity. There is, however, no similar simple way to do away with the rotation and reflection ambiguity, so it might seem that there is no way to uniquely determine the

¹Up to a sign.

phase of $\langle \mathbf{a}, \boldsymbol{\xi} \rangle$. This is where the discussion from the preceding subsection comes to the rescue. Since \mathbf{R} is arbitrary, as long as it is kept fixed for all the frames, we can arbitrarily set the orientation of any given frame and use it as a reference, making sure that the relative phases are computed correctly.

2.2.3 Proposed algorithm

As defined previously, the columns of $\mathbf{X} \in \mathbb{R}^{N \times Q}$ list the signal of interest and the anchors. Recall that all the entries of \mathbf{X} are known. Using the OPU, we can compute a noisy (quantized) version of

$$|y_{qr}|^2 = |\langle \mathbf{a}, \mathbf{x}_q - \mathbf{x}_r \rangle|^2 = |y_q - y_r|^2, \quad (2.4)$$

for all (q, r) , which gives us $Q(Q - 1)/2$ squared Euclidean distances between points $\{y_q \in \mathbb{C}\}_{q=1}^Q$ on the complex plane. These distances can be used to populate a Euclidean (squared) distance matrix $\mathbf{D} \in \mathbb{R}^{Q \times Q}$ as $\mathbf{D} = (d_{qr}^2)_{q,r=1}^Q = (|y_{qr}|^2)_{q,r=1}^Q$, which we will use to localize all complex points, y_q .

We start by defining the matrix of all the complex points in \mathbb{R}^2 which we want to recover as

$$\boldsymbol{\Upsilon} = \begin{bmatrix} \text{Re}(y_1) & \text{Re}(y_2) & \cdots & \text{Re}(y_Q) \\ \text{Im}(y_1) & \text{Im}(y_2) & \cdots & \text{Im}(y_Q) \end{bmatrix} \in \mathbb{R}^{2 \times Q}.$$

Denoting the q th column of $\boldsymbol{\Upsilon}$ by \mathbf{v}_q , we have $d_{qr}^2 = \|\mathbf{v}_q - \mathbf{v}_r\|_2^2 = \mathbf{v}_q^T \mathbf{v}_q - 2\mathbf{v}_q^T \mathbf{v}_r + \mathbf{v}_r^T \mathbf{v}_r$ so that

$$\mathbf{D} = \text{diag}(\mathbf{G}) \mathbf{1}_Q^T - 2\mathbf{G} + \mathbf{1}_Q \text{diag}(\mathbf{G})^T =: \mathcal{K}(\mathbf{G}), \quad (2.5)$$

where $\text{diag}(\mathbf{G}) \in \mathbb{R}^Q$ is the column vector of the diagonal entries in the Gram matrix $\mathbf{G} := \boldsymbol{\Upsilon}^T \boldsymbol{\Upsilon} \in \mathbb{R}^{Q \times Q}$ and $\mathbf{1}_Q \in \mathbb{R}^Q$ is the column vector of Q ones. This establishes a relationship between the measured distances in \mathbf{D} and the locations of the complex points in \mathbb{R}^2 which we seek. We denote by \mathbf{J} the geometric centering matrix, $\mathbf{J} := \mathbf{I} - \frac{1}{Q} \mathbf{1}_Q \mathbf{1}_Q^T$ so that

$$\widehat{\mathbf{G}} = -\frac{1}{2} \mathbf{J} \mathbf{D} \mathbf{J} = \mathbf{J} \mathbf{G} \mathbf{J} = (\boldsymbol{\Upsilon} \mathbf{J})^T (\boldsymbol{\Upsilon} \mathbf{J}) \quad (2.6)$$

is the Gram matrix of the centered point set in terms of Υ . Matrices $\widehat{\mathbf{G}}$ and \mathbf{J} are known as the Gram matrix of the centered point set and the geometric centering matrix because $\Upsilon\mathbf{J}$ is the points in Υ with their mean subtracted. An estimate $\widehat{\Upsilon}$ of the centered point set, $\Upsilon\mathbf{J}$, is then obtained by eigendecomposition as $\widehat{\mathbf{G}} = \mathbf{V} \text{diag}(\lambda_1, \dots, \lambda_Q)\mathbf{V}^T$ and taking $\widehat{\Upsilon} = [\sqrt{\lambda_1}\mathbf{v}_1, \sqrt{\lambda_2}\mathbf{v}_2]^T$ where \mathbf{v}_1 and \mathbf{v}_2 are the first and second columns of \mathbf{V} and assuming that the eigenvalue sequence is nonincreasing. This process is the classical MDS algorithm [31, 32]. Finally, the phases can be calculated via a four-quadrant inverse tangent, $\phi(y_q) = \arctan(v_{q2}, v_{q1})$.

Procrustes analysis. As we recovered a centered point set via MDS with a geometric centering matrix \mathbf{J} , the point set will have its centroid at the origin. This is a consequence of the used algorithm, and not the “true” origin. As described above, we know that $|y_q|^2$ defines squared distances to the origin and $y_Q = \langle \mathbf{a}, \mathbf{x}_Q \rangle = 0 + 0j$ (as \mathbf{x}_Q was set to the origin). This means that we can correctly center the recovered points by translating the centered point set, $\widehat{\Upsilon}$, by $-\mathbf{v}_Q$.

The correct absolute rotation and reflection cannot be recovered. However, since we only care about working with some effective $\mathbf{R}(\mathbf{A})$ with the correct distribution, we only need to ensure that the relative phases between the frames are correct. We can thus designate the first frame as the reference frame. This sets the rotation (which directly corresponds to the phase) and reflection (corresponding to conjugation) arbitrarily. Once these are chosen, the location of the anchors $\mathbf{r}_1, \dots, \mathbf{r}_K$ are fixed, which in turn fixes the phasing-conjugation operator \mathbf{R} .

Since \mathbf{A} is unknown, \mathbf{R} is also unknown, but fixed anchors allow us to compute the correct relative phase with respect to $\mathbf{R}(\mathbf{A})$ for the subsequent frames. Namely, upon receiving a new input $\boldsymbol{\xi}_s$ to be randomly projected, we now localize it with respect to a fixed set of anchors. This is achieved by Procrustes analysis. Denoting by $\widetilde{\Upsilon}_1$ our reference estimate of the anchor positions in frame 1 (columns 2, \dots , Q of $\widehat{\Upsilon}$ above which was recovered from $\widehat{\mathbf{G}}$ in (2.6)), and by $\widetilde{\Upsilon}_s$ the MDS estimate of anchor positions in frame s , adequately centered. Let $\widetilde{\Upsilon}_s\widetilde{\Upsilon}_1^T = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$ be the singular value decomposition of $\widetilde{\Upsilon}_s\widetilde{\Upsilon}_1^T$. The optimal transformation matrix in the least squares sense is then $\mathbf{R} = \mathbf{V}\mathbf{U}^T$ so that $\mathbf{R}\widetilde{\Upsilon}_s \approx \widetilde{\Upsilon}_1$ [42].

Finally, we note that with a good estimate of the anchors, one can imagine

not relocalizing them in every frame. The localization problem for $\boldsymbol{\xi}$ then boils down to multilateration, cf. A.3 in the Appendix.

2.2.4 Sensitivity threshold and missing measurements

As we further elaborate in Appendix A.1, in practice, some measurements fall below the sensitivity threshold of the camera and produce spurious values. A nice benefit of multiple “views” of $\boldsymbol{\xi}$ via its interaction with reference signals is that we can ignore those measurements. This introduces missing values in \mathbf{D} which can be modeled via a binary mask matrix \mathbf{W} . The recovery problem can be modeled as estimating $\boldsymbol{\Upsilon}$ from $\mathbf{W} \odot (\mathbf{D} + \mathbf{E})$ where $\mathbf{W} \in \mathbb{R}^{N \times N}$ contains zeros for the entries which fall below some prescribed threshold, and ones otherwise. Matrix \mathbf{E} models quantization noise.

We can predict the performance of the proposed method when modeling the entries of \mathbf{W} as iid Bernoulli random variables with parameter p , where $1-p$ is the probability that an entry falls below the sensitivity threshold. We model \mathbf{E} as uniform quantization noise distributed as $\mathcal{U}\left(-\frac{\kappa}{2(2^b-1)}, \frac{\kappa}{2(2^b-1)}\right)$, where b is the number of bits, and κ an upper bound on the entries of \mathbf{D} (in our case $2^8 - 1 = 255$).

Adapting existing results on the performance of multidimensional scaling [43] (by noting that \mathbf{E} is sub-Gaussian), we can get the following scaling of the distance recovery error with K anchors (for K sufficiently large),

$$\frac{1}{K} \mathbb{E} \left[\left\| \hat{\mathbf{D}} - \mathbf{D} \right\|_F \right] \lesssim \frac{\kappa}{\sqrt{pK}}, \quad (2.7)$$

where \lesssim denotes inequality up to a constant which depends on the number of bits b , the sub-Gaussian norm of the entries in \mathbf{E} , and the dimension of the ambient space (here \mathbb{R}^2). We use $\hat{\mathbf{D}}$ to denote an estimate of the masked Euclidean distance matrix [43]. An important implication is that even for coarse quantization (small b) and for a large fraction of entries below the sensitivity threshold (small p), we can achieve arbitrarily small amplitude and phase errors *per point* by increasing the number of reference signals K .

Refinement with gradient descent. The output of the classical MDS method described above can be further refined via a local search. A standard

Algorithm 1 MPR algorithm for S frames.

Input: Squared distances $[|y_{j,m} - y_{l,m}|^2]_s$ for all $1 \leq j, l \leq Q$ for frames $1 \leq s \leq S$ and rows $1 \leq m \leq M$; $[\cdot]_s$ denotes frame s

Output: $\mathbf{Y} \in \mathbb{C}^{M \times S}$ containing all localized points such that $\mathbf{y}_s = \mathbf{R}(\mathbf{A})\boldsymbol{\xi}_s$ for some fixed \mathbf{R} .

- 1: $\mathbf{Y} \leftarrow \mathbf{0}_{M \times S}$ ▷ Initialize \mathbf{Y}
- 2: $m \leftarrow 1$
- 3: **while** $m \leq M$ **do** ▷ Solve each row separately
- 4: Populate all frame $s = 1$ distances into distance matrix \mathbf{D} ▷
 $\mathbf{D} \in \mathbb{R}^{Q \times Q}$
- 5: $[\boldsymbol{\Upsilon}]_1 \leftarrow \text{MDS}(\mathbf{D})$ ▷ $[\boldsymbol{\Upsilon}]_1 \in \mathbb{R}^{2 \times Q}$
- 6: $[\boldsymbol{\Upsilon}]_1 \leftarrow \text{GradientDescent}(\mathbf{D}, [\boldsymbol{\Upsilon}]_1)$
- 7: $[\boldsymbol{\Upsilon}]_1 \leftarrow [\boldsymbol{\Upsilon}]_1 - [\mathbf{v}_Q]_1 \mathbf{1}^T$ ▷ Translate to align with origin
- 8: $s \leftarrow 2$
- 9: **while** $s \leq S$ **do**
- 10: Populate all frame s distances into distance matrix \mathbf{D}
- 11: $[\boldsymbol{\Upsilon}]_s \leftarrow \text{MDS}(\mathbf{D})$
- 12: $[\boldsymbol{\Upsilon}]_s \leftarrow \text{GradientDescent}(\mathbf{D}, [\boldsymbol{\Upsilon}]_s)$
- 13: $[\boldsymbol{\Upsilon}]_s \leftarrow [\boldsymbol{\Upsilon}]_s - [\mathbf{v}_Q]_s \mathbf{1}^T$
- 14: $\mathbf{R} \leftarrow \text{Procrustes}([\mathbf{v}_2, \dots, \mathbf{v}_Q]_1, [\mathbf{v}_2, \dots, \mathbf{v}_Q]_s)$ ▷ \mathbf{R} aligns frames
1 and s anchors
- 15: $[\boldsymbol{\Upsilon}]_s \leftarrow \text{Align}([\boldsymbol{\Upsilon}]_s, \mathbf{R}, [\mathbf{v}_2, \dots, \mathbf{v}_Q]_1)$ ▷ Align anchors
- 16: $s \leftarrow s + 1$
- 17: **end while**
- 18: $\mathbf{U} \leftarrow [[\mathbf{v}_1]_1, [\mathbf{v}_1]_2, \dots, [\mathbf{v}_1]_S]$ ▷ $\mathbf{U} \in \mathbb{R}^{2 \times S}$
- 19: $\mathbf{y}^m \leftarrow \mathbf{u}^1 + j\mathbf{u}^2$ ▷ Multiply second row of \mathbf{U} with j and add to first
row
- 20: $m \leftarrow m + 1$
- 21: **end while**

differentiable objective called the squared stress is defined as follows,

$$\min_{\mathbf{Z}} f(\boldsymbol{\Upsilon}) = \min_{\mathbf{Z}} \|\mathbf{W} \odot (\mathbf{D} - \mathcal{K}(\mathbf{Z}^T \mathbf{Z}))\|_F^2, \quad (2.8)$$

where $\mathcal{K}(\cdot)$ is as defined in (2.5) and $\mathbf{Z} \in \mathbb{R}^{2 \times Q}$ is the point matrix induced by row m of \mathbf{A} . In our experiments we report the result of refining the classical MDS results via gradient descent on (2.8).

Note that the optimization (2.8) is nonconvex. The complete procedure is thus analogous to the usual approach to nonconvex phase retrieval by spectral initialization followed by gradient descent [27, 28]. Algorithm 1 summarizes our proposed method.

2.3 Experimental verification and application

We test the proposed MPR algorithm via simulations and experiments on a real OPU. For hardware experiments, we use a scikit-learn interface to a publicly available cloud-based OPU.²

Evaluation metrics. The main challenge is to evaluate the performance without knowing the transmission matrix \mathbf{A} . To this end, we propose to use the *linearity error*. The rationale behind this metric is that with the phase correctly recovered, the end-to-end system should be linear. That is, if we recover \mathbf{y} and \mathbf{z} from $|\mathbf{y}|^2 = |\mathbf{A}\boldsymbol{\xi}_1|^2$ and $|\mathbf{z}|^2 = |\mathbf{A}\boldsymbol{\xi}_2|^2$, then we should get $(\mathbf{y} + \mathbf{z})$ when applying the method to $|\mathbf{v}|^2 = |\mathbf{A}(\boldsymbol{\xi}_1 + \boldsymbol{\xi}_2)|^2$. With this notation, the relative linearity error is defined as

$$\text{linearity error} = \frac{1}{M} \sum_{m=1}^M \frac{|(y_m + z_m) - v_m|}{|v_m|}. \quad (2.9)$$

The second metric we use is the number of “good” or correct bits. This metric can only be evaluated in simulation since it requires the knowledge of the ground truth measurements. Letting $|y|^2 = |\langle \mathbf{a}, \boldsymbol{\xi} \rangle|^2$ and \hat{y} be our estimate of y , the number of good bits is defined as

$$\text{good bits} = -\frac{20}{6.02} \log \left(\frac{||y|^2 - |\hat{y}|^2|}{|y|^2} \right). \quad (2.10)$$

It is proportional to the signal-to-quantization-noise ratio if the distances uniformly cover all quantization levels.³

2.3.1 Experiments

In all simulations, intensity measurements are quantized to 8 bits, and all signals and references are iid standard (complex) Gaussian random vectors.

We first test the phase recovery performance by evaluating the linearity error. In simulation, we draw random frames $\boldsymbol{\xi}_1$, $\boldsymbol{\xi}_2$, and $\mathbf{A} \in \mathbb{C}^{100 \times 64^2}$. We

²<https://www.lighton.ai/lighton-cloud/>.
Reproducible code available at https://github.com/swing-research/opu_phase under the MIT License.

³Note that the quantity registered by the camera is actually the squared magnitude, hence the factor 20.

apply Algorithm 1 to $|\mathbf{A}\boldsymbol{\xi}_1|^2$, $|\mathbf{A}\boldsymbol{\xi}_2|^2$ and $|\mathbf{A}(\boldsymbol{\xi}_1 + \boldsymbol{\xi}_2)|^2$ and calculate the linearity error (2.9). We use classical MDS and MDS with gradient descent (MDS-GD). Figure 2.2a shows that the system is, indeed, approximately linear and that the linearity error becomes smaller as the number of reference signals grows. In Figure 2.2b, we set the sensitivity threshold to $\tau = 6$ and zero the distances below the threshold per (2.3). Again, the linearity error quickly becomes small as the number of anchors increases, showing that the overall system is robust and that it allows recovery of phase for small-intensity signals.

Next, we test the linearity error with a real hardware OPU. The OPU gives 8-bit unsigned integer measurements. A major challenge is that the DMD (see Figure 2.1) only allows binary input signals. This is a property of the particular OPU we use and, while it imposes restrictions on reference design, the method is unchanged as our algorithm does not assume a particular type of signal. Appendix A.1 describes how we create binary references and addresses other hardware-related practicalities.

Figure 2.2c reports the linearity error on the OPU with suitably designed references and the same size \mathbf{A} . The empirically determined sensitivity threshold of the camera is $\tau = 6$, and the measurements below the threshold were not used. We ignore rows of \mathbf{A} which give points with small norms (less than two) because they are prone to noise and disproportionately influence the relative error. Once again, we observe that the end-to-end system with Algorithm 1 is approximately linear and that the linearity improves as we increase the number of anchors.

Finally, we demonstrate the magnitude denoising performance. We draw $\mathbf{a} \in \mathbb{C}^{100}$, a random signal $\boldsymbol{\xi} \in \mathbb{R}^{100}$ and a set of random reference anchor signals. We run our algorithm for number of anchors varying between two and 15. For each number of anchors, we recover \hat{y} for $|y|^2 = |\langle \mathbf{a}, \boldsymbol{\xi} \rangle|^2$ using either classical MDS or MDS-GD. We then measure the number of good bits. The average results over 100 trials are shown in Figure 2.3a. Figure 2.3b reports the same experiment with the sensitivity threshold set to $\tau = 6$ (that is, the entries below τ are zeroed in the distance matrix per (2.3)). Both Figures show that the proposed algorithm significantly improves the estimated magnitudes in addition to recovering the phases. The approximately 1 additional good bit with gradient descent in Figure 2.3b corresponds to the relative value of $2^1/2^8 \approx 0.8\%$ which is consistent with the gradient descent

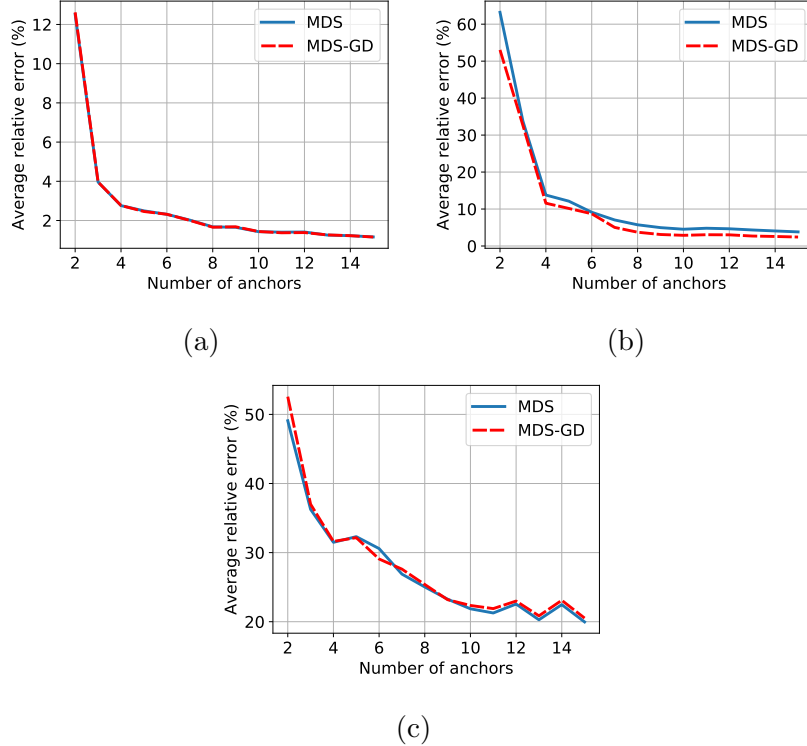


Figure 2.2: Experiments in simulation and on real hardware to evaluate the linearity error as defined in (2.9). The input signals are of dimension 64^2 , M in (2.9) is 100 and the number of anchors signals are varied. The classical MDS and MDS with gradient descent (MDS-GD) are used. In all cases, the error decreases as the number of anchors increases. (a) In simulation with Gaussian signals and Gaussian reference signals; (b) In simulation with Gaussian signals and Gaussian reference signals with sensitivity threshold $\tau = 6$; (c) On a real OPU with binary signals and binary references.

improvement in Figure 2.2b.

We also test a scenario where the anchor positions on the complex plane are known exactly and we only have to localize a single measurement. We compare this to localizing the anchors and the measurements jointly. Localizing a single point via multilateration is performed by minimizing the SR-LS objective (see (A.1) in the Appendix). The input signal dimension is 64^2 and we recover \hat{y} for $|y|^2 = |\langle \mathbf{a}, \boldsymbol{\xi} \rangle|^2$. We perform 100 trials and calculate the SNR of the recovered complex points. Figure 2.3c shows that, although having perfect knowledge of anchor locations helps, classical MDS alone does not perform much worse.

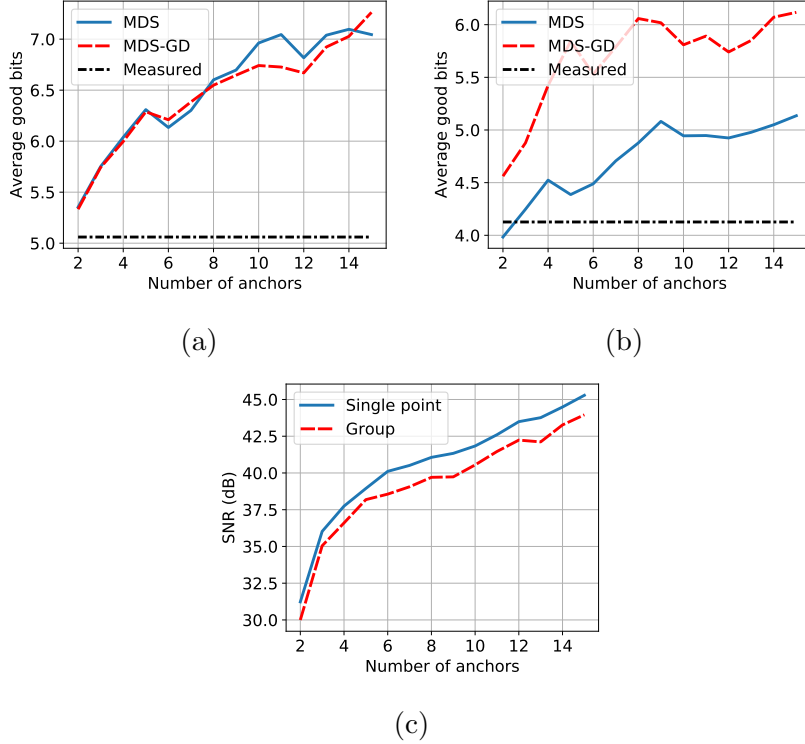


Figure 2.3: (a) Magnitude denoising performance of MDS and MDS-GD over 100 trials. Input signals are Gaussian and of dimension 100; (b) Magnitude denoising performance of MDS and MDS-GD over 100 trials with 100-dimensional Gaussian signals and sensitivity threshold $\tau = 6$; (c) Comparison between recovering a single point and recovering the point and anchors at the same time. SR-LS is used to locate a single point when anchors are known and MDS is used to locate all points when anchors are unknown.

Optical randomized singular value decomposition. We use Algorithm 1 to implement randomized singular value decomposition (RSVD) as described in Halko et al. [44] on the OPU. We use five anchors in all RSVD experiments. The original RSVD algorithm and a variant with adaptations for the OPU are described in Algorithms 3 and 4 of Appendix A.2.

One of the steps in the RSVD algorithm for an input matrix $\mathbf{B} \in \mathbb{R}^{M \times N}$ requires the computation of $\mathbf{B}\mathbf{\Omega}$ where $\mathbf{\Omega} \in \mathbb{R}^{N \times 2K}$ is a standard real Gaussian matrix, K is the target number of singular vectors, and $2K$ may be interpreted as the number of random projections for each row of \mathbf{B} . We use the OPU to compute this random matrix multiplication. An interesting observation is that since in Algorithm 1 we recover the result of

multiplications by a complex matrix with independent real and imaginary parts, we can halve the number of projections when using the OPU with respect to the original algorithm. By treating each row of \mathbf{B} as an input frame, we can obtain $\mathbf{Y} \in \mathbb{C}^{K \times M}$ via Algorithm 1 when $|\mathbf{Y}|^2 = |\mathbf{A}\mathbf{B}^T|^2$ with \mathbf{A} as defined in Problem 1 with K rows. Then, we can construct $\mathbf{P} = \begin{bmatrix} \text{Re}(\mathbf{Y}^*) & \text{Im}(\mathbf{Y}^*) \end{bmatrix} \in \mathbb{R}^{M \times 2K}$ which would be equivalent to computing $\mathbf{B}\mathbf{\Omega}$ for real $\mathbf{\Omega}$. Appendix A.2 describes this in more detail.

Figure 2.4 shows the results when the OPU is used to perform the random matrix multiplication step of the RSVD algorithm on a matrix \mathbf{B} . Figure 2.4 (left) reports experiments with a random binary matrix $\mathbf{B} \in \mathbb{R}^{10 \times 10^4}$, different numbers of random projections (number of rows in \mathbf{A}), and ten trials per number of projections. We plot the average error per entry when reconstructing \mathbf{B} from its RSVD matrices and singular values. Next, we take 500 28×28 samples from the MNIST dataset [45], threshold them to be binary, vectorize them, and stack them into a matrix $\mathbf{B} \in \mathbb{R}^{500 \times 28^2}$. Figure 2.4 (right) shows the seven leading right singular vectors reshaped to 28×28 . The top row shows the singular vectors that are obtained when using the OPU with 500 projections, and the bottom row shows the result when using Python. The error is negligible.

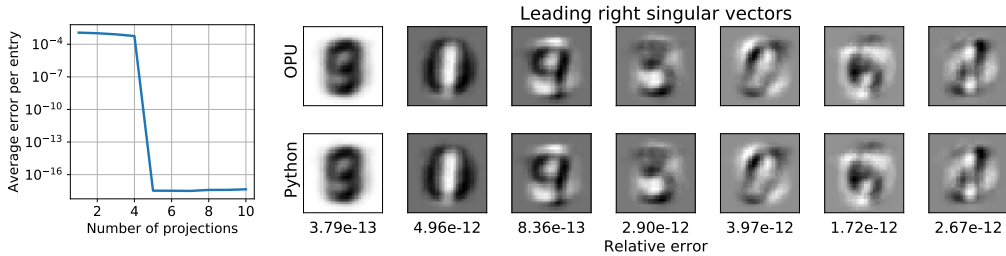


Figure 2.4: Left: Average RSVD error over 10 trials with varying number of projections on hardware with an input matrix of size 10×1000 ; Right: Reshaped leading right singular vectors of an MNIST matrix of size 500×28^2 . The top rows shows the leading right singular vectors after performing RSVD with the OPU and using our algorithm. The bottom row shows the leading right singular vectors from Python. The relative error is below each singular vector.

2.4 Summary

Traditional computation methods are often too slow for processing tasks which involve large data streams. This motivates alternatives which instead use fast physics to “compute” the desired functions. In this chapter, we looked at using optics and multiple scattering media to obtain linear random projections. A common difficulty with optical systems is that off-the-shelf camera sensors only register the intensity of the scattered light. Our results show that there is nevertheless no need to reach for more complicated and more expensive setups. We showed that measurement phase retrieval can be cast as a problem in distance geometry, and that the unknown phase of random projections can be recovered even without knowing the transmission matrix of the medium.

Simulations and experiments on real hardware show that the OPU setup combined with our algorithm indeed approximates an end-to-end linear system. What is more, we also improve intensity measurements. The fact that we get full complex measurements allows us to implement a whole new spectrum of randomized algorithms; we demonstrated the potential by the randomized singular value decomposition.

CHAPTER 3

RAPID TM CALIBRATION

We build on Chapter 2 to develop a fast measurement phase retrieval method. As we control the input and can now obtain the measurements, the only unknown is the TM. Using the new measurement phase retrieval technique, we rapidly calibrate the TM by solving a simple linear system rather than a time-consuming quadratic system. We then use the calibrated TMs to image through scattering media as explained in Figure 1.1. This work falls under both **E1** and **E2** as we do measurement phase and transmission matrix retrieval.

3.1 Imaging through scattering media

Imaging through a complex optical medium is conceptually simple. The relationship between the optical field in the input plane, \mathbf{x} , and the scattered optical field, \mathbf{y} , is given as

$$\mathbf{y} = \mathbf{A}\mathbf{x}, \tag{3.1}$$

with \mathbf{A} being the transmission matrix (TM) of the medium (the optical system). Hence, to find the unknown image \mathbf{x} from measurements \mathbf{y} it suffices to solve the linear inverse problem (3.1). Alas, once we set out to do so we quickly realize that 1) the transmission matrix \mathbf{A} is typically unknown, and 2) $\mathbf{A} \in \mathbb{C}^{M \times N}$ and $\mathbf{y} \in \mathbb{C}^M$ in (3.1) are complex, but typical camera sensors only measure the intensity $|\mathbf{y}|^2$. Finding a solution for these two difficulties is key to multiple applications such as imaging through fog, paint

This work was previously published in S. Gupta, R. Gribonval, L. Daudet, and I. Dokmanić, “Fast Optical System Identification by Numerical Interferometry,” in ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2020, pp. 1474–1478. [46] ©2020 IEEE and is adapted here with permission.

and tissues in the human body, as well as optics-accelerated signal processing [20] (Chapter 2).

There are several ways forward. One is to exploit the statistical properties of \mathbf{A} . This is useful in media like tissues or fog, where \mathbf{A} exhibits certain Gaussian statistics and it enables the design of correlation-based imaging algorithms [47]. The other way, which gives better images and enables a score of other applications, is to somehow learn \mathbf{A} . Unfortunately, identifying \mathbf{A} by probing signals in multiply-scattering media is computationally demanding. Since the phase of the measurements is unknown, it amounts to solving a system of quadratic equations. Due to this difficulty, state-of-the-art methods report long calibration times even for small \mathbf{A} . As an example, Rajaei *et al.* proposed an approximate message passing algorithm known as prSAMP [48] which would take days to reconstruct a TM of size $256^2 \times 64^2$ (that is, for input images of size 64×64). Sharma *et al.* [39] improve this by bringing the calibration time down to under five hours for a matrix of the same size.

3.1.1 Chapter overview and contributions

In this chapter, we propose a fast numerical interferometry method to rapidly identify \mathbf{A} . Running on our system, our proposed method takes 6 minutes for a calibration problem where the method of Sharma *et al.* [39] takes 200 minutes (3.29 hours).¹

The gist of our method is in the special design of the calibration inputs which we control. These inputs allow us to numerically find the phases of the measurements \mathbf{y} by “multilateration” in the complex plane, and then numerically find \mathbf{A} by inverting a linear system. Furthermore, instead of direct inversion, we design calibration inputs so that the linear system is solved efficiently by the fast Fourier transform (FFT). The outline of the algorithm is

1. Recover the phase of the calibration measurements by solving a distance geometry problem similar to Chapter 2 [20];
2. With the measurement phase recovered, estimate the transmission matrix \mathbf{A} by solving a linear system;

¹GPU accelerated code for Sharma *et al.* downloaded from link in their paper. All parameters were kept the same and the same GPU was used.

3. With \mathbf{A} in hand, use a phase retrieval method such as Wirtinger flow [28] to find \mathbf{x} from $|\mathbf{y}|^2$.

We build upon our previous work in Chapter 2 of casting measurement phase retrieval as a distance geometry problem [20]. However, since here we are after speed, we multilaterate all the entries of \mathbf{A} by solving a suitable linear system. We verify that the transmission matrices we compute are correct by using them to reconstruct a known input signal. Experiments on real optical hardware show that our method works even with quantization and other inevitable hardware noise, and that it only takes a few minutes compared to hours with the previous state of the art.

3.1.2 Related work

In a coherent interferometric setup the phase of \mathbf{y} can be measured directly, thus immediately giving a linear system to find \mathbf{A} [49, 50, 51, 52]. The downside is that such devices are harder and more expensive to build, often achieve lower frame rates, and are sensitive to environmental factors [26, 50].

A numerical alternative with intensity-only measurements is the double phase retrieval technique [26, 53]. In double phase retrieval, the TM is directly estimated from measurements obtained by known probe signals at the input. Calibration is posed as a quadratic phase retrieval problem, with the rows of the TM being the signals to be recovered. Here the known probe signals act as the observation matrix, and the TM is the unknown. Solving these phase retrieval problems can be time consuming even with GPU-accelerated implementations [39].

We previously introduced a distance geometry approach to find the phase of \mathbf{y} in Chapter 2 [20]. In that work, the focus was on computing the phase of Gaussian random projections for machine learning tasks such as dimensionality reduction and kernel methods, without determining \mathbf{A} . Thus, while directly adapting the approach presented there for calibration is already faster than the double phase retrieval method, in this chapter we propose methods that are much faster than both.

Knowing \mathbf{A} enables reconstructing \mathbf{x} from the magnitude of complex measurements $|\mathbf{y}|^2 = |\mathbf{A}\mathbf{x}|^2$ via classical phase retrieval techniques [33, 34, 7, 28]. Furthermore, in optically-accelerated computing there are uses for fast mul-

tiplication of large signals with known random Gaussian iid matrices. Some applications are classification with random features [21], matrix sketching [23], and randomized linear algebra [44].

3.2 Rapid transmission matrix computation

In this section we describe the procedure outlined in Section 3.1 in detail. We adopt the classical system identification paradigm and probe the optical system with a collection of K known calibration signals arranged in the matrix $\Xi = [\xi_1, \dots, \xi_K] \in \mathbb{R}^{N \times K}$,

To convert phase retrieval into multilateration we will also use S known signals $\mathbf{v}_1, \dots, \mathbf{v}_S$, which we ascribe to the columns of $\mathbf{V} \in \mathbb{R}^{N \times S}$. Without loss of generality we fix \mathbf{v}_S to be the origin. We will call $\mathbf{A}\mathbf{v}_s$ *anchors*.

We now feed the calibration signals through the optical system, forming

$$\mathbf{Y} = \mathbf{A}\Xi. \quad (3.2)$$

The k th column of \mathbf{Y} will be denoted $\mathbf{y}_k = \mathbf{A}\xi_k$. We do the same with the differences between the columns of Ξ and the vectors \mathbf{v}_s ,

$$\mathbf{r}_s := \mathbf{A}\mathbf{v}_s \quad \text{and} \quad \mathbf{y}_{ks} := \mathbf{A}(\xi_k - \mathbf{v}_s) = \mathbf{y}_k - \mathbf{r}_s, \quad (3.3)$$

for all $k \in \{1, \dots, K\}$ and $s \in \{1, \dots, S\}$. The m th entry of these vectors will be denoted $y_{k,m}$, $r_{s,m}$ and $y_{ks,m}$; the m th row of \mathbf{A} will be denoted \mathbf{a}^m . Optically, we can only measure the entrywise magnitudes of these vectors and matrices. The term *numerical interferometry* comes from the fact that the optically measured magnitude of \mathbf{y}_{ks} is the interference pattern between $\mathbf{A}\xi_k$ and $\mathbf{A}\mathbf{v}_s$.

3.2.1 Rapid measurement phase retrieval

We consider recovering the phase of the calibration measurements \mathbf{Y} from the system

$$|\mathbf{Y}|^2 = |\mathbf{A}\Xi|^2. \quad (3.4)$$

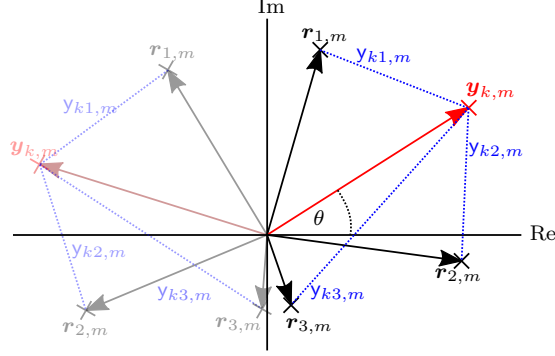


Figure 3.1: Two sets of points on the complex plane which are related by reflection and rotation transformations. $\mathbf{y}_{k,m}$ is the position of calibration signal k on the complex plane. We can optically measure the squared distance between this point and anchor points on the complex plane to obtain the calibration signal measurement phase. ©2020 IEEE.

We begin by noting that the absolute phase of $\mathbf{A}\mathbf{E}$ in (3.4) cannot be recovered since conjugating a subset of rows of \mathbf{A} or adding arbitrary constant phases leads to the same magnitude measurements. This ambiguity is standard in phase retrieval [28].

We remark that for many multiple scattering media, and in particular for the setup used in our experiments, \mathbf{A} is approximately iid standard complex Gaussian. As such, adding constant phase to rows or conjugating them does not change their distribution. As long as the relative phase between the *columns* of \mathbf{Y} does not change, our method recovers an \mathbf{A} which is related to the true one by row phasing and conjugation. This is innocuous for many applications.

Without loss of generality, we explain the rapid measurement phase retrieval algorithm for the m th row of \mathbf{A} . This uses similar ideas to the method in Chapter 2. We begin with the trivial observation that the magnitude of a complex number $r_{s,m} := |r_{s,m}|$ is its distance to the origin. Similarly, the magnitude of a difference between $y_{k,m}$ and $r_{s,m}$, $y_{ks,m} := |y_{k,m} - r_{s,m}| = |\langle \mathbf{a}^m, \boldsymbol{\xi}_k - \mathbf{v}_s \rangle|$, is the distance between these numbers in the complex plane (see Figure 3.1).

Suppose for a moment that the points $r_{s,m}$, $s \in \{1, \dots, S\}$ are known and fixed and that we wish to recover the phase of $y_{k,m}$. By having access to the distances from $y_{k,m}$ to at least three anchors $r_{s,m}$, we can “localize” $y_{k,m}$ by trilateration and hence recover its phase. In practice, having more anchors

leads to more robust results.

To get a fast method, we now show how the phase of all the entries in the m th row of \mathbf{Y} can be computed at once by solving a small linear system with multiple right-hand sides. We adapt a procedure from [54]. First we expand the squared distance to anchors,

$$y_{ks,m}^2 = r_{s,m}^2 + y_{k,m}^2 - 2r_{s,m}^T y_{k,m},$$

where we abused notation by interpreting complex numbers as vectors in \mathbb{R}^2 that can be transposed. Rearranging, we get

$$y_{ks,m}^2 - r_{s,m}^2 = \underbrace{[-2r_{s,m}^T, 1]}_{=: \mathbf{m}_{s,m}^T} \underbrace{\begin{bmatrix} y_{k,m} \\ y_{k,m}^2 \end{bmatrix}}_{=: \mathbf{w}_{k,m} \in \mathbb{R}^3}. \quad (3.5)$$

In (3.5), the left-hand side contains only known quantities (anchors are assumed known and $y_{ks,m}^2$ is optically measured). The row vector on the right-hand side is also known, while the column vector contains the desired, unknown complex point.

With S anchors we get S equations for the column vector in (3.5); the system is invertible if $S \geq 3$ and the anchors are not colinear. Let

$$\mathbf{e}_{k,m} = [y_{k1,m}^2 - r_{1,m}^2, \dots, y_{kS,m}^2 - r_{S,m}^2]^T$$

denote the stack of the left-hand sides of (3.5) for all S anchors, and $\mathbf{E}_m = [\mathbf{e}_{1,m}, \dots, \mathbf{e}_{K,m}] \in \mathbb{R}^{S \times K}$ a horizontal stack of $\mathbf{e}_{k,m}$ for all K calibration signals. Similarly, let $\mathbf{M}_m = [\mathbf{m}_{1,m}, \dots, \mathbf{m}_{S,m}]^T \in \mathbb{R}^{S \times 3}$ and $\mathbf{W}_m = [\mathbf{w}_{1,m}, \dots, \mathbf{w}_{K,m}] \in \mathbb{R}^{3 \times K}$. We can then write (3.5) for all anchors and calibration signals at once as

$$\mathbf{E}_m = \mathbf{M}_m \mathbf{W}_m. \quad (3.6)$$

By solving multiple multilateration problems in (3.6), we get $\widehat{\mathbf{W}}_m = \mathbf{M}_m^\dagger \mathbf{E}_m$. The top two rows of $\widehat{\mathbf{W}}_m$ contain the real and imaginary parts of all the entries in the m th row of \mathbf{Y} , as shown in Figure 3.1.

3.2.2 Initial anchor positioning

For the developments of the previous section to make sense, we need to know the anchor positions. Here we propose to follow Chapter 2 and use classical multidimensional scaling (MDS) as summarized below.

We begin by optically measuring the anchors and all their pairwise differences. For row m of \mathbf{A} for all (q, s) we can get $S(S-1)/2$ squared Euclidean distances between points $\{r_{s,m}\}_{s=1}^S$ on the complex plane,

$$|\langle \mathbf{a}^m, \mathbf{v}_q - \mathbf{v}_s \rangle|^2 = |r_{q,m} - r_{s,m}|^2. \quad (3.7)$$

We arrange these distances into a squared Euclidean distance matrix, $\mathbf{D}_m \in \mathbb{R}^{S \times S}$, where $d_{qs,m} = |r_{q,m} - r_{s,m}|^2$. Next, denoting the geometric centering matrix $\mathbf{J} := \mathbf{I}_S - \frac{1}{S} \mathbf{1}_S \mathbf{1}_S^T$ where $\mathbf{1}_S$ is a column vector of S ones, we compute the eigendecomposition of

$$\mathbf{G}_m = -\frac{1}{2} \mathbf{J} \mathbf{D}_m \mathbf{J} \quad (3.8)$$

as $\mathbf{G}_m = \mathbf{U} \text{diag}(\lambda_1, \dots, \lambda_S) \mathbf{U}^T$, where the eigenvalue sequence $(\lambda_s)_{s=1}^S$ is nonincreasing. Anchor s is then located on the complex plane at the s th element of $(\sqrt{\lambda_1} \mathbf{u}_1 + j \sqrt{\lambda_2} \mathbf{u}_2)$ where \mathbf{u}_1 and \mathbf{u}_2 are the first and second columns of \mathbf{U} . As \mathbf{v}_S was set to be the origin, we subtract the S th element of $(\sqrt{\lambda_1} \mathbf{u}_1 + j \sqrt{\lambda_2} \mathbf{u}_2)$ from all localized anchors.

With the anchors localized, we can solve (3.6) to retrieve the measurement phase for each row. We note that for each row we can apply rotation and reflection transformations to the set of anchor points in the two-dimensional complex plane while still maintaining the optically measured distances. As mentioned earlier, rotation and reflection correspond to adding constant phase or conjugating rows.

We note that this approach of measurement phase retrieval varies from the one presented earlier in Chapter 2. In that chapter, rather than using (3.6) to obtain the phase of the measurements, we used MDS to solve a separate distance geometry problem for each element of \mathbf{Y} which is much more time consuming.

3.3 Obtaining the transmission matrix

With the phases of \mathbf{Y} computed and recalling that the probe signals $\mathbf{\Xi}$ are known, we can numerically compute the transmission matrix by solving

$$\mathbf{Y} = \mathbf{A}\mathbf{\Xi}. \quad (3.9)$$

The least-squares fit $\hat{\mathbf{A}} = \arg \min_{\mathbf{A}} \|\mathbf{Y} - \mathbf{A}\mathbf{\Xi}\|_F^2$ is formally given by the pseudoinverse. We design $\mathbf{\Xi}$ to ensure that it has full row rank and that the number of probe signals, K , is larger than N ; then $\hat{\mathbf{A}} = \mathbf{Y}\mathbf{\Xi}^\dagger$, where † denotes the pseudoinverse.

Although this method (see Section 3.4) already gives significant speed gains over the state-of-the-art, for large signals and large K , calculating the pseudoinverse can be slow. We next show how to further speed up the algorithm using fast Fourier transforms (FFTs) and a generalized right inverse of $\mathbf{\Xi}$ instead of a pseudoinverse.

3.3.1 Circulant probes for FFT

We first note that due to noise and other adversarial effects, it is favorable to have more calibration signals than the minimum number N . For simplicity, in this section we choose $K = 2N$. If $\mathbf{\Xi}^\dagger$ was a circulant matrix, the multiplication $\mathbf{Y}\mathbf{\Xi}^\dagger$ would be efficiently computed by an FFT as circulant matrices are diagonalized with the discrete Fourier transform (DFT) matrix. With this in mind, we design $\mathbf{\Xi}$ instead of $\mathbf{\Xi}^\dagger$. We use the following proposition with \mathbf{F} denoting the unitary DFT matrix and $*$ denoting the Hermitian transpose:

Proposition 1. *Let $\mathbf{C}_1, \mathbf{C}_2$ be invertible circulant matrices of same size, diagonalized as $\mathbf{F}^* \mathbf{\Lambda}_1 \mathbf{F}$ and $\mathbf{F}^* \mathbf{\Lambda}_2 \mathbf{F}$. Then for any α, β such that $\alpha + \beta = 1$, $[\alpha \mathbf{C}_1^{-1}, \beta \mathbf{C}_2^{-1}]^T$ is a generalized right inverse of $[\mathbf{C}_1, \mathbf{C}_2]$, with \mathbf{C}_1^{-1} and \mathbf{C}_2^{-1} being the circulant matrices $\mathbf{F}^* \mathbf{\Lambda}_1^{-1} \mathbf{F}$ and $\mathbf{F}^* \mathbf{\Lambda}_2^{-1} \mathbf{F}$.*

We design $\mathbf{\Xi}$ as a concatenation of two circulant matrices of size $N \times N$, $\mathbf{\Xi} = [\mathbf{\Xi}_A, \mathbf{\Xi}_B] \in \mathbb{R}^{N \times 2N}$. They are diagonalized by the DFT matrix $\mathbf{F} \in \mathbb{C}^{N \times N}$,

$$\mathbf{\Xi}_A = \mathbf{F}^* \mathbf{\Lambda}_A \mathbf{F} \quad \text{and} \quad \mathbf{\Xi}_B = \mathbf{F}^* \mathbf{\Lambda}_B \mathbf{F},$$

where $\mathbf{\Lambda}_A$ and $\mathbf{\Lambda}_B$ are diagonal eigenvalue matrices whose entries are the DFT of the first columns of $\mathbf{\Xi}_A$ and $\mathbf{\Xi}_B$, and $f_{mn} = e^{-j2\pi mn/N}/\sqrt{N}$. With this notation we can write

$$\mathbf{Y} = \mathbf{A} \begin{bmatrix} \mathbf{F}^* \mathbf{\Lambda}_A \mathbf{F} & \mathbf{F}^* \mathbf{\Lambda}_B \mathbf{F} \end{bmatrix}.$$

From here, splitting \mathbf{Y} into halves as $\mathbf{Y} = [\mathbf{Y}_A, \mathbf{Y}_B]$ and applying the proposition with $\alpha = \beta = \frac{1}{2}$ to obtain a right inverse gives

$$\hat{\mathbf{A}} = \frac{1}{2} (\mathbf{Y}_A \mathbf{F}^* \mathbf{\Lambda}_A^{-1} + \mathbf{Y}_B \mathbf{F}^* \mathbf{\Lambda}_B^{-1}) \mathbf{F} \quad (3.10)$$

Multiplications by \mathbf{F} and \mathbf{F}^* can be implemented efficiently using the FFT in time $\mathcal{O}(N \log N)$. Since $\mathbf{\Lambda}_A$ and $\mathbf{\Lambda}_B$ are diagonal, multiplying by them takes time $\mathcal{O}(N)$ and their entries are calculated efficiently using the FFT.

3.3.2 Algorithm complexity

To evaluate our algorithm's complexity, we first consider measurement phase retrieval which is the same process repeated M times. Anchor localization via MDS entails creating \mathbf{G}_m (3.8) which is $\mathcal{O}(S^2)$ thanks to the special structure of \mathbf{J} , and performing a singular value decomposition on it which is $\mathcal{O}(S^3)$. Next, with $K = 2N$, solving (3.6) requires some $\mathcal{O}(SN)$ operations to obtain \mathbf{E}_m plus computing a pseudoinverse of an $S \times 3$ matrix which is $\mathcal{O}(S)$ and finally multiplying the pseudoinverse which is $\mathcal{O}(SN)$. As these steps are repeated M times, the complexity of measurement phase retrieval is $\mathcal{O}(MS^3) + \mathcal{O}(MSN)$. The number of anchors, S , is a fixed parameter which gives $\mathcal{O}(MN)$.

Recovering \mathbf{A} from $\mathbf{Y} = \mathbf{A}\mathbf{\Xi}$ via (3.10) involves three FFT uses with each time on M signals of length N which gives complexity $\mathcal{O}(MN \log N)$. Multiplications with the diagonal matrices are $\mathcal{O}(MN)$ and so computing (3.10) is $\mathcal{O}(MN \log N)$. Putting this together with $\mathcal{O}(MN)$ from measurement phase retrieval results in our algorithm ultimately being $\mathcal{O}(MN \log N)$. Without using the FFT, it would have been $\mathcal{O}(N^3) + \mathcal{O}(MN^2)$.

3.4 Experimental verification

We numerically compute TMs from real optical hardware measurements and use them for optical imaging as described in Figure 1.1. We use the same scikit-learn interface to a publicly available cloud-based optical processing unit (OPU) as in Chapter 2.² As shown in Figure 1.2, the OPU “imprints” a signal onto a coherent light beam using a digital micro-mirror device (DMD) which is then shined through a multiple scattering medium such as a white polymer layer. The scattering medium acts like an iid standard complex Gaussian matrix. A camera with 8-bit precision measures the intensity of the scattered light.

As explained in Appendix A.1, one challenge of using this hardware is that the DMD only produces binary inputs. Therefore, the anchor signals have to be designed so that the difference between any two anchors is binary and they can be used to localize anchors on the complex plane. Furthermore, the probe signals, Ξ , should be binary and remain binary when any anchor signal is subtracted from them. Using the scheme described in Appendix A.1, we design anchors by summing all probe signals and existing anchors, thresholding indices with values greater than one to one and then making 15% of the values at indices where the sum was zero to one.

From (3.9), we need all rows of Ξ to have at least one nonzero value in order to reconstruct all columns of \mathbf{A} . However, this results in anchors which are all one. We therefore apply our method twice with two different sets of probe signals with different zero rows. Each set of probe signals recovers a subset of the columns of the TM. Since the two sets of probe signals share some non-zero row indices, the corresponding recovered columns should be identical. We use this fact to rotate and conjugate the rows of the two recovered TMs so that the common columns coincide. Finally, columns of \mathbf{A} that are only recovered by one set of probing signals are collected to form a complete TM.

More concretely, each $\Xi \in \mathbb{R}^{N \times K}$ is built by concatenating two circulant matrices, each of size $K/2 \times K/2$. The entries of the non-zero columns of the circulant matrices are drawn iid from $\text{Bernoulli}(\frac{1}{2})$ and $N - (K/2)$ all-zero rows are inserted at random indices so that Ξ is $N \times K$. When inverting Ξ

²<https://www.lighton.ai/lighton-cloud/>.
Code available at https://github.com/swing-research/numerical_interferometry

N	M/N	Time taken (minutes)	Reconstructed image relative error
32^2	32	0.97	10.2%
32^2	64	2.05	7.8%
32^2	128	4.01	6.2%
64^2	16	6.15	21.5%
64^2	32	11.69	15.3%
64^2	64	24.14	12.0%
96^2	16	31.36	25.8%
128^2	12	71.97	32.0%

Table 3.1: The time taken to reconstruct TMs, $\mathbf{A} \in \mathbb{C}^{M \times N}$ and the relative error of reconstructed images using these TMs. All times were measured on the same system. ©2020 IEEE.

to calibrate the TM, we remove the inserted zero rows to get block circulant matrices and use (3.10) to recover a subset of the columns of \mathbf{A} .

3.4.1 Optical imaging with numerically computed TMs

We reconstruct images of different sizes and vary the oversampling factor, M/N , of the optically-obtained quadratic magnitude measurements which results in different TM sizes. Wirtinger flow with random uniform initialization is used for reconstruction [28, 55]. During gradient descent, any elements of the signal with absolute value greater than one are normalized. We do 500 iterations of gradient descent for 32×32 images. For the other images, the absolute value of the recovered signal after 500 initial iterations with $4N$ measurements is used to start 2000 further iterations which includes all measurements. We use 20 anchor signals for all images except the 128×128 ones which use 15. For 32×32 and 64×64 images $K = 1.5N$, for 96×96 , $K = 1.125N$ and for 128×128 , $K = 1.03125N$.

Table 3.1 shows the TM calibration time as well as the reconstructed image relative error which we define as $\min_{\varphi \in [0, 2\pi)} \|\mathbf{x} - e^{j\varphi} \hat{\mathbf{x}}\|_2 / \|\mathbf{x}\|_2$ where $\hat{\mathbf{x}}$ is the reconstructed signal and \mathbf{x} is the true signal. We note that we are doing phase retrieval with noise as the TM has been estimated from 8-bit precision camera measurements. The error decreases with increasing oversampling factor, which matches the theory. As mentioned in Section 3.1, a $256^2 \times 64^2$

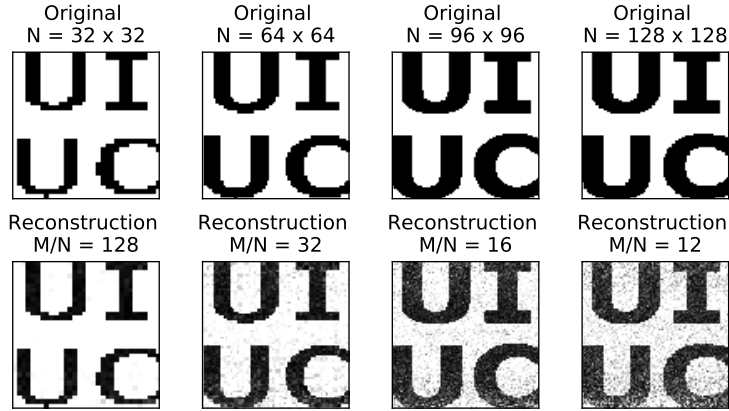


Figure 3.2: Reconstructions with some of the transmission matrices reported in Table 3.1. The top row shows the original, and the bottom row shows the absolute value of its corresponding reconstruction. ©2020 IEEE.

TM took 3.26 hours when prVAMP was used [39]. In contrast, our method takes 6.15 minutes. In fact, the biggest TM listed in Table 3.1 is 12 times the size of this TM. Figure 3.2 displays some reconstructions using our calibrated TMs.

3.4.2 Computation time scaling

From Table 3.1 and Figure 3.3 (left) we can see that the time taken increases linearly with the oversampling factor. In Figure 3.3 (left), $N = 64^2$, $K = 1.5N$ and 20 anchor signals are used. In Figure 3.3 (right), the oversampling factor is fixed at 16, $K = 1.125N$, 20 anchors are used and the input dimension is varied. As signal dimension increases, using the FFT to solve (3.9) yields increasing gains.

3.5 Summary

Calibrating a transmission matrix with intensity-only measurements has traditionally been a time consuming process. In this chapter, we proposed a method to reduce the calibration time for typical size matrices from hours to minutes. The crux of our method is the rapid recovery of the measurement phase due to probe input signals. With the obtained measurement phase

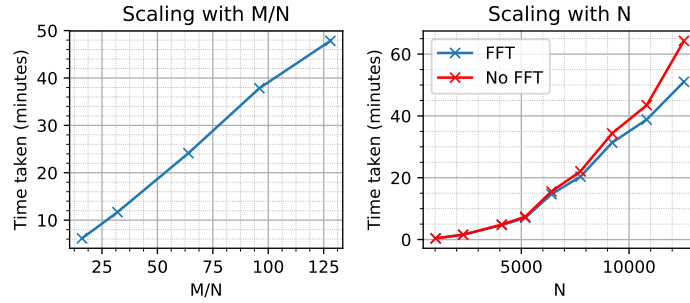


Figure 3.3: (Left) Time taken to calibrate TM when the oversampling factor is varied; (Right) Time taken when input dimension varies and oversampling factor is fixed at 16. ©2020 IEEE.

and probe signals, transmission matrix calibration then amounts to solving a simple linear system, which we do efficiently with the FFT. Experiments on optical hardware confirm that our method is indeed faster than existing methods and reconstructs transmission matrices which can be used for imaging.

CHAPTER 4

TOTAL LEAST SQUARES PHASE RETRIEVAL

In Chapter 3, we presented a method to calibrate the TMs of scattering media from real optical measurements. However, due to camera quantization and thermal effects, these optical measurements have errors. Consequently, the calibrated TMs have errors which are not accounted for when imaging through scattering media (Figure 3.2 which can result in poor reconstruction. Therefore, in this chapter, we present a framework to account for both measurement and TM errors when solving phase retrieval problems such as imaging through scattering media. This work falls under **E2** because we correct calibrated TMs.

4.1 Phase retrieval with sensing system errors

In the classical phase retrieval problem, we seek to recover the signal $\mathbf{x} \in \mathbb{C}^N$ from complex quadratic measurements

$$y_m \approx |\langle \mathbf{a}_m, \mathbf{x} \rangle|^2, \quad m = 1, \dots, M \quad (4.1)$$

where $y_m \in \mathbb{R}$ are observed measurements and $\mathbf{a}_m \in \mathbb{C}^N$ are sensing vectors. This problem appears in applied science applications such as x-ray diffraction crystallography or astronomy where the sensing vectors are Fourier basis vectors [57] and imaging through scattering media where the sensing vectors may be complex random Gaussian [46].

In a prototypical phase retrieval problem, an object, \mathbf{x} , is illuminated and the resulting optical field is measured with a detector. This optical field is complex-valued but common camera sensors only measure intensity,

This work was previously published in S. Gupta and I. Dokmanić, “Total Least Squares Phase Retrieval,” IEEE Transactions on Signal Processing, vol. 70, pp. 536–549, 2022 [56] and is adapted here with permission.

$\{|\langle \mathbf{a}_m, \mathbf{x} \rangle|^2\}_{m=1}^M$, and thus the measurement phase information is lost. The left and right hand sides in (4.1) are only approximately equal because in practical settings there can be errors in both the measurements and sensing vectors. In this work, we focus on gradient-based optimization strategies to solve (4.1) when $M > N$. Gradient-based methods have proven successful when imaging through random scattering media [46] or with coded diffraction Fourier patterns [28].

Many recent approaches for solving the phase retrieval problem solve variants of the following nonlinear and nonconvex least squares (LS) problem,

$$\underset{\mathbf{x}}{\text{minimize}} \sum_{m=1}^M (y_m - |\langle \mathbf{a}_m, \mathbf{x} \rangle|^2)^2, \quad (\text{LS-PR})$$

which we can alternatively rewrite as

$$\begin{aligned} & \underset{\substack{\mathbf{x}, \\ r_1, \dots, r_M}}{\text{minimize}} \quad \sum_{m=1}^M r_m^2 & (4.2) \\ & \text{s.t.} \quad y_m + r_m = |\langle \mathbf{a}_m, \mathbf{x} \rangle|^2, \quad m = 1, \dots, M \end{aligned}$$

with $r_m \in \mathbb{R}$. Thus, LS seeks the smallest correction to the measurements so that $(y_m + r_m)$ can be obtained from quadratic measurements $|\langle \mathbf{a}_m, \mathbf{x} \rangle|^2$ for each m . This is analogous to LS for linear inverse problems where corrections that bring the measurements into the range space of the linear operator are required instead.

In many practical settings, the sensing vectors, $\{\mathbf{a}_m\}_{m=1}^M$, are only approximately known via calibration. In this chapter we show that properly accounting for errors in the sensing vectors may lead to a more accurate estimate of \mathbf{x} . Inspired by the total least squares (TLS) framework for linear [58, 59] and nonlinear [60] inverse problems, we extend the LS formulation (4.2) to find corrections for both the measurements and the sensing vectors. In TLS phase retrieval, we optimize the objective

$$\begin{aligned} & \underset{\substack{\mathbf{x}, \\ r_1, \dots, r_M, \\ \mathbf{e}_1, \dots, \mathbf{e}_M}}{\text{minimize}} \quad \sum_{m=1}^M \lambda_y r_m^2 + \lambda_a \|\mathbf{e}_m\|_2^2 & (4.3) \\ & \text{s.t.} \quad y_m + r_m = |\langle \mathbf{a}_m + \mathbf{e}_m, \mathbf{x} \rangle|^2, \quad m = 1, \dots, M \end{aligned}$$

with corrections $\mathbf{e}_m \in \mathbb{C}^N$ for $1 \leq m \leq M$. Scalars $\lambda_y \in \mathbb{R}$ and $\lambda_a \in \mathbb{R}$ are nonnegative regularization weights. Now, for each m we want to find minimum weighted norm corrections so that $(y_m + r_m)$ can be obtained from quadratic measurements $|\langle \mathbf{a}_m + \mathbf{e}_m, \mathbf{x} \rangle|^2$. Efficiently obtaining the sensing vector corrections $\{\mathbf{e}_m\}_{m=1}^M$ is a major challenge when moving from the LS problem (4.2) to the TLS problem (4.3).

4.1.1 Chapter overview and contributions

We propose a TLS framework for solving the phase retrieval problem when there are errors in the sensing vectors. In Section 4.2 we explain our gradient descent strategy to solve the TLS phase retrieval problem which motivates an alternating updates procedure to solve the problem. With this approach there are additional computational challenges, which we show can be made efficient by incorporating the geometry of the phase retrieval problem. In Section 4.3 we derive expressions for the reconstruction errors for the TLS and LS solutions. This gives us insight into when each method should perform well. This derivation requires the usage of theorems about differentiation of argmins and different matrix inversion lemmas. Through simulations in Section 4.4 we show that the TLS approach can lead to solutions of greater accuracy when there are sensing vector errors. We further verify the applicability of our framework through experiments on real optical hardware in Section 4.5. We see that TLS outperforms LS when aiming to recover random signals and real images. We summarize TLS phase retrieval in Section 4.6.

4.1.2 Related work

Algorithms by Gerchberg and Saxton [61] and Fienup [62] are the most well-known approaches for solving the phase retrieval problem when the sensing vectors are the rows of the Fourier matrix, as in many practical imaging scenarios [1]. These methods iteratively reduce the error between the observed measurements and the measurements generated from the solution at the current iterate. Another class of algorithms based on message passing has also been developed [48, 39]. Despite the nonconvexity of the problem,

these error reduction and message passing algorithms work well in practice. They do not directly use gradient descent to obtain a solution.

Recently a series of works have shown that for suitable measurement models, the nonconvex LS objective (LS-PR) can be globally optimized via gradient descent updates. The Wirtinger flow algorithm is one of the most well-known methods and proposes the framework comprising a spectral initialization followed by gradient descent updates [28]. Spectral initialization ensures that the iterates start in a convex basin near a global optimum when there are enough measurements in an error-free setting. This initialization was first proposed as part of the AltMinPhase algorithm [27]. Multiple works have extended this approach by modifying the initialization, gradient updates and objective for phase retrieval [63, 64], and for other quadratic problems with sensing matrices rather than sensing vectors [65] like the unassigned distance geometry problem [66]. There are also extensions that incorporate signal priors such as sparsity [67, 68]. None of these gradient descent approaches, however, account for sensing vector or sensing matrix errors.

Another group of works have developed convex optimization approaches, which are closely related to low-rank matrix recovery techniques, for solving the phase retrieval problem [69]. These methods use the fact that the measurements in (4.1) can be expressed using the Frobenius matrix inner product, $y_m \approx |\langle \mathbf{a}_m, \mathbf{x} \rangle|^2 = \mathbf{x}^* \mathbf{a}_m \mathbf{a}_m^* \mathbf{x} = \langle \mathbf{a}_m \mathbf{a}_m^*, \mathbf{x} \mathbf{x}^* \rangle$. With this formulation, phase retrieval amounts to recovering a rank-1 positive semidefinite matrix, $\mathbf{X} = \mathbf{x} \mathbf{x}^*$, from linear matrix inner product measurements, $\{\langle \mathbf{a}_m \mathbf{a}_m^*, \mathbf{x} \mathbf{x}^* \rangle\}_{m=1}^M$ [70, 71]. In practice, lifting the problem from recovering vectors in \mathbb{C}^N to matrices in $\mathbb{C}^{N \times N}$ poses significant computational and memory challenges for even moderately sized problems. Matrix sketching algorithms [24] and convex methods which do not require lifting [72] have since been developed to address these challenges.

For linear inverse problems, the TLS method is an established approach for handling errors in both the measurements and the operator [58, 59]. For linear problems, TLS can be efficiently solved using the singular value decomposition (SVD). For the quadratic case considered in this chapter, such an approach is not apparent because of the magnitude-squared nonlinearity in (4.1). We, therefore, also cannot use the SVD to analyze the solution error as is done in the linear case [73]. Linear TLS has been extended to settings with structured operator errors [74, 75], sparse signals [76], and signals with

norm constraints [77]. We note that Yagle and Bell use linear TLS to solve a particular subproblem of a phase retrieval algorithm which only addresses errors in the measurements [78].

There also exist algorithms for nonlinear TLS which aim to solve a general optimization problem for inverse problems with arbitrary nonlinearities [60, 79, 80]. The general optimization problem is similar to (4.3), except for the constraint which requires nonlinear rather than quadratic consistency. However, by using the specific structure of the phase retrieval problem in this chapter (4.1) we are able to obtain efficient algorithms and perform error analysis for TLS phase retrieval.

Our gradient descent strategy uses alternating updates to solve the TLS phase retrieval problem. While alternating updates have been successfully utilized to solve the linear TLS problem [76], it is not straightforward to extend this approach to phase retrieval because of its quadratic nature. We show how to use the geometry of the optimization problem (4.3) to perform alternating updates for TLS phase retrieval.

4.2 TLS for phase retrieval

In this section we show how to solve the TLS phase retrieval problem. Recall (4.3),

$$\begin{aligned} \underset{\substack{\mathbf{x}, \\ r_1, \dots, r_M, \\ \mathbf{e}_1, \dots, \mathbf{e}_M}}{\text{minimize}}}{\quad} & \frac{1}{2M} \sum_{m=1}^M \lambda_y r_m^2 + \lambda_a \|\mathbf{e}_m\|_2^2, \\ \text{s.t.} & \quad y_m + r_m = |\langle \mathbf{a}_m + \mathbf{e}_m, \mathbf{x} \rangle|^2, \quad m = 1, \dots, M, \end{aligned} \quad (4.4)$$

which has been normalized by the number of measurements by the scaling $\frac{1}{M}$. We can rearrange the constraint and substitute $r_m = |\langle \mathbf{a}_m + \mathbf{e}_m, \mathbf{x} \rangle|^2 - y_m$ for $1 \leq m \leq M$ to obtain,

$$\underset{\mathbf{x}}{\text{minimize}} \frac{1}{2M} \sum_{m=1}^M \underset{\mathbf{e}_m}{\text{minimize}} \lambda_a \|\mathbf{e}_m\|_2^2 + \lambda_y (y_m - |\langle \mathbf{a}_m + \mathbf{e}_m, \mathbf{x} \rangle|^2)^2. \quad (4.5)$$

Further we denote the m th corrected sensing vector as $\hat{\mathbf{a}}_m := (\mathbf{a}_m + \mathbf{e}_m)$

to obtain the equivalent formulations

$$\underset{\mathbf{x}}{\text{minimize}} \frac{1}{2M} \sum_{m=1}^M \underbrace{\min_{\hat{\mathbf{a}}_m} \lambda_a \|\mathbf{a}_m - \hat{\mathbf{a}}_m\|_2^2 + \lambda_y (y_m - |\langle \hat{\mathbf{a}}_m, \mathbf{x} \rangle|^2)^2}_{\mathcal{I}_m(\mathbf{x})}, \quad (\text{TLS-PR1})$$

and

$$\underset{\substack{\mathbf{x}, \\ \hat{\mathbf{a}}_1, \dots, \hat{\mathbf{a}}_M}}{\text{minimize}} \underbrace{\frac{1}{2M} \sum_{m=1}^M \lambda_a \|\mathbf{a}_m - \hat{\mathbf{a}}_m\|_2^2 + \lambda_y (y_m - |\langle \hat{\mathbf{a}}_m, \mathbf{x} \rangle|^2)^2}_{\mathcal{J}(\mathbf{x}, \hat{\mathbf{a}}_1, \dots, \hat{\mathbf{a}}_M)}. \quad (\text{TLS-PR2})$$

As each data consistency term, $(y_m - |\langle \hat{\mathbf{a}}_m, \mathbf{x} \rangle|^2)^2$, is proportional to $\|\mathbf{x}\|_2^4$ in an error-free setting, we set $\lambda_y = \frac{\lambda_y^\dagger}{\|\mathbf{x}^{(0)}\|_2^4}$ in order to make the scaling of the objective invariant with respect to the norm of \mathbf{x} . The vector $\mathbf{x}^{(0)}$ is an initial guess for \mathbf{x} and λ_y^\dagger is a regularization parameter. Furthermore, to account for the fact that the sensing vector corrections, $(\mathbf{a}_m - \hat{\mathbf{a}}_m)$, are N -dimensional and the data consistency terms are scalar we set $\lambda_a = \frac{\lambda_a^\dagger}{N}$ where λ_a^\dagger is a regularization parameter.

In line with recent methods such as the Wirtinger flow algorithm [28], our high level strategy is to obtain \mathbf{x} by solving

$$\arg \min_{\mathbf{x}} \frac{1}{2M} \sum_{m=1}^M \mathcal{I}_m(\mathbf{x}), \quad (4.6)$$

using gradient descent. To perform gradient descent with respect to \mathbf{x} we can use Wirtinger gradient updates [28],

$$\mathbf{x}^{(\tau+1)} = \mathbf{x}^{(\tau)} - \frac{\mu}{\|\mathbf{x}^{(0)}\|_2^2} \cdot \frac{1}{2M} \sum_{m=1}^M \nabla_{\mathbf{x}} \mathcal{I}_m(\mathbf{x}^{(\tau)}), \quad (4.7)$$

where μ is the step size and $\|\mathbf{x}^{(0)}\|_2$ is a guess for $\|\mathbf{x}\|_2$. The gradient is given by

$$\nabla_{\mathbf{x}} \mathcal{I}_m(\mathbf{x}) = 2 \left(\left| \langle \hat{\mathbf{a}}_m^\dagger, \mathbf{x} \rangle \right|^2 - y_m \right) \hat{\mathbf{a}}_m^\dagger \hat{\mathbf{a}}_m^{\dagger*} \mathbf{x}, \quad (4.8)$$

where $\hat{\mathbf{a}}_m^\dagger$ is the solution to the following nonconvex optimization problem

that is seen in (TLS-PR1)

$$\hat{\mathbf{a}}_m^\dagger = \arg \min_{\mathbf{a}} \lambda_a \|\mathbf{a}_m - \mathbf{a}\|_2^2 + \lambda_y (y_m - |\langle \mathbf{a}, \mathbf{x} \rangle|^2)^2. \quad (4.9)$$

This motivates the following alternating updates procedure to solve the TLS problem:

1. Obtain an initial guess, $\mathbf{x}^{(0)} \in \mathbb{C}^N$, for \mathbf{x} .
2. Repeat steps 2a and 2b until convergence:
 - (a) With \mathbf{x} fixed, obtain corrected sensing vectors, $\{\hat{\mathbf{a}}_m^\dagger\}_{m=1}^M$, by solving (4.9) for $1 \leq m \leq M$.
 - (b) With $\{\hat{\mathbf{a}}_m^\dagger\}_{m=1}^M$ fixed, take one gradient descent step to update \mathbf{x} using (4.7).

The main challenge in our approach is obtaining corrected sensing vectors $\{\hat{\mathbf{a}}_m^\dagger\}_{m=1}^M$ by solving (4.9) so that we can perform gradient descent updates for \mathbf{x} using (4.7). As (TLS-PR2) is nonconvex, a good initial guess, $\mathbf{x}^{(0)}$, can place us near a global minimum. There are multiple initialization options, such as the spectral initialization for certain measurement models [27].

In the remainder of this section, we will examine the geometry of the optimization problem in (TLS-PR1) and show how it can be leveraged to efficiently solve (4.9) and obtain corrected sensing vectors. This is summarized by Proposition 2 below. We will then present the complete TLS phase retrieval algorithm. Lastly, we also interpret the regularization parameters, λ_a and λ_y , by showing that the TLS solution is the maximum likelihood estimator for a quadratic complex-valued error-in-variables (EIV) model.

4.2.1 Optimization geometry

Moving from the LS formulation to the TLS formulation introduces significant computational issues. In addition to optimizing over vector \mathbf{x} , we must additionally optimize over M sensing vectors in (TLS-PR1), with typically $M > N$. We now study the optimization geometry of (TLS-PR1) and show that the M inner minimizations over the N -dimensional vectors, $\{\hat{\mathbf{a}}_m\}_{m=1}^M$, can be simplified to minimizing over M scalars which improves efficiency.

For ease of visualization in this subsection, we consider the real-valued problem (all quantities in (4.1), (LS-PR) and (TLS-PR1) are real) and we set $\lambda_a = \lambda_y = 1$.

For a given vector \mathbf{x} we compare the values of the LS and TLS objectives, (LS-PR) and (TLS-PR1). The left column of Figure 4.1 visualizes the phase retrieval problem with $M = 5$ data points, $\{(\mathbf{a}_m, y_m)\}_{m=1}^M$, when $N = 2$ and $\|\mathbf{x}\|_2 = 1$. The middle column shows the same data points from a different viewing angle. In phase retrieval we fit a paraboloid, $y(\mathbf{a}) = |\langle \mathbf{a}, \mathbf{x} \rangle|^2$ that is parameterized by \mathbf{x} to the data points, $\{(\mathbf{a}_m, y_m)\}_{m=1}^M$. If there is no sensing vector or measurement error, the data points lie on the paraboloid ((4.1) holds with equality). The left and middle figure show that the surface $y(\mathbf{a}) = |\langle \mathbf{a}, \mathbf{x} \rangle|^2$ does not change in the subspace perpendicular to \mathbf{x} , denoted as \mathbf{x}^\perp . This can also be verified by considering the values of \mathbf{a} that would result in the inner product $\langle \mathbf{a}, \mathbf{x} \rangle$ being zero. Crucially, this means that the shortest paths between the data points and the paraboloid have no component in the \mathbf{x}^\perp subspace. As a result, we can view the problem in 2D from a viewpoint that looks into the \mathbf{x}^\perp subspace as shown in the right column of Figure 4.1. This 2D plot shows two options for measuring closeness between the surface and the data points. The LS objective (LS-PR), is the sum of the squared vertical distance between the 2D parabola and each data point as indicated by the dashed lines. On the other hand, due to the minima over all $\hat{\mathbf{a}}_m$, the TLS objective (TLS-PR1), is the sum of the squared Euclidean or orthogonal distance between the 2D parabola and each data point as shown by the solid lines. A similar geometrical interpretation is seen with linear TLS [58, 59].

Considering this geometry, to solve the inner minimizations in (TLS-PR1), we find the closest point on the paraboloid to each data point. As the shortest path has no component in the \mathbf{x}^\perp subspace, our task of finding the closest point on a $(N+1)$ -dimensional paraboloid to a $(N+1)$ -dimensional data point reduces to a 2D geometry problem of finding the closest point on a parabola to a 2D data point. Rather than finding the minimizing N -dimensional $\hat{\mathbf{a}}_m^\dagger$ for each data point, we instead only need to find the component of $\hat{\mathbf{a}}_m^\dagger$ in the \mathbf{x} direction that is closest. This component is a scalar and is given by the inner product, $\nu_m = \langle \hat{\mathbf{a}}_m^\dagger, \mathbf{x} \rangle$. We can then construct $\hat{\mathbf{a}}_m^\dagger$ by adding the

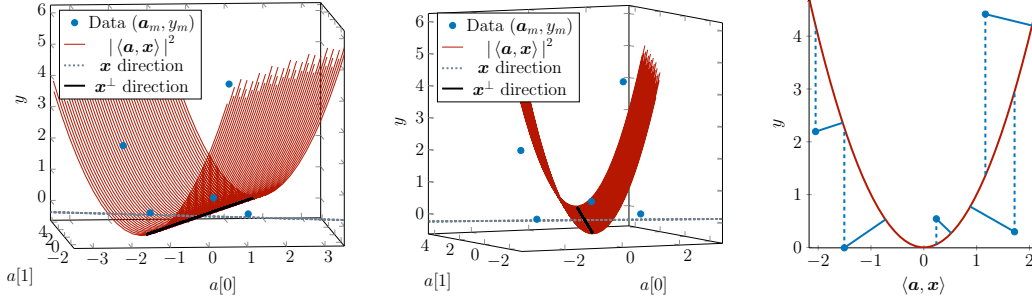


Figure 4.1: Visualization of the phase retrieval problem when $\|\mathbf{x}\|_2 = 1$. The left column shows $M = 5$ data points, $\{(\mathbf{a}_m, y_m)\}_{m=1}^M$, when $N = 2$. A paraboloid is fitted to the data points. The middle column shows the same paraboloid and data points from a different viewing angle. The right column shows the problem from a viewpoint that looks into the \mathbf{x}^\perp subspace. The dashed lines show the distances minimized by the LS objective, (LS-PR). The solid lines show the distances minimized by the TLS objective, (TLS-PR1).

unchanged component in the \mathbf{x}^\perp subspace,

$$\hat{\mathbf{a}}_m^\dagger := \hat{\mathbf{a}}_m^\dagger(\nu_m) = \frac{\nu_m}{\|\mathbf{x}\|_2} \hat{\mathbf{x}} + (\mathbf{a}_m - \langle \mathbf{a}_m, \hat{\mathbf{x}} \rangle \hat{\mathbf{x}}), \quad (4.10)$$

where $\hat{\mathbf{x}}$ is \mathbf{x} normalized.

If λ_a and λ_y are not one, a perpendicular distance is not minimized. As $\frac{\lambda_a}{\lambda_y}$ gets larger, the solid lines in the right column of Figure 4.1 become more vertical because there is a relatively larger penalty for correcting the sensing vectors and the problem moves towards a LS approach. Conversely, the lines become more horizontal as $\frac{\lambda_a}{\lambda_y}$ gets smaller. Irrespective of the values of λ_a and λ_y , the shortest paths between the paraboloid and the data points still have no component in the \mathbf{x}^\perp subspace and (4.10) can be used to obtain each $\hat{\mathbf{a}}_m^\dagger$. We further note that this geometry also holds for the complex-valued phase retrieval problem (4.1).

4.2.2 Correcting complex-valued sensing vectors

Our strategy is to set up each inner minimization over $\hat{\mathbf{a}}_m$ in (TLS-PR1) as the minimization of a fourth degree equation with respect to scalar $\nu_m = \langle \hat{\mathbf{a}}_m, \mathbf{x} \rangle$ rather than vector $\hat{\mathbf{a}}_m$. We then directly obtain the minimizer of this equation.

The M inner minimization problems in (TLS-PR1) are independent of each other and we can independently solve each summand for a fixed vector \mathbf{x} . Consider the objective function of optimization problem $\mathcal{I}_m(\mathbf{x})$,

$$f_m(\widehat{\mathbf{a}}_m) = \lambda_a \|\mathbf{a}_m - \widehat{\mathbf{a}}_m\|_2^2 + \lambda_y (y_m - |\mathbf{x}^* \widehat{\mathbf{a}}_m|^2)^2. \quad (4.11)$$

Proposition 2 states that $\arg \min_{\widehat{\mathbf{a}}_m} f_m(\widehat{\mathbf{a}}_m)$ can be obtained by solving two scalar variable cubic equations and using (4.10).

Proposition 2. *Let sets R_+ and R_- be the positive real solutions of*

$$\alpha r^3 + \beta r \pm |\gamma| = 0 \quad (4.12)$$

where $\alpha = 2\lambda_y \|\mathbf{x}\|_2^2 \in \mathbb{R}$, $\beta = \lambda_a - 2\lambda_y y_m \|\mathbf{x}\|_2^2 \in \mathbb{R}$ and $\gamma = -\lambda_a \mathbf{x}^* \mathbf{a}_m \in \mathbb{C}$. Further, with κ denoting the phase of γ , let

$$S_+ = \{e^{j\kappa} r \mid r \in R_+\} \text{ and } S_- = \{-e^{j\kappa} r \mid r \in R_-\}. \quad (4.13)$$

Then $f_m(\widehat{\mathbf{a}}_m)$ is minimized by $\widehat{\mathbf{a}}_m^\dagger(s^\dagger)$ where

$$s^\dagger = \arg \min_{s \in S_+ \cup S_-} f_m(\widehat{\mathbf{a}}_m^\dagger(s)) \quad (4.14)$$

and $\widehat{\mathbf{a}}_m^\dagger(\cdot)$ is defined in (4.10).

Proof. Expanding $f_m(\widehat{\mathbf{a}}_m)$ gives

$$\begin{aligned} f_m(\widehat{\mathbf{a}}_m) &= \lambda_a (\|\mathbf{a}_m\|_2^2 - \mathbf{a}_m^* \widehat{\mathbf{a}}_m - \widehat{\mathbf{a}}_m^* \mathbf{a}_m + \widehat{\mathbf{a}}_m^* \widehat{\mathbf{a}}_m) \\ &\quad + \lambda_y (y_m^2 - 2y_m \widehat{\mathbf{a}}_m^* \mathbf{x} \mathbf{x}^* \widehat{\mathbf{a}}_m + (\widehat{\mathbf{a}}_m^* \mathbf{x} \mathbf{x}^* \widehat{\mathbf{a}}_m)^2). \end{aligned} \quad (4.15)$$

We can use Wirtinger derivatives to calculate the derivative of the real-valued $f_m(\widehat{\mathbf{a}}_m)$ with respect to the complex vector $\widehat{\mathbf{a}}_m$ [28],

$$\begin{aligned} \nabla_{\widehat{\mathbf{a}}_m} f_m &= (\lambda_a (-\mathbf{a}_m^* + \widehat{\mathbf{a}}_m^*) + \lambda_y (-2y_m \widehat{\mathbf{a}}_m^* \mathbf{x} \mathbf{x}^* + 2|\mathbf{x}^* \widehat{\mathbf{a}}_m|^2 \widehat{\mathbf{a}}_m^* \mathbf{x} \mathbf{x}^*))^* \\ &= \lambda_a (\widehat{\mathbf{a}}_m - \mathbf{a}_m) + \lambda_y (-2y_m \mathbf{x} \mathbf{x}^* \widehat{\mathbf{a}}_m + 2|\mathbf{x}^* \widehat{\mathbf{a}}_m|^2 \mathbf{x} \mathbf{x}^* \widehat{\mathbf{a}}_m). \end{aligned} \quad (4.16)$$

Setting the derivative to zero, and then left-multiplying by nonzero \mathbf{x}^*

gives

$$2\lambda_y \|\mathbf{x}\|_2^2 |\mathbf{x}^* \hat{\mathbf{a}}_m|^2 (\mathbf{x}^* \hat{\mathbf{a}}_m) + (\lambda_a - 2\lambda_y y_m \|\mathbf{x}\|_2^2) (\mathbf{x}^* \hat{\mathbf{a}}_m) - \lambda_a \mathbf{x}^* \mathbf{a}_m = 0. \quad (4.17)$$

The left hand side is now scalar-valued and is a function of scalar $\nu_m = \langle \hat{\mathbf{a}}_m, \mathbf{x} \rangle = \mathbf{x}^* \hat{\mathbf{a}}_m \in \mathbb{C}$ instead of a vector. Recalling our analysis of the optimization geometry in Section 4.2.1, we can solve for ν_m and then obtain $\hat{\mathbf{a}}_m^\dagger = \hat{\mathbf{a}}_m^\dagger(\nu_m)$ using (4.10). If we substitute $\alpha = 2\lambda_y \|\mathbf{x}\|_2^2 \in \mathbb{R}$, $\beta = \lambda_a - 2\lambda_y y_m \|\mathbf{x}\|_2^2 \in \mathbb{R}$ and $\gamma = -\lambda_a \mathbf{x}^* \mathbf{a}_m \in \mathbb{C}$ we wish to solve the following for ν_m ,

$$\alpha |\nu_m|^2 \nu_m + \beta \nu_m + \gamma = 0. \quad (4.18)$$

Because the sensing vectors and ground truth signal are complex, this cubic equation is a function of $\nu_m \in \mathbb{C}$ and its conjugate $\bar{\nu}_m$ ($|\nu_m|^2 = \nu_m \bar{\nu}_m$). We therefore cannot use standard cubic root finding formulae. Further note that the coefficients α and β are always real and γ may be complex. To solve, first multiply by $\bar{\nu}_m$,

$$\alpha |\nu_m|^4 + \beta |\nu_m|^2 + \gamma \bar{\nu}_m = 0. \quad (4.19)$$

Next, with complex-exponential representation, $\nu_m = r e^{j\phi}$ and $\gamma = |\gamma| e^{j\kappa}$ (recall γ is known), the equation becomes

$$\alpha r^3 + \beta r + |\gamma| e^{j(\kappa - \phi)} = 0. \quad (4.20)$$

The real and imaginary parts of the left hand side should both equate to zero. Using Euler's identity, $e^{j\theta} = \cos(\theta) + j \sin(\theta)$, we arrive at the following simultaneous equations,

$$\begin{cases} \sin(\kappa - \phi) = 0 \\ \alpha r^3 + \beta r + |\gamma| \cos(\kappa - \phi) = 0. \end{cases} \quad (4.21)$$

For the first equation to hold, $\cos(\kappa - \phi) = \pm 1$ and so the phase of ν_m has two possible values; $\phi = \kappa$ or $\phi = (\kappa - \pi)$. To obtain the magnitude of ν_m we can solve the following two cubic equations for r to get six values, three

from each,

$$\alpha r^3 + \beta r + |\gamma| = 0 \quad \text{and} \quad \phi = \kappa \quad (4.22)$$

$$\alpha r^3 + \beta r - |\gamma| = 0 \quad \text{and} \quad \phi = \kappa - \pi. \quad (4.23)$$

As the solutions of these two cubic equations are magnitudes of complex numbers, we let sets R_+ and R_- be the positive real solutions of (4.22) and (4.23) respectively. To obtain values for ν_m we combine R_+ and R_- with their phases to get S_+ and S_- —multiply the elements of R_+ by $e^{j\kappa}$ and multiply the elements of R_- by $e^{j(\kappa-\pi)} = -e^{j\kappa}$. We then construct candidate minimizers of $f_m(\cdot)$ by using the possible values for ν_m , the set, $S_+ \cup S_-$, as the argument for (4.10). Finally, the global minimizer is the candidate minimizer that gives the minimum value as the argument of $f_m(\cdot)$. \square

To solve (4.22) and (4.23) for r , Cardano’s formula for cubic equations or a general cubic root formula derived from Cardano’s formula can be used (see Appendix B.1). Furthermore, we note that this procedure to update the sensing vectors is independent of the sensing vector measurement model.

4.2.3 TLS phase retrieval algorithm

Now that we have a method for solving the inner minimizations in (TLS-PR1), we present the complete TLS phase retrieval algorithm in Algorithm 2. We say that the algorithm has converged if the value of $\mathcal{J}(\mathbf{x}, \hat{\mathbf{a}}_1^\dagger, \dots, \hat{\mathbf{a}}_M^\dagger)$ in (TLS-PR2) between consecutive iterates is less than some threshold. In practice all sensing vectors can be updated (lines 5-16 of Algorithm 2) in parallel for a given \mathbf{x} because all sensing vectors are independent of one another.

4.2.4 ML estimator for EIV models

Proposition 3 below provides an interpretation of the regularization parameters in (TLS-PR1) by connecting them to the error level. It states that under certain assumptions the solution to (TLS-PR1) is the maximum likelihood

Algorithm 2 TLS phase retrieval.

Input: Erroneous sensing vectors $\{\mathbf{a}_m\}_{m=1}^M$; Erroneous observations $\{y_m\}_{m=1}^M$; Threshold T ; Regularization parameters λ_y and λ_a .

Output: Recovered signal $\mathbf{x} \in \mathbb{C}^N$.

```

1:  $\mathbf{x} \leftarrow \text{Initialization}(y_1, \dots, y_M, \mathbf{a}_1, \dots, \mathbf{a}_M)$ 
2:  $\text{loss\_previous} \leftarrow -\infty$ 
3:  $\text{loss\_current} \leftarrow \infty$ 
4: while  $|\text{loss\_current} - \text{loss\_previous}| > T$  do
    // Update each sensing vector for a given  $\mathbf{x}$ 
5:   for each  $m \in \{1, \dots, M\}$  do
6:      $\alpha \leftarrow 2\lambda_y \|\mathbf{x}\|_2^2$ 
7:      $\beta \leftarrow \lambda_a - 2\lambda_y y_m \|\mathbf{x}\|_2^2$ 
8:      $\gamma \leftarrow -\lambda_a \mathbf{x}^* \mathbf{a}_m$ 
9:      $\kappa \leftarrow \text{Angle}(\gamma)$ 
10:     $R_+ \leftarrow \text{PositiveRealRoots}(\alpha r^3 + \beta r + |\gamma|)$ 
11:     $R_- \leftarrow \text{PositiveRealRoots}(\alpha r^3 + \beta r - |\gamma|)$ 
12:     $S_+ \leftarrow e^{j\kappa} \cdot R_+$ 
13:     $S_- \leftarrow -e^{j\kappa} \cdot R_-$ 
14:     $s^\dagger = \arg \min_{s \in S_+ \cup S_-} f_m(\widehat{\mathbf{a}}_m^\dagger(s))$ 
15:     $\widehat{\mathbf{a}}_m^\dagger \leftarrow \widehat{\mathbf{a}}_m^\dagger(s^\dagger)$ 
16:   end for
    // Update  $\mathbf{x}$  with sensing vectors fixed
17:    $\mathbf{x} \leftarrow \text{x\_gradient\_step}(\mathbf{x}, \widehat{\mathbf{a}}_1^\dagger, \dots, \widehat{\mathbf{a}}_M^\dagger)$ 
18:    $\text{loss\_previous} \leftarrow \text{loss\_current}$ 
19:    $\text{loss\_current} \leftarrow \mathcal{J}(\mathbf{x}, \widehat{\mathbf{a}}_1^\dagger, \dots, \widehat{\mathbf{a}}_M^\dagger)$ 
20: end while

```

(ML) estimator for the complex-valued EIV model given by

$$y_m = |\langle \widetilde{\mathbf{a}}_m, \widetilde{\mathbf{x}} \rangle|^2 + (-\eta_m), \quad \mathbf{a}_m = \widetilde{\mathbf{a}}_m + (-\boldsymbol{\delta}_m) \quad (4.24)$$

for $1 \leq m \leq M$. With this EIV model we aim to recover $\widetilde{\mathbf{x}}$ and $\{\widetilde{\mathbf{a}}_m\}_{m=1}^M$ from $\{y_m\}_{m=1}^M$ and $\{\mathbf{a}_m\}_{m=1}^M$ which are known. The quantities $\{\eta_m\}_{m=1}^M$ and $\{\boldsymbol{\delta}_m\}_{m=1}^M$ are random error perturbations. This result is an extension of the relationship between linear TLS and the linear EIV model [74, 81]. Similarly, this result is a specific instance of what is seen for nonlinear TLS [60].

Proposition 3. *Assume in (4.24) that $\{\eta_m\}_{m=1}^M$ are iid zero-mean Gaussian with covariance $\sigma_\eta^2 \mathbf{I}$, $\{\boldsymbol{\delta}_m\}_{m=1}^M$ are independent of each other and each is an iid zero-mean complex Gaussian vector with covariance $2\sigma_\delta^2$, i.e. $\text{vec}([\text{Re}(\boldsymbol{\delta}_m) | \text{Im}(\boldsymbol{\delta}_m)]) \sim \mathcal{N}(\mathbf{0}, \sigma_\delta^2 \mathbf{I})$. Further assume that $\{\eta_m\}_{m=1}^M$ and*

$\{\boldsymbol{\delta}_m\}_{m=1}^M$ are independent of each other and that $\{\tilde{\mathbf{a}}_m\}_{m=1}^M$ and $\tilde{\mathbf{x}}$ are deterministic. Under these assumptions, the solution to optimization problem (TLS-PR1), when $\lambda_a = \frac{1}{\sigma_\delta^2}$ and $\lambda_y = \frac{1}{\sigma_n^2}$, is the maximum likelihood estimator for (4.24).

Proof. The proof follows a standard procedure and is provided in Appendix B.2. \square

4.3 TLS and LS solution reconstruction errors

In this section, we evaluate the reconstruction error for the TLS and LS phase retrieval solutions by deriving their Taylor expansions. Through these expressions, we are able to gain insight into the behavior of the TLS solution relative to the LS solution and understand when each method performs well. We also use these expressions to understand how the reconstruction errors rely on the level of the measurement and the sensing vector errors when all the errors are Gaussian. Since this analysis is cumbersome, in this section we will consider the real-valued phase retrieval problem where the ground truth signal, the sensing vectors and the sensing vector errors in (4.1) are real. Simulations in Section 4.4 show that the reasoning carries through to the complex problem. In our derivations, we will use theorems about differentiation of argmins and various matrix inversion lemmas.

We denote the ground truth signal as $\mathbf{x}^\#$ and the TLS and LS solutions as $\mathbf{x}_{\text{TLS}}^\dagger$ and $\mathbf{x}_{\text{LS}}^\dagger$. If there are no errors in the sensing vectors or measurements, $\mathbf{x}^\#$ and $-\mathbf{x}^\#$ are both optimum LS and TLS solutions for (LS-PR) and (TLS-PR2) (with the m th corrected sensing vector being \mathbf{a}_m). Due to this inherent sign ambiguity it is standard to define the reconstruction errors as

$$\min_{\sigma} \left\| \mathbf{x}^\# - \sigma \cdot \mathbf{x}_{\text{TLS}}^\dagger \right\|_2 \quad \text{and} \quad \min_{\sigma} \left\| \mathbf{x}^\# - \sigma \cdot \mathbf{x}_{\text{LS}}^\dagger \right\|_2 \quad (4.25)$$

where $\sigma \in \{1, -1\}$. Our results are unchanged if the analysis is done with optimum solution $\mathbf{x}^\#$ ($\sigma = 1$) or with optimum solution $-\mathbf{x}^\#$ ($\sigma = -1$). Consequently, we choose optimum solution $\mathbf{x}^\#$ with $\sigma = 1$ in the following analysis.

4.3.1 Reconstruction error analysis

The erroneous sensing vectors and measurements in (4.1) can be expressed as perturbed versions of error-free sensing vectors and measurements, $\{\tilde{\mathbf{a}}_m\}_{m=1}^M$ and $\{\tilde{y}_m\}_{m=1}^M$. We denote the sensing vector and measurement error perturbations as $\{\boldsymbol{\delta}_m\}_{m=1}^M$ and $\{\eta_m\}_{m=1}^M$. Stacking these into vectors we define,

$$\tilde{\mathbf{t}} = [\tilde{\mathbf{a}}_1^T, \dots, \tilde{\mathbf{a}}_M^T, \tilde{y}_1, \dots, \tilde{y}_M]^T \in \mathbb{R}^{(MN+M)}, \quad (4.26)$$

$$\boldsymbol{\gamma} = [\boldsymbol{\delta}_1^T, \dots, \boldsymbol{\delta}_M^T, \eta_1, \dots, \eta_M]^T \in \mathbb{R}^{(MN+M)}, \quad (4.27)$$

$$\begin{aligned} \mathbf{t} &= \tilde{\mathbf{t}} + \boldsymbol{\gamma} \\ &= [\mathbf{a}_1^T, \dots, \mathbf{a}_M^T, y_1, \dots, y_M]^T \in \mathbb{R}^{(MN+M)}. \end{aligned} \quad (4.28)$$

In order to calculate the reconstruction errors, we need access to expressions for $\mathbf{x}_{\text{TLS}}^\dagger$ and $\mathbf{x}_{\text{LS}}^\dagger$. We begin by noting that the solutions are functions of the sensing vectors and measurements, $\mathbf{x}_{\text{TLS}}^\dagger(\mathbf{t})$ and $\mathbf{x}_{\text{LS}}^\dagger(\mathbf{t})$ (an application of (1.2)). If there are no errors in the sensing vectors or measurements, an optimum LS solution for (LS-PR) is $\mathbf{x}_{\text{LS}}^\dagger(\tilde{\mathbf{t}}) = \mathbf{x}^\#$. Similarly, an optimum TLS solution in (TLS-PR2) for $\mathbf{x}_{\text{TLS}}^\dagger(\tilde{\mathbf{t}}) = \mathbf{x}^\#$ with the m th corrected sensing vector being \mathbf{a}_m (no correction needed). Now, if we instead have sensing vector and measurement errors, our solutions are $\mathbf{x}_{\text{TLS}}^\dagger(\tilde{\mathbf{t}} + \boldsymbol{\gamma})$ and $\mathbf{x}_{\text{LS}}^\dagger(\tilde{\mathbf{t}} + \boldsymbol{\gamma})$ which we can interpret as perturbed versions of $\mathbf{x}_{\text{LS}}^\dagger(\tilde{\mathbf{t}}) = \mathbf{x}_{\text{TLS}}^\dagger(\tilde{\mathbf{t}}) = \mathbf{x}^\#$. Assuming $\|\boldsymbol{\gamma}\|$ is small, we can study the first-order terms in the Taylor series expansions of $\mathbf{x}_{\text{TLS}}^\dagger(\mathbf{t})$ and $\mathbf{x}_{\text{LS}}^\dagger(\mathbf{t})$ to measure the perturbation from $\mathbf{x}^\#$.

The Taylor series expansion of $\mathbf{x}_{\text{TLS}}^\dagger(\mathbf{t}) = \mathbf{x}_{\text{TLS}}^\dagger(\tilde{\mathbf{t}} + \boldsymbol{\gamma})$ at the no error point, $\tilde{\mathbf{t}}$, is

$$\begin{aligned} \mathbf{x}_{\text{TLS}}^\dagger(\tilde{\mathbf{t}} + \boldsymbol{\gamma}) &= \mathbf{x}_{\text{TLS}}^\dagger(\tilde{\mathbf{t}}) + \nabla_{\mathbf{t}} \mathbf{x}_{\text{TLS}}^\dagger(\mathbf{t})|_{\mathbf{t}=\tilde{\mathbf{t}}} \boldsymbol{\gamma} + \mathcal{O}(\|\boldsymbol{\gamma}\|_2^2) \\ &= \mathbf{x}^\# + \nabla_{\mathbf{t}} \mathbf{x}_{\text{TLS}}^\dagger(\mathbf{t})|_{\mathbf{t}=\tilde{\mathbf{t}}} \boldsymbol{\gamma} + \mathcal{O}(\|\boldsymbol{\gamma}\|_2^2), \end{aligned} \quad (4.29)$$

where $\mathcal{O}(\|\boldsymbol{\gamma}\|_2^2)$ represents terms with norm of order $\|\boldsymbol{\gamma}\|_2^2$. The Taylor series expansion for $\mathbf{x}_{\text{LS}}^\dagger(\mathbf{t})$ can be written similarly. Using these expansions, to the first-order when $\|\boldsymbol{\gamma}\|$ is small, the reconstruction errors for the TLS and LS

problems are

$$e_{\text{TLS}} := \left\| \nabla_{\mathbf{t}} \mathbf{x}_{\text{TLS}}^\dagger(\mathbf{t}) \Big|_{\mathbf{t}=\tilde{\mathbf{t}}} \boldsymbol{\gamma} \right\|_2 \quad (4.30)$$

$$e_{\text{LS}} := \left\| \nabla_{\mathbf{t}} \mathbf{x}_{\text{LS}}^\dagger(\mathbf{t}) \Big|_{\mathbf{t}=\tilde{\mathbf{t}}} \boldsymbol{\gamma} \right\|_2 \quad (4.31)$$

To evaluate e_{TLS} and e_{LS} we must calculate the derivatives $\nabla_{\mathbf{t}} \mathbf{x}_{\text{TLS}}^\dagger(\mathbf{t}) \in \mathbb{R}^{N \times (MN+M)}$ and $\nabla_{\mathbf{t}} \mathbf{x}_{\text{LS}}^\dagger(\mathbf{t}) \in \mathbb{R}^{N \times (MN+M)}$ which are the derivatives of the argmins of (TLS-PR2) and (LS-PR). We use the method by Gould et al. to take derivatives of argmin problems [82].

With the substitution $\mathbf{e}_m = \hat{\mathbf{a}}_m - \mathbf{a}_m$ and multiplicative constants absorbed into λ_a and λ_y , the TLS optimization problem (TLS-PR2) can be rewritten as

$$\begin{aligned} \mathbf{q}^\dagger = \arg \min_{\mathbf{q}} \underbrace{\sum_{m=1}^M \lambda_a \|\mathbf{e}_m\|_2^2 + \lambda_y (y_m - |\langle \mathbf{a}_m + \mathbf{e}_m, \mathbf{x} \rangle|)^2}_{f(\mathbf{q}, \mathbf{t})} \\ \text{s.t. } \mathbf{q} = \begin{bmatrix} \mathbf{e}_1^T & \cdots & \mathbf{e}_M^T & \mathbf{x}^T \end{bmatrix}^T \in \mathbb{R}^{MN+N}. \end{aligned} \quad (4.32)$$

The solution, $g(\mathbf{t}) := \mathbf{q}^\dagger$, is a function of \mathbf{t} and $\mathbf{x}_{\text{TLS}}^\dagger(\mathbf{t})$ is the last N entries of $g(\mathbf{t})$, denoted as $g(\mathbf{t})_{-N}$,

$$g(\mathbf{t}) := \mathbf{q}^\dagger = \arg \min_{\mathbf{q}} f(\mathbf{q}, \mathbf{t}) \in \mathbb{R}^{MN+N}, \quad (4.33)$$

$$\mathbf{x}_{\text{TLS}}^\dagger(\mathbf{t}) = g(\mathbf{t})_{-N} \in \mathbb{R}^N. \quad (4.34)$$

The derivatives of $g(\mathbf{t})$ with respect to the k th sensing vector and measurement can be computed after specific second derivatives of $f(\mathbf{q}, \mathbf{t})$ are computed [82],

$$\nabla_{\mathbf{a}_k} g(\mathbf{t}) = -(\nabla_{\mathbf{q}\mathbf{q}}^2 f(\mathbf{q}, \mathbf{t}))^{-1} (\nabla_{\mathbf{a}_k \mathbf{q}}^2 f(\mathbf{q}, \mathbf{t})) \in \mathbb{R}^{(MN+N) \times N}. \quad (4.35)$$

$$\frac{d}{dy_k} g(\mathbf{t}) = -(\nabla_{\mathbf{q}\mathbf{q}}^2 f(\mathbf{q}, \mathbf{t}))^{-1} \left(\frac{d}{dy_k} \nabla_{\mathbf{q}} f(\mathbf{q}, \mathbf{t}) \right) \in \mathbb{R}^{MN+N}. \quad (4.36)$$

We can then obtain $\nabla_{\mathbf{t}} \mathbf{x}_{\text{TLS}}^\dagger(\mathbf{t})$ by vertically stacking the derivatives (4.35)

and (4.36) for $1 \leq k \leq M$ to form $\nabla_{\mathbf{t}}g(\mathbf{t}) \in \mathbb{R}^{(MN+N) \times (MN+M)}$,

$$\nabla_{\mathbf{t}}g(\mathbf{t}) = \left[\nabla_{\mathbf{a}_1}g(\mathbf{t}), \dots, \nabla_{\mathbf{a}_M}g(\mathbf{t}), \frac{d}{dy_1}g(\mathbf{t}), \dots, \frac{d}{dy_M}g(\mathbf{t}) \right], \quad (4.37)$$

and taking the last N rows. Appendix B.3.1 contains the derivations for the last N rows of (4.35) and (4.36).

The same approach can be used for the LS problem by considering its optimization problem,

$$\mathbf{x}_{\text{LS}}^\dagger = \arg \min_{\mathbf{x}} \sum_{m=1}^M (y_m - |\langle \mathbf{a}_m, \mathbf{x} \rangle|^2)^2. \quad (4.38)$$

The corresponding derivative derivations are in Appendix B.3.2.

Proposition 4 below states the expressions for e_{TLS} and e_{LS} . We denote

$$\tilde{\mathbf{Y}} = \text{diag}(\tilde{y}_1, \dots, \tilde{y}_M) \in \mathbb{R}^{M \times M} \quad (4.39)$$

$$\tilde{\mathbf{A}} = \begin{bmatrix} -\tilde{\mathbf{a}}_1^T & - \\ \vdots & \\ -\tilde{\mathbf{a}}_M^T & - \end{bmatrix} \in \mathbb{R}^{M \times N} \quad (4.40)$$

$$\mathbf{E}_Y = \text{diag}(\eta_1, \dots, \eta_M) \in \mathbb{R}^{M \times M} \quad (4.41)$$

$$\mathbf{E}_A = \begin{bmatrix} -\boldsymbol{\delta}_1^T & - \\ \vdots & \\ -\boldsymbol{\delta}_M^T & - \end{bmatrix} \in \mathbb{R}^{M \times N} \quad (4.42)$$

and use these quantities to define diagonal matrix, \mathbf{D} , and vector, \mathbf{w} ,

$$\mathbf{D} = \left(\mathbf{I}_M + 4 \frac{\lambda_y}{\lambda_a} \|\mathbf{x}^\#\|_2^2 \tilde{\mathbf{Y}} \right)^{-1} \in \mathbb{R}^{M \times M} \quad (4.43)$$

$$\mathbf{w} = \left((2\tilde{\mathbf{Y}})^{-1} \mathbf{E}_Y \tilde{\mathbf{A}} - \mathbf{E}_A \right) \mathbf{x}^\# \in \mathbb{R}^M. \quad (4.44)$$

Proposition 4. *To the first-order, the reconstruction errors for the solution $\mathbf{x}_{\text{TLS}}^\dagger$ to the TLS optimization problem (4.32), and, the solution $\mathbf{x}_{\text{LS}}^\dagger$ to the*

LS optimization problem (4.38) are

$$e_{\text{TLS}} = \left\| \left(\tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \mathbf{D} \tilde{\mathbf{A}} \right)^{-1} \tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \mathbf{D} \mathbf{w} \right\|_2 \quad (4.45)$$

$$e_{\text{LS}} = \left\| \left(\tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \tilde{\mathbf{A}} \right)^{-1} \tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \mathbf{w} \right\|_2. \quad (4.46)$$

Proof. Lemma 1 in Appendix B.3 states the Taylor series expansions around the no error point, $\tilde{\mathbf{t}}$, for the TLS and LS solutions. The result in this proposition follows by considering only the zeroth and first-order terms. \square

As expected, when $\gamma \rightarrow 0$, the errors \mathbf{E}_A and \mathbf{E}_Y tend to zero, which makes the vector \mathbf{w} zero and the reconstruction errors are zero. The difference between the TLS and LS reconstruction errors in Proposition 4 is due to the diagonal matrix \mathbf{D} . As $\frac{\lambda_y}{\lambda_a} \rightarrow 0$, $\mathbf{D} \rightarrow \mathbf{I}_M$ and $e_{\text{TLS}} \rightarrow e_{\text{LS}}$. This is because the relative weighting of the sensing error consistency terms in (TLS-PR2), $\|\mathbf{a}_m - \hat{\mathbf{a}}_m\|_2^2$ for all m , increases which makes modifying the sensing vectors increasingly costly and the TLS problem moves closer to the LS problem. Additionally, there are also error models under which the reconstruction errors are equal. For example, if $\mathbf{E}_Y = r_y \tilde{\mathbf{Y}}$ and $\mathbf{E}_A = r_A \tilde{\mathbf{A}}$ where $r_y, r_A \in \mathbb{R}$.

Furthermore, if $M = N$ and $\tilde{\mathbf{A}}$ is invertible, we can again have $e_{\text{TLS}} = e_{\text{LS}}$. However, having $M = N$ is not a practical setting for the real-valued phase retrieval problem because the map from $\mathbf{x}^\#$ to $\left[\langle \tilde{\mathbf{a}}_1, \mathbf{x}^\# \rangle^2, \dots, \langle \tilde{\mathbf{a}}_m, \mathbf{x}^\# \rangle^2 \right]^T$ is not injective, even after accounting for the sign ambiguity [83]. The same holds for the complex-valued phase retrieval problem, even after accounting for the global phase shift [84]. Therefore, we can expect to require more measurements to obtain a unique solution to the phase retrieval problem with the TLS framework.

The reconstruction errors in Proposition 4 can be further interpreted by assuming a distribution for the measurement and sensing vectors errors; in Proposition 5 we assume that the nonzero entries of \mathbf{E}_Y and \mathbf{E}_A are iid zero-mean Gaussian (with different variances for \mathbf{E}_Y and \mathbf{E}_A).

Proposition 5. *With the setting of Proposition 4, assume that the diagonal elements of the diagonal matrix \mathbf{E}_Y are iid zero-mean Gaussian with variance σ_η^2 and that the rows of \mathbf{E}_A are independent zero-mean Gaussian random*

vectors with covariance $\sigma_{\delta}^2 \mathbf{I}$. If \mathbf{E}_Y and \mathbf{E}_A are independent of each other, the expected squared first-order reconstruction errors are

$$\begin{aligned} \mathbb{E} [e_{\text{TLS}}^2] &= \sigma_{\delta}^2 \cdot \|\mathbf{x}^{\#}\|_2^2 \left\| \left(\tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \mathbf{D} \tilde{\mathbf{A}} \right)^{-1} \tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \mathbf{D} \right\|_F^2 \\ &\quad + \frac{\sigma_{\eta}^2}{4} \cdot \left\| \left(\tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \mathbf{D} \tilde{\mathbf{A}} \right)^{-1} \tilde{\mathbf{A}}^T \tilde{\mathbf{Y}}^{\frac{1}{2}} \mathbf{D} \right\|_F^2 \end{aligned} \quad (4.47)$$

$$\begin{aligned} \mathbb{E} [e_{\text{LS}}^2] &= \sigma_{\delta}^2 \cdot \|\mathbf{x}^{\#}\|_2^2 \left\| \left(\tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \tilde{\mathbf{A}} \right)^{-1} \tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \right\|_F^2 \\ &\quad + \frac{\sigma_{\eta}^2}{4} \cdot \left\| \left(\tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \tilde{\mathbf{A}} \right)^{-1} \tilde{\mathbf{A}}^T \tilde{\mathbf{Y}}^{\frac{1}{2}} \right\|_F^2. \end{aligned} \quad (4.48)$$

Proof. The expectations are computed in Appendix B.4. \square

Just as in Proposition 4, the difference between the TLS and LS expressions in Proposition 5 are due to the diagonal matrix \mathbf{D} . Each expression is a sum of two terms—the first term shows how the expectations depend on σ_{δ}^2 and the second term shows how they depend on σ_{η}^2 .

4.3.2 Reconstruction error numerical experiments

The expressions in Proposition 4 provide a means to understand when each approach should perform well. Furthermore, their squared-expectations in Proposition 5 allow us to verify the optimal maximum likelihood parameters stated in Proposition 3.

Impact of varying error strength and number of measurements We compare TLS and LS by numerically evaluating (4.45) and (4.46) with different measurement and sensing vector error levels while varying the number of measurements.

These experiments only consider the first-order error. The actual error is computed in a variety of experiments in Section 4.4. We will use SNR to quantify the measurement and sensing vector error level. The measurement SNR is $-20 \log_{10}(\|\mathbf{E}_Y\|_F / \|\tilde{\mathbf{Y}}\|_F)$ and similarly the sensing vector SNR is $-20 \log_{10}(\|\mathbf{E}_A\|_F / \|\tilde{\mathbf{A}}\|_F)$. Furthermore we define the relative reconstruction errors, $\text{rel.}e_{\text{TLS}} = \frac{e_{\text{TLS}}}{\|\mathbf{x}^{\#}\|}$ and $\text{rel.}e_{\text{LS}} = \frac{e_{\text{LS}}}{\|\mathbf{x}^{\#}\|}$.

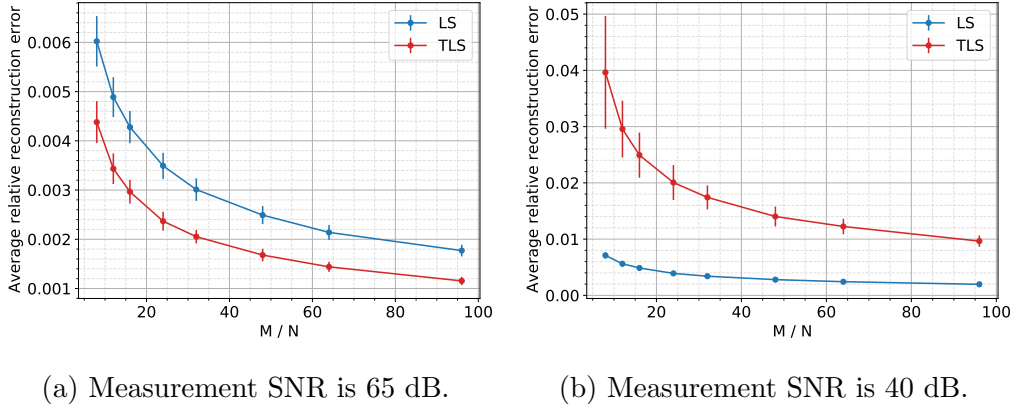


Figure 4.2: Relative reconstruction errors, (4.45) and (4.46) for different values of $\frac{M}{N}$ when sensing vector SNR is 40 dB and measurement SNR is varied. All errors are Gaussian.

We plot the relative reconstruction errors as the oversampling ratio $\frac{M}{N}$ is varied with $N = 100$. Regularization parameters λ_y and λ_a are set to one. For each value of $\frac{M}{N}$ we do 100 trials, and each trial uses new sensing vectors, ground truth signals, and errors. The standard deviation of the trials is indicated by error bars in the plots. The sensing vectors and ground truth signal are iid standard real Gaussian. Furthermore, the measurement and sensing vector errors are iid zero-mean real Gaussian with variance such that the sensing vector SNR is 40 dB. In Figure 4.2a the measurement SNR is 65 dB and TLS has lower reconstruction error than LS. When the measurement SNR decreases to 40 dB in Figure 4.2b, LS outperforms TLS. Although these experiments use the first-order error, they are consistent with our intuition. The relative performance of TLS is better when most of the error is due to sensing vector error. We also see that the performance of both methods improves as the number of measurements increases. Lastly, from Figure 4.2b, TLS may improve relatively faster than LS as the number of measurements increase.

Verification of optimal ML parameters The expression for TLS (4.47) in Proposition 5 enables us to verify the optimal maximum likelihood parameters for λ_y and λ_a from Proposition 3. Although Proposition 3 is stated for the complex-valued phase retrieval problem, the same procedure shows that the optimal parameters are the same for real-valued phase retrieval which we

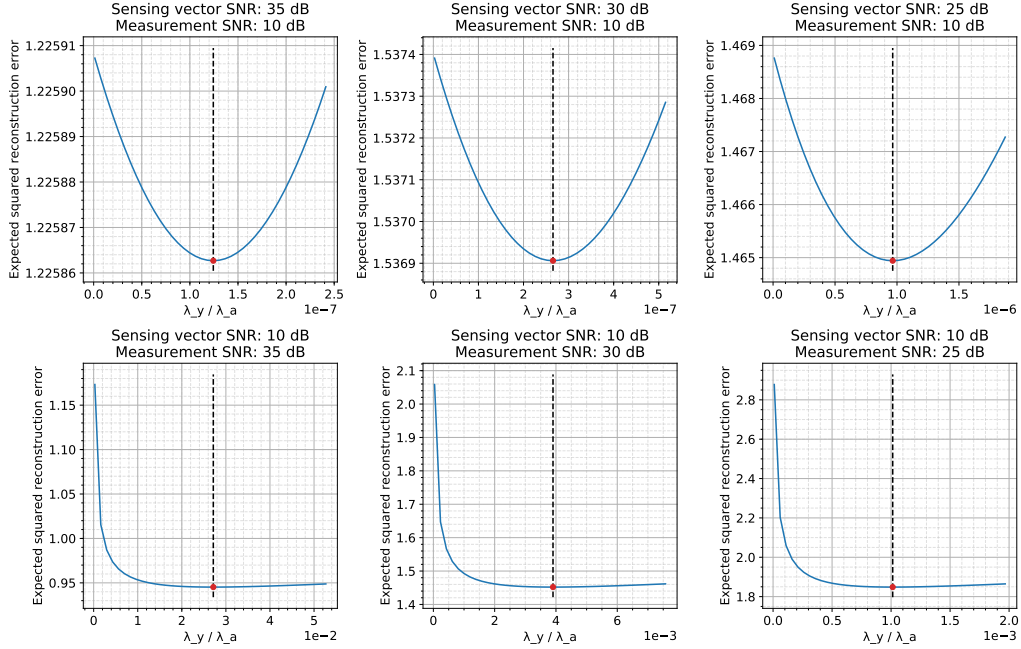


Figure 4.3: The TLS expected squared reconstruction error (4.47) is plotted for different ratios, $\frac{\lambda_y}{\lambda_a}$, to verify the optimal maximum likelihood parameters. Each subplot shows a different combination of sensing vector and measurement SNR. The minima are marked in red, and the theoretically optimal ratio is indicated by the dashed black lines.

consider here. The theoretically optimal parameter ratio is $\frac{\lambda_y}{\lambda_a} = \frac{\sigma_\delta^2}{\sigma_\eta^2}$.

To verify numerically whether this agrees with Proposition 5, we vary $\frac{\lambda_y}{\lambda_a}$ (which is contained in \mathbf{D}) around the optimal ratio and plot the TLS expression (4.47). We do this multiple times and in each run use a different iid standard real Gaussian ground truth signal and a different set of iid standard real Gaussian sensing vectors. As in Proposition 5, the errors in each run are iid zero-mean Gaussian and their variances are set to obtain different SNRs. Figure 4.3 shows the different runs with the minima marked in red and the theoretically optimal ratio indicated by the dashed black lines. We can see that all the minima are at the optimal ratio which verifies Proposition 3. In the top row, most of the error is due to measurement error and the optimal ratio is low. This further highlights that TLS sensing vector corrections are less important when most of the error is due to measurement error.

We further note that the consistency between Propositions 3 and 5 demonstrates that the first-order expressions can be used to explain the performance

of our TLS framework. Section 4.4 shows that the real reconstruction errors follow the same trends as the numerical simulations in this section.

4.4 TLS phase retrieval simulations

We compare the performance of TLS phase retrieval against LS phase retrieval through simulations.¹ To obtain a LS solution we use the Wirtinger flow method [28].

In this section we set the regularization parameters of (TLS-PR2) to $\lambda_a = \frac{1}{N}$ and $\lambda_y = \frac{1}{\|\mathbf{x}^{(0)}\|_2^4}$ in all experiments with $\mathbf{x}^{(0)}$ being an initial guess for $\mathbf{x}^\#$. These regularization parameters are tuned later in Section 4.5. We fix the ground truth signal to be iid complex Gaussian with $N = 100$. Furthermore, the TLS and LS iterations are stopped when their objective function values change by less than 10^{-6} between successive iterates. The ground truth signal, TLS solution, and LS solution are denoted as $\mathbf{x}^\#$, $\mathbf{x}_{\text{TLS}}^\dagger$ and $\mathbf{x}_{\text{LS}}^\dagger$. In all experiments, we generate M quadratic measurements using M clean sensing vectors. The TLS and LS methods must then recover the signal $\mathbf{x}^\#$ from erroneous measurements and sensing vectors. We use SNR, as defined in Section 4.3, to quantify measurement and sensing vector error. Also as in Section 4.3, the plots in this section indicate the standard deviation of the trials using error bars.

4.4.1 Measurement models

In our experiments, we will consider the complex-valued Gaussian and coded diffraction pattern measurement models. However, Algorithm 2 is not restricted to these measurement models. Recently in optical computing applications, random Gaussian scattering media have been used to do rapid high-dimensional randomized linear algebra, kernel classification, and dimensionality reduction using laser light [20, 30]. The coded diffraction pattern model modulates the signal with different patterns before taking the Fourier transform. It is inspired by the fact that in coherent x-ray imaging the field at the detector is the Fourier transform of the signal [1].

¹Code available at https://github.com/swing-research/tls_phase.

When using the Gaussian measurement model, the n th entry of sensing vector m , a_{mn} , is distributed by the complex normal distribution for the complex-valued problem, $a_{mn} \sim \mathcal{N}(0, 1) + j\mathcal{N}(0, 1)$. For the real-valued problem, it is the standard normal distribution, $a_{mn} \sim \mathcal{N}(0, 1)$. The Gaussian measurement model sensing vector entries are independent of each other and the sensing vectors are also independent of each other. A description of the coded diffraction pattern measurement model is in Appendix B.7.

In this section, the complex Gaussian measurement model is used. In Appendix B.7.1 these experiments are repeated for the coded diffraction pattern measurement model and the same behavior is seen.

4.4.2 Algorithm initialization

In our experiments we opt to do the initialization of the signal being recovered (line 1 of Algorithm 2) via a spectral initialization. The method comprising a spectral initialization followed by gradient descent updates has been proven to lead to globally optimal solutions for the LS phase retrieval problem (LS-PR) in an error-free setting under the Gaussian and coded diffraction pattern models [27, 28].

The spectral initialization is the leading eigenvector of the matrix $\sum_m y_m \mathbf{a}_m \mathbf{a}_m^* \in \mathbb{C}^{N \times N}$ which we efficiently compute using 50 power method iterations. This eigenvector is scaled appropriately by estimating the norm of the signal of interest as $(\frac{1}{2M} \sum_m y_m)^{1/2}$.

4.4.3 Signal recovery

To evaluate performance, we compute the distance between the ground truth signal and the recovered signal. As the value of the objective function (TLS-PR1) is the same for \mathbf{x} and phase shifted $e^{j\varphi} \mathbf{x}$, we cannot distinguish between \mathbf{x} and its phase-shifted variant. We therefore use a standard definition of distance that is invariant to phase shifts which is detailed in Definition 1.

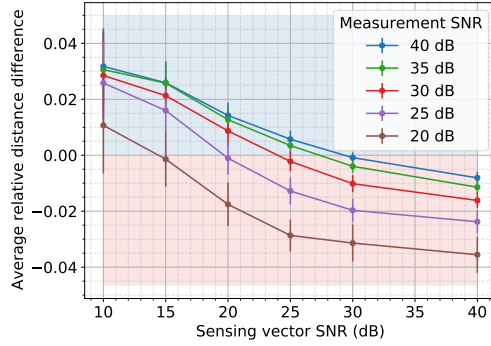
Definition 1. Denote the ground truth as $\mathbf{x}^\# \in \mathbb{C}^N$ and let $\mathbf{x}^\dagger \in \mathbb{C}^N$ be a solution to the phase retrieval problem. The distance between $\mathbf{x}^\#$ and \mathbf{x}^\dagger ,

$\text{dist}(\mathbf{x}^\#, \mathbf{x}^\dagger)$, is defined as $\text{dist}(\mathbf{x}^\#, \mathbf{x}^\dagger) = \min_{\varphi \in [0, 2\pi)} \|\mathbf{x}^\# - e^{j\varphi} \mathbf{x}^\dagger\|_2$. Furthermore, the relative distance is defined as $\text{rel.dist}(\mathbf{x}^\#, \mathbf{x}^\dagger) = \frac{\text{dist}(\mathbf{x}^\#, \mathbf{x}^\dagger)}{\|\mathbf{x}^\#\|_2}$ and the reconstruction SNR in dB is defined as $-20 \log_{10}(\text{rel.dist}(\mathbf{x}^\#, \mathbf{x}^\dagger))$.

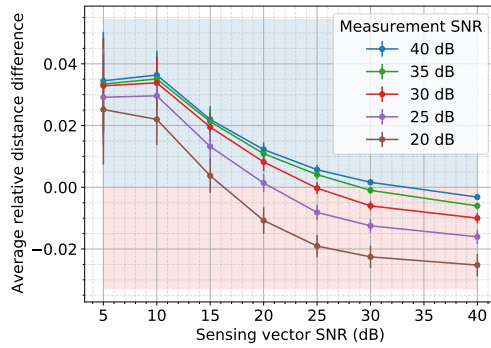
Combinations of sensing vector and measurement error To understand how performance changes with different amounts of sensing vector and measurement error, we add different amounts of random iid complex Gaussian error to sensing vectors and random iid real Gaussian error to measurements. For each combination of sensing vector error and measurement error we perform 100 phase retrieval trials. In each trial, we generate a new ground truth signal and M new sensing vectors to produce M new error-free measurements. In each trial, we then add new random error perturbations to the sensing vectors and measurements. We evaluate performance by subtracting the relative distance of the TLS solution from that of the LS solution, $(\text{rel.dist}(\mathbf{x}^\#, \mathbf{x}_{\text{LS}}^\dagger) - \text{rel.dist}(\mathbf{x}^\#, \mathbf{x}_{\text{TLS}}^\dagger))$, and average across all 100 trials. If this average is positive, TLS has outperformed LS.

We use a step size of $\mu = \frac{0.5}{\lambda_a}$ for TLS and $\mu = 0.02$ for LS to perform the gradient update for \mathbf{x} in (4.7). The TLS step size is inversely proportional to λ_a because the relative importance of the data consistency term is inversely proportional to the sensing vector consistency term in (TLS-PR2). Figure 4.4 shows the performance for $\frac{M}{N} \in \{8, 16, 32\}$. Note that the minimum sensing vector SNR is 10 dB when $\frac{M}{N} = 8$ and 5 dB in the other cases. For a fixed sensing vector SNR, the performance of TLS decreases when the measurement SNR decreases. This is expected because more of the error is in the measurements which LS is designed for. In general, TLS is better when the sensing vector SNR decreases for a fixed measurement SNR because TLS phase retrieval accounts for sensing vector error. However, this trend starts to break for very low sensing vector SNR as shown at 5 dB when $\frac{M}{N} = 16$. Increasing the number of measurements overcomes this issue and in general improves TLS performance as was indicated by the first-order reconstruction errors with Gaussian error in Figures 4.2a and 4.2b.

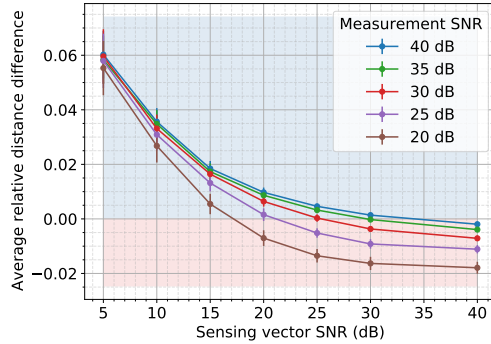
Impact of varying the number of measurements To clearly see the impact of varying the number of measurements we fix the measurement SNR to 20 dB and sensing vector SNR to 10 dB and plot the reconstruction relative distance for TLS and LS in Figure 4.5a. We do 100 trials for each value of $\frac{M}{N}$.



(a) $\frac{M}{N} = 8$



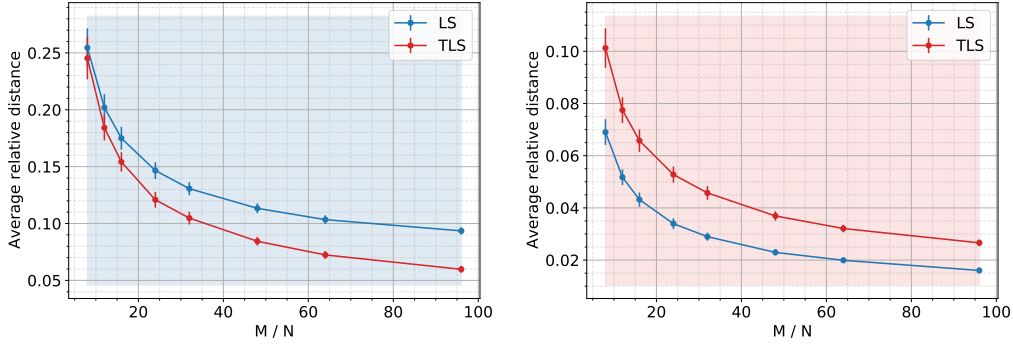
(b) $\frac{M}{N} = 16$



(c) $\frac{M}{N} = 32$

Figure 4.4: Average difference in relative distance of TLS and LS solutions, $\text{rel.dist}(\mathbf{x}^\#, \mathbf{x}_{\text{LS}}^\dagger) - \text{rel.dist}(\mathbf{x}^\#, \mathbf{x}_{\text{TLS}}^\dagger)$, for the Gaussian measurement model for different measurement and sensing vector SNR combinations when $\frac{M}{N} \in \{8, 16, 32\}$.

The performance improvement of TLS over LS increases as the number of measurements are increased. In Figure 4.5b, we increase the sensing vector SNR to 30 dB. When the balance of the Gaussian error shifts more towards the measurements, LS performs better. This is identical to what was seen



(a) Sensing vector SNR is 10 dB.

(b) Sensing vector SNR is 30 dB.

Figure 4.5: Relative distance of reconstructions using TLS and LS for the Gaussian measurement model for different $\frac{M}{N}$ when measurement SNR is 20 dB and measurement SNR is varied. All errors are Gaussian.

with the first-order reconstruction errors in Figures 4.2a and 4.2b.

Accuracy of first-order reconstruction errors Appendix B.5 contains the description of an experiment where we verify the accuracy of the reconstruction error expressions in Proposition 4 against the real errors. As expected, it shows that the first-order expressions in Proposition 4 increase in accuracy as the sensing vector and measurement error decreases—this corresponds to the norm of γ in (4.27) decreasing, but that these expressions may only serve as a rough rule of thumb when errors are large.

Sensing vectors and measurements error model The simulations in this section use iid random Gaussian errors. In Appendix B.6, we design errors that require access to the ground truth signal norm and to the error-free measurements. We show that the improvement of TLS over LS can be larger in this artificial scenario.

4.4.4 Corrected sensing vector verification

To characterize the sensing vector corrections performed by our algorithm, we define a metric sensitive to the relative correction error of the sensing vectors. The metric only considers corrections in the direction of the recovered signal because our algorithm only corrects the component of the sensing vectors in the direction of this signal due to the optimization geometry (Section 4.2.1).

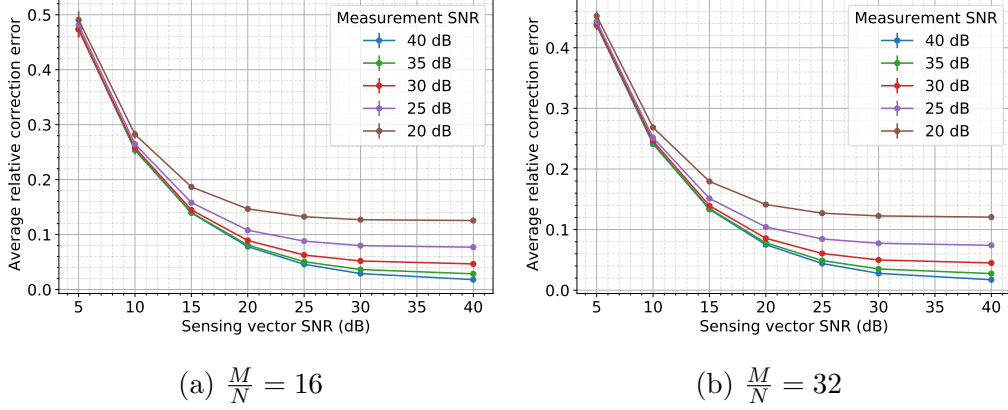


Figure 4.6: Average relative sensing vector correction error when using TLS, $\text{rel.corr}(\{\tilde{\mathbf{a}}, \mathbf{x}^\#\}, \{\hat{\mathbf{a}}^\dagger, e^{j\varphi} \mathbf{x}_{\text{TLS}}^\dagger\})$, for the Gaussian measurement model for different measurement and sensing vector SNR combinations when $\frac{M}{N} \in \{16, 32\}$.

Definition 2. Denote the complex-valued ground truth signal and sensing vectors as $\mathbf{x}^\#$ and $\mathbf{a}^\# := \{\mathbf{a}_m^\#\}_{m=1}^M$. Let their counterparts obtained by solving the TLS phase retrieval problem be \mathbf{x}^\dagger and $\mathbf{a}^\dagger := \{\mathbf{a}_m^\dagger\}_{m=1}^M$. Further let $\varphi = \arg \min_{\varphi \in [0, 2\pi)} \|\mathbf{x}^\# - e^{j\varphi} \mathbf{x}^\dagger\|_2$. Then, denoting $\mathbf{y}(\mathbf{a}, \mathbf{x}) = [\langle \mathbf{a}_1, \mathbf{x} \rangle, \dots, \langle \mathbf{a}_M, \mathbf{x} \rangle]$, the relative sensing vector correction error between $\mathbf{a}^\#$ and \mathbf{a}^\dagger is defined as $\text{rel.corr}(\{\mathbf{a}^\#, \mathbf{x}^\#\}, \{\mathbf{a}^\dagger, \mathbf{x}^\dagger\}) = \frac{\|\mathbf{y}(\mathbf{a}^\#, \mathbf{x}^\#) - \mathbf{y}(\mathbf{a}^\dagger, e^{j\varphi} \mathbf{x}^\dagger)\|_2}{\|\mathbf{y}(\mathbf{a}^\#, \mathbf{x}^\#)\|_2}$.

To evaluate performance, we denote the ground truth and TLS corrected sensing vectors as $\{\tilde{\mathbf{a}}_m\}_{m=1}^M$ and $\{\hat{\mathbf{a}}_m^\dagger\}_{m=1}^M$. For the sensing vectors in the previous experiments of Figure 4.4 we compute $\text{rel.corr}(\{\tilde{\mathbf{a}}, \mathbf{x}^\#\}, \{\hat{\mathbf{a}}^\dagger, \mathbf{x}_{\text{TLS}}^\dagger\})$ and average across the 100 trials. Figure 4.6 shows the relative correction error when $\frac{M}{N} \in \{16, 32\}$. We see that as the sensing vector SNR increases, the relative correction error decreases. Furthermore, as the measurement SNR decreases, the relative correction error increases. These relative correction error increases are more pronounced when the sensing vector SNR is high, a setting where sensing vector correction is needed less. This observation is consistent with Figure 4.4—when sensing vector SNR is high, the TLS sensing vector corrections hinder TLS performance and LS outperforms TLS.

Next we investigate how the number of measurements impacts the relative correction error. We do this with the sensing vectors from the previous

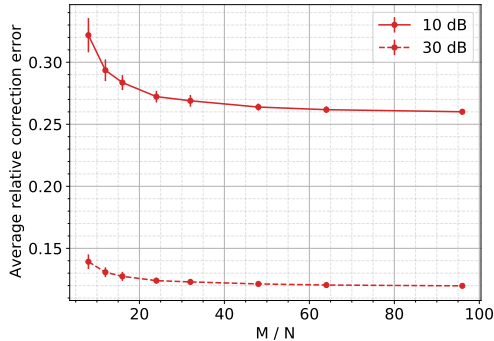


Figure 4.7: Relative sensing vector correction error when using TLS for the Gaussian measurement model for different $\frac{M}{N}$ when measurement SNR is 20 dB. The sensing vector SNR is 10 dB or 30 dB.

experiments in Figures 4.5a and 4.5b. Figure 4.7 shows the averages over the 100 trials. Here, the measurement SNR was fixed to 20 dB and the sensing vector SNR was 10 dB or 30 dB. Consistent with what was seen previously, the performance of TLS improves with increasing number of measurements. Additionally, increasing the number of measurements provides greater gains when the sensing vector SNR is lower.

4.5 Experiments on real optical hardware

In this section, we show that TLS phase retrieval outperforms LS phase retrieval when using real optical hardware. We use the same optical processing unit (OPU) that was used in the hardware experiments of Chapters 2 and 3.² A known signal $\mathbf{x}^\# \in \mathbb{R}^N$ is encoded onto coherent laser light using a digital micro-mirror device (DMD) which is then shined through a Gaussian multiple scattering medium as shown in Figure 4.8. We denote the transmission matrix of the Gaussian medium as $\mathbf{A} \in \mathbb{C}^{M \times N}$. The M rows of the transmission matrix are sensing vectors, $\mathbf{a}_m \in \mathbb{C}^N$ for $1 \leq m \leq M$. The intensity of the scattered light in the sensor plane, $y_m \approx |\langle \mathbf{a}_m, \mathbf{x}^\# \rangle|^2$ for all m , is then measured using a camera. We do phase retrieval using the optical measurements to reconstruct the input, $\mathbf{x}^\#$. The input signals are limited to real-valued binary images due to the DMD.

²Visit <https://www.lighton.ai/lighton-cloud/> for a publicly available cloud OPU with a scikit-learn interface.

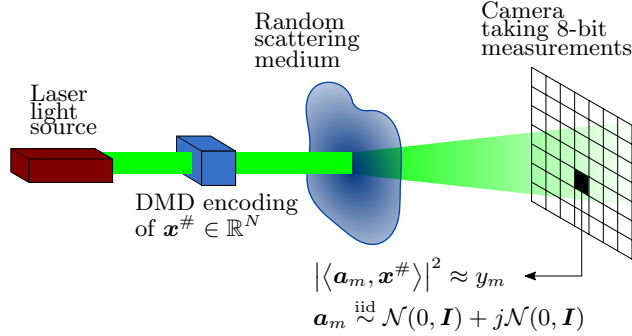


Figure 4.8: The optical processing unit (OPU). A coherent laser beam spatially encodes a signal, \mathbf{x} , via a digital micro-mirror device (DMD) which is then shined through a random medium. A camera measures the squared magnitude of the scattered light.

The OPU measurements and sensing vectors both contain errors. Errors in the optical measurements are caused by 8-bit quantized camera measurements and Poisson noise which scales with the square root of the mean intensity of the scattered light. Additionally, there are measurement errors due to thermal effects and other system properties that result in a noise floor. Thus, the lowest intensity that can be measured is not zero, even if an all-zero signal, $\mathbf{x}^\# = \mathbf{0}$, is encoded on the laser light and shined through the scattering medium. The sensing vectors are erroneous because the entries of \mathbf{A} are unknown and must be calibrated from erroneous optical measurements as explained in Chapter 3. There may also be other sources of experimental error. Unlike in the computer simulations of Section 4.4, when using the OPU, we do not know the exact error model and we also do not know the levels of the errors in the measurements and sensing vectors.

In the experiments, the TLS and LS step sizes are tuned to $\frac{0.4}{\lambda_a}$ and 0.005. The initialization method and termination criteria are the same as in Section 4.4. Additionally, we use the fact that the images being reconstructed are real-valued and binary to regularize both the TLS and LS methods. After the initialization (Algorithm 2, Step 1 for TLS) and each \mathbf{x} update step (Algorithm 2, Step 17 for TLS), we take the elementwise absolute value of the signal to set the phase of all elements to zero. We then normalize the entries of $\mathbf{x}_{\text{TLS}}^\dagger$ and $\mathbf{x}_{\text{LS}}^\dagger$ with absolute value larger than one to one.

Appendix B.8 contains details of the sensing vector calibration method used and further OPU experimental details.

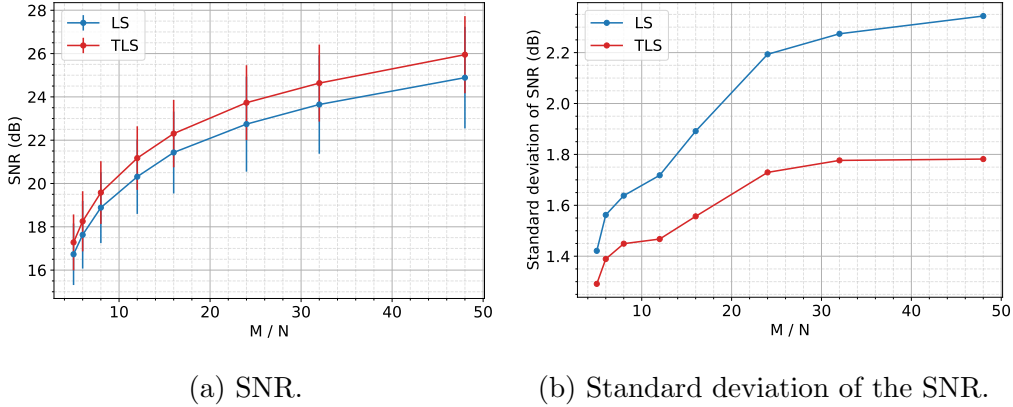


Figure 4.9: Performance when reconstructing random binary images when using TLS and LS for phase retrieval on the OPU. Values of $\frac{M}{N}$ between five and 48 are used.

Random ground truth signals Our ground truth signals are real-valued random binary images of size $N = 16 \times 16 = 256$. We vary the oversampling ratio, $\frac{M}{N}$, and perform 100 trials for each ratio. In each trial a new ground truth image and set of calibrated sensing vectors is used. On a held out set of ten images and with $\frac{M}{N} = 8$ we tune $\lambda_a = 40$ and $\lambda_y = \|\mathbf{x}^{(0)}\|_2^{-4}$ in (TLS-PR2) where $\mathbf{x}^{(0)}$ is the initialization. Figure 4.9a shows that the SNR of the reconstructed images using TLS is higher than when using LS for all numbers of measurements. Additionally, in Figure 4.9b we plot the standard deviation of the results in Figure 4.9a and show that the TLS method has lower variability.

Real image ground truth signals We reconstruct binary images of size $N = 32 \times 32 = 1024$ for $\frac{M}{N} \in \{5, 8, 12\}$. On a held out set of five images and with $\frac{M}{N} = 5$ we tune $\lambda_a = 20$ and again $\lambda_y = \|\mathbf{x}^{(0)}\|_2^{-4}$ in (TLS-PR2) where $\mathbf{x}^{(0)}$ is the initialization. Figure 4.10 shows the original images, their reconstructions and their reconstruction SNR. For a given oversampling ratio, the TLS approach reports better SNR values and reconstructs images of better visual quality as compared to the LS approach.

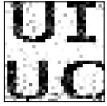


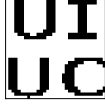
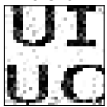


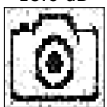
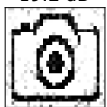
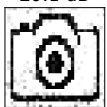


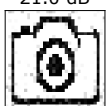

	$M / N = 5$	$M / N = 8$	$M / N = 12$	Original	
LS	15.8 dB 	17.4 dB 	18.9 dB 		
	16.9 dB 	18.2 dB 	19.5 dB 		
TLS	18.6 dB 	19.2 dB 	20.1 dB 		
	20.2 dB 	21.0 dB 	22.5 dB 		

Figure 4.10: Reconstructions of 32×32 images and reconstruction SNR when using TLS and LS on the OPU. The oversampling ratio is varied.

4.6 Summary

We have developed a TLS framework for solving the phase retrieval problem that accounts for both sensing vector error and measurement error. One of the keys to solving the TLS problem via gradient descent was studying the geometry of the TLS optimization problem to realize that the sensing vectors can be efficiently updated by solving a scalar variable optimization problem instead of a vector variable optimization problem. By deriving the Taylor series expansions for the TLS and LS solutions, we have also obtained approximate expressions for their reconstruction error. These expressions enabled us to anticipate the accuracy of the TLS solution relative to the LS solution and understand when which approach will lead to a better solution. We verify the TLS method through a range of computer simulations. Furthermore, in experiments with real optical hardware, TLS outperforms LS.

CHAPTER 5

RANDOM MESH PROJECTORS

Motivated by settings commonly seen in the applied sciences, in this chapter propose a deep learning method to solve inverse problems when there are few measurements and no training data. This work falls under **E1** because we transform our measurements to solve a new inverse problem with different measurements.

5.1 Scarce measurements and limited training data

As described in Chapter 1, a variety of imaging inverse problems can be discretized to a linear system $\mathbf{y} = \mathbf{A}\mathbf{x} + \boldsymbol{\eta}$ where $\mathbf{y} \in \mathbb{R}^M$ is the measured data, $\mathbf{A} \in \mathbb{R}^{M \times N}$ is the imaging or forward operator, $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^N$ is the object being probed by applying \mathbf{A} , and $\boldsymbol{\eta}$ is the noise. Depending on the application, the set of plausible reconstructions \mathcal{X} can model natural, seismic, or biomedical images. In many cases, the resulting inverse problem is ill-posed, either because of the poor conditioning of \mathbf{A} (a consequence of the underlying physics) or because $M \ll N$.

A classical approach to solve ill-posed inverse problems is to minimize an objective functional regularized via a certain norm (e.g. ℓ_1 , ℓ_2 , total variation (TV) seminorm) of the object. These methods promote general properties such as sparsity or smoothness of reconstructions, sometimes in combination with learned synthesis or analysis operators or dictionaries [85].

In this chapter, we address situations with very scarce measurement data ($M \ll N$) so that even a coarse reconstruction of the unknown object is hard to get with traditional regularization schemes. Unlike artifact-removal scenarios where applying a regularized pseudoinverse of the imaging operator

This work was previously published in S. Gupta, K. Kothari, M. v. de Hoop, and I. Dokmanić, “Random mesh projectors for inverse problems,” in International Conference on Learning Representations, 2019 [17] and is adapted here with permission.

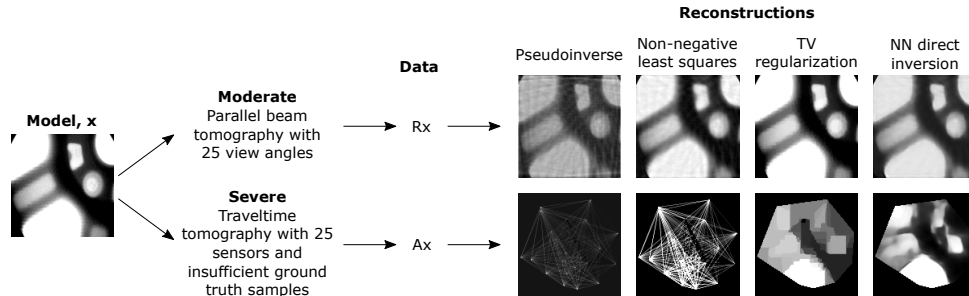


Figure 5.1: We reconstruct an image \mathbf{x} from its tomographic measurements. In moderately ill-posed problems, conventional methods based on the pseudoinverse and regularized non-negative least squares ($\mathbf{x} \in [0, 1]^N$, N is image dimension) give correct structural information. In fact, total variation (TV) approaches give very good results. A neural network [89] can be trained to directly invert and remove the artifacts (NN). In a severely ill-posed problem on the other hand (explained in Figure 5.5) with insufficient ground truth training data, neither the classical techniques nor a neural network recover salient geometric features.

already brings out considerable structure, we look at applications where standard techniques cannot produce a reasonable image. An example is shown in Figure 5.1. This highly imaging setting is common in geophysics and requires alternative, more involved strategies [86].

An appealing alternative to classical regularizers is to use deep neural networks. For example, generative models (GANs) based on neural networks have recently achieved impressive results in regularization of inverse problems [87], [88]. However, a difficulty in geophysical applications is that there are very few examples of ground truth objects available for training (sometimes none at all). Since GANs require many, they cannot be applied to such problems. This suggests to look for methods that are not very sensitive to the training dataset. Conversely, it means that the sought reconstructions are less detailed than what is expected in data-rich settings.

5.1.1 Chapter overview and contributions

In this chapter, we propose a two-stage method to solve ill-posed inverse problems using random low-dimensional projections and convolutional neural networks. We first decompose the inverse problem into a collection of simpler learning problems of estimating projections into random (but structured)

low-dimensional subspaces of piecewise-constant images.

In the second stage, we solve a new reformulated linear inverse problem that combines the estimates from the different subspaces. We show that this converts the original problem with possibly non-local (often tomographic) measurements into a new inverse problem with localized measurements, and that in expectation over random subspaces the problem becomes a deconvolution. Intuitively, projecting into piecewise-constant subspaces is equivalent to estimating local averages—a simpler problem than estimating individual pixel values. Combining the local estimates lets us recover the underlying structure.

We test our method on linearized seismic traveltime tomography [90, 91] with scarce measurements and show that it outperforms learned direct inversion in quality of achieved reconstructions, robustness to measurement errors, and (in)sensitivity to the training data. The latter is essential in domains with insufficient ground truth images.

5.1.2 Related work

Although neural networks have long been used to address inverse problems [92, 93, 94], the past few years have seen the number of related deep learning papers grow exponentially. The majority address biomedical imaging [95, 96] with several special issues¹ and review papers [100, 101] dedicated to the topic. All these papers address reconstruction from subsampled or low-quality data, often motivated by reduced scanning time or lower radiation doses. Beyond biomedical imaging, machine learning techniques are emerging in geophysical imaging [102, 103, 104], though at a slower pace, perhaps partly due to the lack of standard open datasets.

Existing methods can be grouped into non-iterative methods that learn a feed-forward mapping from the measured data \mathbf{y} (or some standard manipulation such as adjoint or a pseudoinverse) to the object \mathbf{x} [89, 105, 106, 107, 108, 109, 110]; and iterative energy minimization methods, with either the regularizer being a neural network [111], or neural networks replacing various iteration components such as gradients, projectors, or proximal mappings [112, 113, 114, 115]. These are further related to the notion of plug-and-play

¹IEEE Transactions on Medical Imaging, May 2016 [97]; IEEE Signal Processing Magazine, November 2017, January 2018 [98, 99].

regularization [116], as well as early uses of neural nets to unroll and adapt standard sparse reconstruction algorithms [117, 118]. An advantage of the first group of methods is that they are fast; an advantage of the second group is that they are better at enforcing data consistency.

A rather different take was proposed in the context of compressed sensing where the reconstruction is constrained to lie in the range of a pretrained generative network [119, 87]. This scheme achieves impressive results on random sensing operators and comes with theoretical guarantees. However, training generative networks requires many examples of ground truth and the method is inherently subject to dataset bias. Here, we focus on a setting where ground-truth samples are very few or impossible to obtain.

There are connections between this work and sketching [120, 121] where the learning problem is also simplified by random low-dimensional projections of some object—either the data or the unknown reconstruction itself [24]. This also exposes natural connections with learning via random features [122, 123].

5.2 Regularization by random mesh projections

The two stages of our method are 1) decomposing a “hard” learning task of directly learning a map from measurements to the unknown object into an ensemble of “easy” tasks of estimating projections of the unknown object into low-dimensional subspaces, and 2) combining these projection estimates to solve a reformulated inverse problem for \mathbf{x} . The two stages are summarized in Figure 5.2. The method outlined in Figure 5.2, can be interpreted as a variance reduction ensembling strategy [124] that allows us to combine multiple “views” of the unknown object and obtain a more robust final solution.

5.2.1 Learning random projections

In imaging, numerous signal processing and machine learning methods have relied on our ability to represent or transform signals in an effective manner. If the goal is interpolation, splines are widely used to model signals as continuous quantities [125, 126]. A general motive for finding a good representation is that it provides some convenience. For example, representing images in the Fourier or wavelet domain helps us remove artifacts due to noise be-

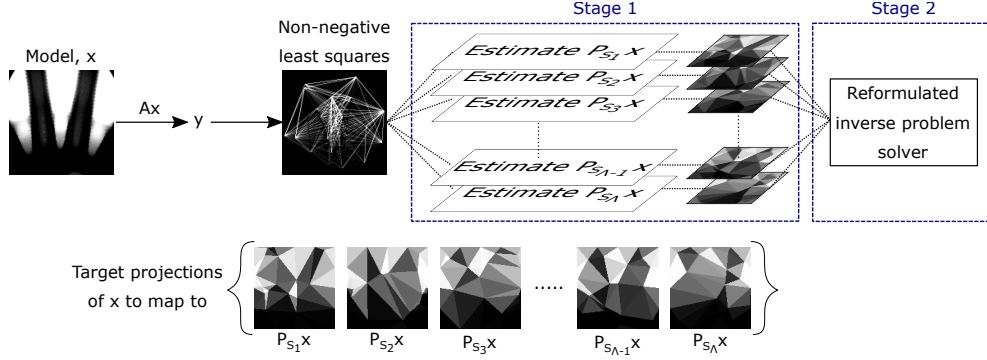


Figure 5.2: Regularization by Λ random projections: 1) each orthogonal projection is approximated by a convolutional neural network which maps from a non-negative least squares reconstruction of an image to its projection onto a lower dimension subspace of Delaunay triangulations; 2) projections are combined to estimate the original image using regularized least squares.

cause these domains emphasize the inherent structure in images [127]. There are also benefits to breaking away from these predetermined domains and learning a task-specific domain or basis using dictionaries [85].

In particular, representing signals in low dimensional spaces has proven to be particularly beneficial in a variety of techniques [119] and has led to projecting data into random low-dimensional subspaces becoming a fundamental area of development [24, 21]. As local areas of pixels in an image are correlated, in this work we propose to use random piecewise-constant low-dimensional subspaces to accurately learn local image structures.

We sample Λ sets of points in the image domain from a uniform-density Poisson process and construct Λ (discrete) Delaunay triangulations with those points as vertices. Let $\mathcal{S} = \{S_\lambda \mid 1 \leq \lambda \leq \Lambda\}$ be the collection of Λ subspaces of piecewise-constant functions on these triangulations. Let further G_λ be the map from \mathbf{y} to the projection of the object into subspace S_λ , $G_\lambda \mathbf{y} = \mathbf{P}_{S_\lambda} \mathbf{x}$. Instead of learning the “hard” inverse mapping from measurements to the unknown object, we propose to learn an ensemble of simpler mappings $\{G_\lambda\}_{\lambda=1}^\Lambda$. Intuitively, projecting into piecewise-constant subspaces requires estimating local averages which is a simpler task than estimating each pixel values with an inverse mapping.

We approximate each G_λ by a convolutional neural network, $\Gamma_{\theta(\lambda)}(\tilde{\mathbf{y}}) : \mathbb{R}^N \rightarrow \mathbb{R}^N$, parameterized by a set of trained weights $\theta(\lambda)$. Similar to [89], we do not use the measured data $\mathbf{y} \in \mathbb{R}^M$ directly as this would require

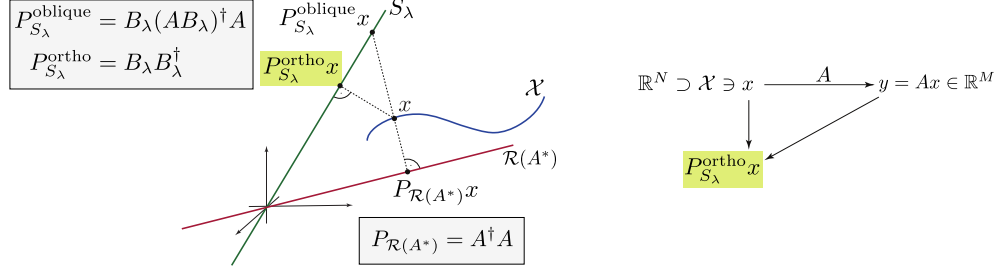


Figure 5.3: Orthogonal and oblique projections. There is no linear operator acting on \mathbf{y} or on the orthogonal projection $\tilde{\mathbf{y}} = \mathbf{P}_{\mathcal{R}(A^*)}\mathbf{x} = \mathbf{A}^\dagger\mathbf{y}$ that can compute the orthogonal projection into S .

the network to first learn to map \mathbf{y} back to the image domain; we rather warm-start the reconstruction by a non-negative least squares reconstruction, $\tilde{\mathbf{y}} \in \mathbb{R}^N$, computed from \mathbf{y} . The weights are chosen by minimizing empirical risk:

$$\theta(\lambda) = \arg \min_{\theta} \frac{1}{J} \sum_{j=1}^J \left\| \Gamma_{\theta(\lambda)}(\tilde{\mathbf{y}}_j) - \mathbf{P}_{S_\lambda} \mathbf{x}_j \right\|_2^2, \quad (5.1)$$

where $\{(\mathbf{x}_j, \tilde{\mathbf{y}}_j)\}_{j=1}^J$ is a set of J training objects and non-negative least squares measurements.

Need for non-linear operators. Projecting \mathbf{x} into a given known subspace is a simple linear operation, so it may not be a priori clear why we use non-linear neural networks to estimate the projections. However, we do not know \mathbf{x} and only have access to \mathbf{y} . Suppose that there exists a linear operator (a matrix) $\mathbf{F} \in \mathbb{R}^{N \times M}$ which acts on \mathbf{y} and computes the projection of \mathbf{x} on S_λ . A natural requirement on \mathbf{F} is consistency: if \mathbf{x} already lives in S_λ , then we would like to have $\mathbf{F}\mathbf{A}\mathbf{x} = \mathbf{x}$. This implies that for any \mathbf{x} , not necessarily in S_λ , we require $\mathbf{F}\mathbf{A}\mathbf{F}\mathbf{A}\mathbf{x} = \mathbf{F}\mathbf{A}\mathbf{x}$ which implies that $\mathbf{F}\mathbf{A} = (\mathbf{F}\mathbf{A})^2$ is an idempotent operator. Letting the columns of \mathbf{B}_λ be a basis for S_λ , it can be shown that the least squares minimizer for \mathbf{F} is $\mathbf{B}_\lambda(\mathbf{A}\mathbf{B}_\lambda)^\dagger$. However, because $\mathcal{R}(\mathbf{F}) = S_\lambda \neq \mathcal{R}(\mathbf{A}^*)$ (\mathbf{A}^* is the adjoint of \mathbf{A} , simply a transpose for real matrices), in general it will not hold that $(\mathbf{F}\mathbf{A})^* = \mathbf{F}\mathbf{A}$. Thus, $\mathbf{F}\mathbf{A}$ is an oblique, rather than orthogonal projection into S . In Figure 5.3 this corresponds to the point $\mathbf{P}_{S_\lambda}^{\text{oblique}}\mathbf{x}$ which can be arbitrarily far from the orthogonal projection $\mathbf{P}_{S_\lambda}^{\text{ortho}}\mathbf{x}$. Furthermore, the nullspace of the oblique projection is $\mathcal{N}(\mathbf{A}) = \mathcal{R}(\mathbf{A}^*)^\perp$.

Thus consistent linear operators can at best yield oblique projections which

can be far from the orthogonal one. This can be seen geometrically from Figure 5.3. As the angle between S_λ and $\mathcal{R}(\mathbf{A}^*)$ increases to $\pi/2$, the oblique projection point travels to infinity. Note that the oblique projection always happens along the nullspace of \mathbf{A} , which is the line orthogonal to $\mathbf{P}_{\mathcal{R}(\mathbf{A}^*)}$. Since our subspaces are chosen at random, in general they are not aligned with $\mathcal{R}(\mathbf{A}^*)$. The only subspace on which we can linearly compute an orthogonal projection from \mathbf{y} is $\mathcal{R}(\mathbf{A}^*)$ —this is given by the Moore-Penrose pseudoinverse of \mathbf{A} . Therefore, to get the orthogonal projection onto random subspaces, we must use non-linear operators.

As shown in the right half of Figure 5.3, if we assume \mathbf{A} is injective on \mathcal{X} , the existence of this non-linear map is guaranteed by construction—since \mathbf{y} can determine \mathbf{x} , it can also determine $\mathbf{P}_{S_\lambda}\mathbf{x}$. We show the results of numerical experiments in Figures C.1 and C.2 of the Appendix. These illustrate the performance difference between linear oblique projectors and our non-linear learned operator when estimating the projection of an image into a random subspace.

5.2.2 The new inverse problem

By learning projections onto random subspaces, we transform our original problem into that of estimating \mathbf{x} from $\{\Gamma_{\theta(\lambda)}(\tilde{\mathbf{y}})\}_{\lambda=1}^\Lambda$. To see how this can be done, ascribe to the columns of $\mathbf{B}_\lambda \in \mathbb{R}^{N \times K}$ a natural orthogonal basis for the subspace S_λ , $\mathbf{B}_\lambda = [\boldsymbol{\chi}_{\lambda,1}, \dots, \boldsymbol{\chi}_{\lambda,K}]$, with $\boldsymbol{\chi}_{\lambda,k}$ being the indicator function of the k th triangle in subspace λ . Denote by $\mathbf{q}_\lambda \stackrel{\text{def}}{=} \mathbf{q}_\lambda(\mathbf{y})$ the mapping from the data \mathbf{y} to an estimate of the expansion coefficients of \mathbf{x} in the basis for S_λ :

$$\mathbf{q}_\lambda(\mathbf{y}) \stackrel{\text{def}}{=} \mathbf{B}_\lambda^T \Gamma_{\theta(\lambda)}(\tilde{\mathbf{y}}) \quad (5.2)$$

Let $\mathbf{B} \stackrel{\text{def}}{=} [\mathbf{B}_1 \ \mathbf{B}_2 \ \dots \ \mathbf{B}_\Lambda] \in \mathbb{R}^{N \times K\Lambda}$, and let $\mathbf{q} \stackrel{\text{def}}{=} \mathbf{q}(\mathbf{y}) \stackrel{\text{def}}{=} [\mathbf{q}_1^T, \mathbf{q}_2^T, \dots, \mathbf{q}_\Lambda^T]^T \in \mathbb{R}^{K\Lambda}$. We can then estimate \mathbf{x} using the following reformulated new inverse problem

$$\mathbf{q} \approx \mathbf{B}^T \mathbf{x}, \quad (5.3)$$

by solving the corresponding regularized reconstruction problem

$$\hat{\mathbf{x}} = \arg \min_{\mathbf{x} \in [0,1]^N} \|\mathbf{q} - \mathbf{B}^T \mathbf{x}\|^2 + \lambda \varphi(\mathbf{x}), \quad (5.4)$$

with $\varphi(\mathbf{x})$ chosen as the TV-seminorm $\|\mathbf{x}\|_{\text{TV}}$. The regularization is not essential. As we show experimentally, if $K\Lambda$ is sufficiently large, $\varphi(\mathbf{x})$ is not required. Note that solving the original problem directly using the $\|\mathbf{x}\|_{\text{TV}}$ regularizer failed to recover the structure of the object (Figure 5.1).

Local information. As the number of mesh subspaces Λ grows large and their direct sum approaches the image’s ambient space, \mathbb{R}^N , we should expect the performance to deteriorate. To see this consider solving (5.4) without the regularization. This leads to a pseudoinverse solution, $(\mathbf{B}^T)^\dagger \mathbf{q}$, whose stability is indicated by the reciprocal of the lowest singular value, $\sigma_{\min}(\mathbf{B})^{-1}$. Empirically we observe that $\sigma_{\min}(\mathbf{B})^{-1}$ grows large as the number of subspaces increases which reflects the fact that the full-resolution reconstruction remains ill-posed despite the learning step being easier.

Estimates of individual subspace projections give correct *local* information. They convert possibly non-local measurements (e.g. integrals along curves in tomography) into local ones. The key is that these local averages (subspace projection coefficients) can be estimated accurately (see Section 5.3).

To illustrate the impact of this, consider a simple numerical experiment with our reformulated problem, $\mathbf{q} = \mathbf{B}^T \mathbf{x}$, where \mathbf{x} is an all-zero image with a few pixels “on”. For the sake of clarity, we assume the coefficients \mathbf{q} are perfect. Recall that \mathbf{B} is a block matrix comprising Λ subspace bases stacked side by side. It is a random matrix because the subspaces are generated at random, and therefore the reconstruction $\hat{\mathbf{x}} = (\mathbf{B}^T)^\dagger \mathbf{q}$ is also random. We approximate $\mathbb{E} \hat{\mathbf{x}}$ by simulating a large number of Λ -tuples of meshes and averaging the obtained reconstructions. The results are shown in Figure 5.4 for different numbers of triangles per subspace, K , and subspaces per reconstruction, Λ . As Λ or K increase, the expected reconstruction becomes increasingly localized around non-zero pixels. The following proposition (proved in Appendix C.2) tells us that this phenomenon can be modeled by a convolution.

Proposition 6. *Let $\hat{\mathbf{x}}$ be the solution to $\mathbf{q} = \mathbf{B}^T \mathbf{x}$ given as $(\mathbf{B}^T)^\dagger \mathbf{q}$. Then*

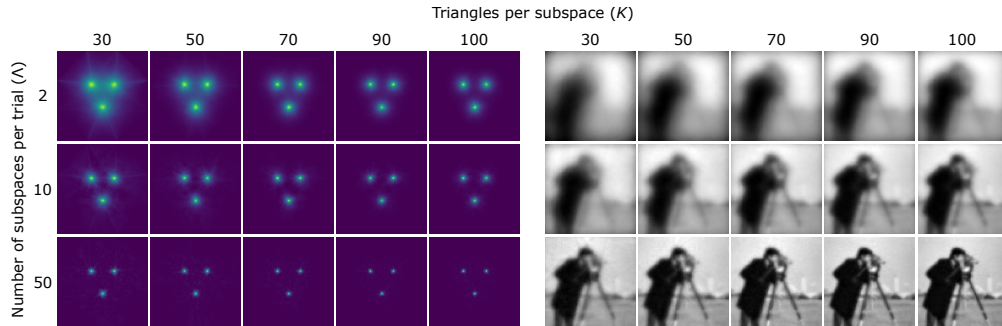


Figure 5.4: Illustration of the expected kernel $\kappa(u, v)$ with varying subspace dimension, K , and number of subspaces, Λ . Reconstruction of a sparse three-pixel image (left) and the cameraman image (right).

there exists a kernel $\tilde{\kappa}(u)$, with u a discrete index, such that $\mathbb{E}\hat{\mathbf{x}} = \mathbf{x} * \tilde{\kappa}$. Furthermore, $\tilde{\kappa}(u)$ is isotropic.

While Figure 5.4 suggests that more triangles are better, we note that this increases the subspace dimension, which makes getting correct projection estimates harder. Instead, we choose to stack more meshes with a smaller number of triangles.

5.3 Numerical results

5.3.1 Application: traveltime tomography

To demonstrate our method's benefits we consider linearized seismic traveltime tomography [91, 90], but we note that the method applies to any inverse problem with scarce data.

In traveltime tomography, we measure $\binom{P}{2}$ wave travel times between P sensors as in Figure 5.5. Travel times depend on the medium property called slowness (inverse of speed) and the task is to reconstruct the spatial slowness map. Image intensities are a proxy for slowness maps—the lower the image intensity the higher the slowness. In the straight-ray approximation, the problem data is modeled as integral along line segments:

$$y(\mathbf{s}_i, \mathbf{s}_j) = \int_0^1 x(t\mathbf{s}_i + (1-t)\mathbf{s}_j) dt, \quad \forall \mathbf{s}_i \neq \mathbf{s}_j \quad (5.5)$$

where $x : \mathbb{R}^2 \rightarrow \mathbb{R}^+$ is the continuous slowness map and $\mathbf{s}_i, \mathbf{s}_j$ are sensor

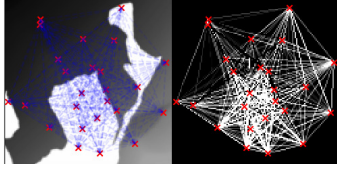


Figure 5.5: Linearized traveltime tomography illustration: On the left we show a sample object, with red crosses indicating 25 sensor locations and dashed blue lines indicating linearized travel paths; on the right we show a reconstruction from $\binom{25}{2} = 300$ measurements by non-negative least squares.

locations. In our experiments, we use a 128×128 pixel grid with 25 sensors (300 measurements) placed uniformly in an inscribed circle, and corrupt the measurements with zero-mean iid Gaussian noise.

5.3.2 Architectures and reconstruction

We generate random Delaunay meshes with 50 triangles each. The corresponding projector matrices compute average intensity over triangles to yield a piecewise constant approximation $\mathbf{P}_{S_\lambda} \mathbf{x}$ of \mathbf{x} . We test two distinct architectures: 1) **ProjNet**, tasked with estimating the projection into a single subspace, and 2) **SubNet**, tasked with estimating the projection over multiple subspaces.²

The ProjNet architecture is inspired by the FBPCConvNet [89] and the U-Net [128], as shown in Figure C.3a in the Appendix. Crucially, we constrain the network output to live in S_λ by fixing the last layer of the network to be a projector, \mathbf{P}_{S_λ} (Figure C.3a). A similar trick in a different context was proposed in [129].

We combine projection estimates from many ProjNets by regularized linear least-squares (5.4) to get the reconstructed object (cf. Figure 5.2) with the regularization parameter λ determined on five held-out images. A drawback of this approach is that a separate ProjNet must be trained for each subspace. This motivates the SubNet (shown in Figure C.3b). Each input to SubNet is the concatenation of a non-negative least squares reconstruction and 50 basis functions, one for each triangle forming a 51-channel input. This approach scales to any number of subspaces which allows us to get visually smoother reconstructions without any further regularization as in (5.4). On the other

²Code available at <https://github.com/swing-research/deepmesh>.

hand, the projections are less precise which can lead to slightly degraded performance.

As a quantitative figure of merit we use signal-to-noise ratio (SNR). The input noise SNR is defined as $10 \log_{10}(\sigma_{\text{signal}}^2/\sigma_{\text{noise}}^2)$ where σ_{signal}^2 and σ_{noise}^2 are the signal and noise variance; the output SNR is defined as $\sup_{a,b} 20 \log_{10}(\|\mathbf{x}\|_2/\|\mathbf{x} - a\hat{\mathbf{x}} - b\|_2)$ with \mathbf{x} the ground truth and $\hat{\mathbf{x}}$ the reconstruction.

We train 130 ProjNets for 130 different subspaces with measurements at various SNRs. Similarly, a single SubNet is trained with 350 different subspaces and the same noise levels. We compare the ProjNet and SubNet reconstructions with a direct U-net baseline convolutional neural network that reconstructs images from their non-negative least squares reconstructions. The direct baseline has the same architecture as SubNet except the input is a single channel non-negative least squares reconstruction like in ProjNet and the output is the target reconstruction. Such an architecture was proposed by [89] and is used as a baseline in recent learning-based inverse problem works [88, 130] and is inspiring other architectures for inverse problems [108]. We simulate the lack of training data by testing on a dataset that is different than that used for training.

5.3.3 Robustness to corruption

To demonstrate that our method is robust against arbitrary assumptions made at training time, we consider two experiments. First, we corrupt the measurements with zero-mean iid Gaussian noise and reconstruct with networks trained at different input noise levels. In Figures 5.6a, C.5 and Table 5.1, we summarize the results with reconstructions of geo images taken from the BP2004 dataset³ and x-ray images of metal castings [131]. The direct baseline and SubNet are trained on a set of 20,000 images from the arbitrarily chosen LSUN bridges dataset [132] and tested with the geophysics and x-ray images. ProjNets are trained with 10,000 images from the LSUN dataset. Our method reports better SNRs compared to the baseline. We note that direct reconstruction is unstable when trained on clean and tested on noisy measurements as it often hallucinates details that are artifacts of the train-

³http://software.seg.org/datasets/2D/2004_BP_Vel_Benchmark/

Average SNR over 102 x-ray images		Training SNR					
		10 dB			∞ dB		
		Direct	ProjNets	SubNet	Direct	ProjNets	SubNet
Testing SNR	10 dB	13.51	14.49	13.92	10.34	12.88	12.85
	∞ dB	13.78	15.38	14.04	16.67	17.23	16.86

Table 5.1: Average reconstruction SNR for various training and testing SNR combinations.

ing data. For applications in geophysics, it is important that our method correctly captures the shape of the cavities. The direct inversion produces sharp but incorrect geometries (see outlines in Figure 5.6a).

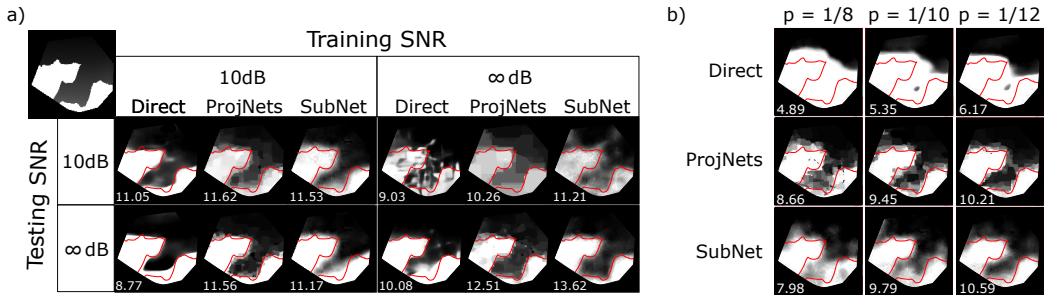


Figure 5.6: a) Reconstructions for different combinations of training and testing input SNR. The output SNR is indicated for each reconstruction. Our method stands out when the training and testing noise levels do not match; b) reconstructions with erasures with probability $\frac{1}{8}$, $\frac{1}{10}$ and $\frac{1}{12}$. The reconstructions are obtained from networks which are trained with input SNR of 10 dB. The direct network cannot produce a reasonable image in any of the cases.

Second, we consider a different corruption mechanism where traveltime measurements are erased (set to zero) independently with probability $p \in \{\frac{1}{12}, \frac{1}{10}, \frac{1}{8}\}$, and use networks trained with 10 dB input SNR on the LSUN dataset. Figure 5.6b and Table 5.2 summarizes our findings. Unlike with Gaussian noise (Figure 5.6a), the direct method completely fails to recover coarse geometry in all test cases. In our entire test dataset of 102 x-ray images, there is not a single example where the direct network captures a geometric feature that our method misses. This demonstrates the strengths of our approach. For more examples of x-ray images, see Appendix C.4.2.

Average SNR over 102 x-ray images	$p = \frac{1}{8}$	$p = \frac{1}{10}$	$p = \frac{1}{12}$
Direct	9.03	9.62	10.06
ProjNets	11.09	11.70	12.08
SubNet	11.33	11.74	11.99

Table 5.2: Average SNR values for reconstructions from measurements with erasure probability, p . All networks were trained for 10dB noisy measurements on the LSUN bridges dataset. Refer to Appendix C.4.2 for actual reconstructions.

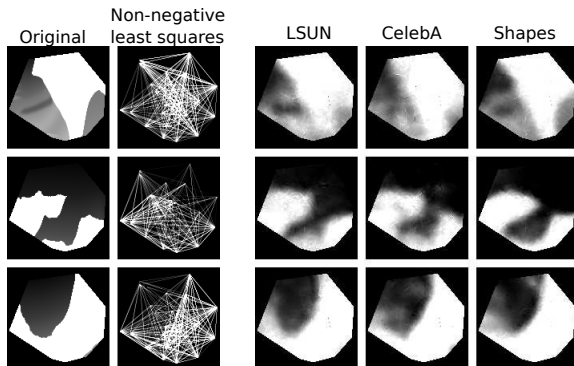


Figure 5.7: Reconstructions from networks trained on different datasets (LSUN, CelebA and Shapes) with 10dB training SNR.

5.3.4 Robustness against dataset overfitting

Figure 5.7 illustrates the influence of the training data on reconstructions. Training with LSUN, CelebA [133] and a synthetic dataset of random overlapping shapes (see Figure C.4 in Appendix for examples) all give comparable reconstructions—a desirable property in applications where real ground truth is unavailable.

We complement our results with reconstructions of checkerboard phantoms (standard resolution tests) and x-rays of metal castings in Figure 5.8. We note that in addition to better SNR, our method produces more accurate geometry estimates, as per the annotations in the figure.

5.4 Summary

We proposed an approach to regularize ill-posed inverse problems in imaging. The key idea is to decompose an unstable inverse mapping into a collection of stable mappings which only estimate low-dimensional projections of the

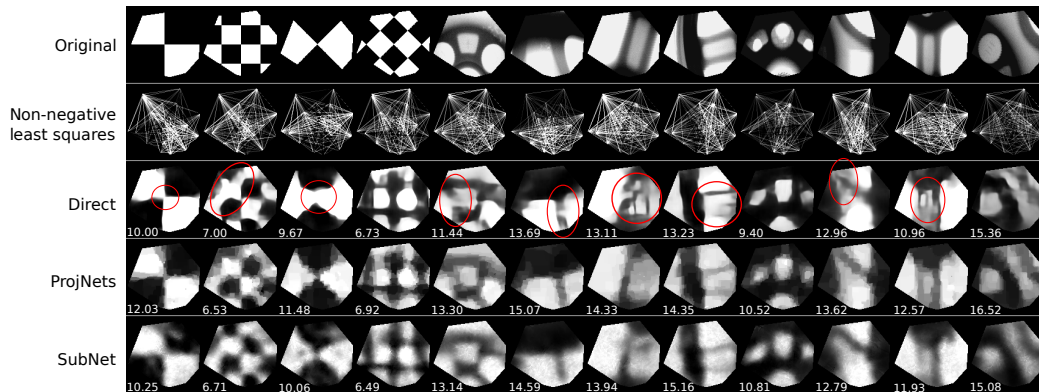


Figure 5.8: Reconstructions on checkerboards and x-rays with 10dB measurement SNR tested on 10dB trained networks. Red annotations highlight where the direct net fails to reconstruct correct geometry.

object. By using piecewise-constant Delaunay subspaces, we showed that the projections can indeed be accurately estimated. Combining the projections leads to a deconvolution-like problem. Compared to directly learning the inverse map, our method is more robust against noise and corruptions. We also showed that regularizing via projections allows our method to generalize across training datasets. Our reconstructions are better both quantitatively, in terms of SNR, and qualitatively, in the sense that they estimate correct geometric features even when measurements are corrupted in ways not seen at training time.

CHAPTER 6

DIFFERENTIABLE UNCALIBRATED IMAGING

In Chapter 4, we saw that accounting for phase retrieval operator uncertainties leads to better reconstructions. In this chapter, we propose a general framework to account for errors in the operator which are caused by measurement coordinate uncertainties. Our framework jointly recovers the unknown measurement coordinates of the true operator (**E2**), and the reconstructed image. This is done by learning a continuous representation of the measurements (**E1**).

6.1 Measurement coordinates

In computational imaging a physical process \mathcal{A} such as 2D computed tomography (CT) relates the object we want to image, x , to an observable field y . Both x and y are naturally functions of continuous coordinates: x could be a density over 2D spatial coordinates (e.g., $[0, 1]^2$), and y a sinogram over angles and 1D projection coordinates (e.g., $[0, \pi) \times [-1, 1]$).

In practice, however, we have finite sensors. Denoting the space of continuous measurement coordinates by Ω , the sensors sample the field y at $\tilde{\boldsymbol{\mu}} = (\tilde{\mu}_1, \dots, \tilde{\mu}_M) \in \Omega^M$, such that the observed measurements $\mathbf{y} \in \mathbb{R}^M$ follow

$$\mathbf{y} = \mathbf{A}_{\tilde{\boldsymbol{\mu}}}(x) + \boldsymbol{\eta}, \tag{6.1}$$

where

$$\mathbf{y} \stackrel{\text{def}}{=} \mathbf{y}(\tilde{\boldsymbol{\mu}}) \stackrel{\text{def}}{=} [y(\tilde{\mu}_1), \dots, y(\tilde{\mu}_M)]^T, \tag{6.2}$$

This work was previously published in S. Gupta, V. Debarnot, K. Kothari, and I. Dokmanić, “Differentiable Uncalibrated Imaging,” arXiv preprint arXiv:2211.10525, 2022 [134] and is adapted here with permission.

and $\boldsymbol{\eta} \in \mathbb{R}^M$ is the measurement noise. The discrete forward operator $\mathbf{A}_{\tilde{\boldsymbol{\mu}}}$ parameterized by $\tilde{\boldsymbol{\mu}}$ samples the output of \mathcal{A} . In computational imaging we work with a discretization or some other finite-dimensional approximation of x denoted by $\boldsymbol{x} \in \mathbb{R}^N$. Hereafter, we let $\mathbf{A}_{\tilde{\boldsymbol{\mu}}}$ act on \boldsymbol{x} rather than on x .

This chapter addresses the scenario where the true measurement coordinates $\tilde{\boldsymbol{\mu}}$ are only approximately known: the imaging system is out of calibration. Consequently, the true operator $\mathbf{A}_{\tilde{\boldsymbol{\mu}}}$ is unknown and we work with an operator $\mathbf{A}_{\boldsymbol{\mu}}$ for measurement coordinates $\boldsymbol{\mu} = (\mu_1, \dots, \mu_M)$ which *would* be correct if the system was calibrated. The assumed measurement coordinates $\boldsymbol{\mu}$ are related to the *unknown* true measurement coordinates $\tilde{\boldsymbol{\mu}}$ by small perturbations.

Not accounting for the mismatch between $\tilde{\boldsymbol{\mu}}$ and $\boldsymbol{\mu}$ can lead to a poor reconstruction. The gist of our method is to learn a representation of the measurement space that can be evaluated and differentiated at arbitrary measurement coordinates $\boldsymbol{\mu}$. This gives us measurements $\mathbf{y}(\boldsymbol{\mu})$ that are then well-suited for a reconstruction method that uses $\boldsymbol{\mu}$. Our proposed framework enables us to

1. Jointly reconstruct the image \boldsymbol{x} and learn the true unknown measurement coordinates $\tilde{\boldsymbol{\mu}}$ by using gradient-based optimization.
2. Learn continuous measurement representations that take measurement coordinates as input. These representations can be evaluated at $\boldsymbol{\mu}$ and are in essence interpolations of the discrete measurements with unknown interpolation knots.
3. Leverage standard differentiable image reconstruction methods that exist for the assumed $\boldsymbol{\mu}$ even though the observations correspond to unknown $\tilde{\boldsymbol{\mu}}$. This helps learn consistent measurement representations.

A strength of our approach is that we can use any continuous interpolation method that admits backpropagation to input coordinates. To showcase this, we use implicit neural networks (neural fields) but also develop fully-differentiable variable-knot splines, which allow us to optimize spline control points and control point weights. We show that splines perform as well as implicit neural networks while being faster to fit and simpler to interpret. A key property of both these representation types is that we can perform

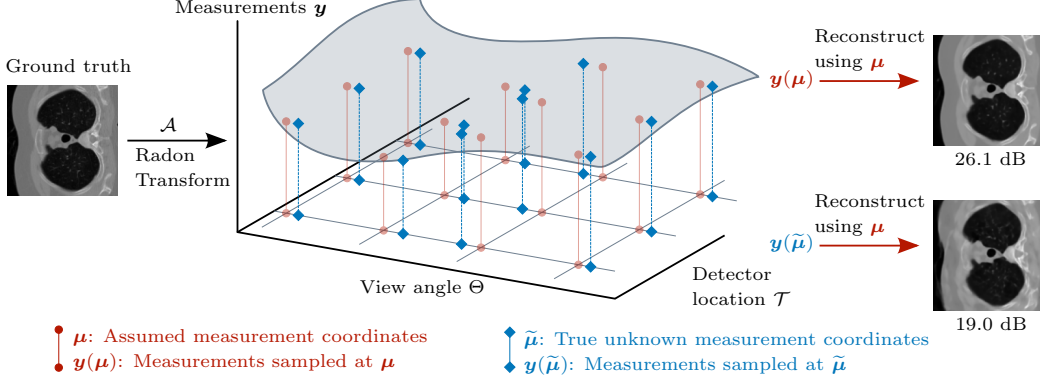


Figure 6.1: Visualization of the parameter mismatch between $\tilde{\mu}$ and μ in 2D CT imaging. A mismatch in the view angles between μ and $\tilde{\mu}$ can produce a significant drop in reconstruction quality.

automatic differentiation with respect to their input coordinates and recover $\tilde{\mu}$ by gradient-based optimization.

6.1.1 Example: Computed tomography

We illustrate our framework with 2D CT, where we measure parallel beam projections of an image at different view angles in a detector plane. Let $\Omega = [0, \pi) \times \mathbb{R}$, $x \in L^2([-1, 1]^2)$,¹ and $y \in L^2(\Omega)$. The measured projections \mathbf{y} are obtained from a finite set of discrete view angles $\tilde{\Theta} = (\tilde{\theta}_1, \dots, \tilde{\theta}_J)$, and are sampled at a finite set of discrete detector locations $\tilde{\mathcal{T}} = (\tilde{t}_1, \dots, \tilde{t}_K)$. The true set of unknown measurement coordinates are then $(\tilde{\mu}_m)_{m=1}^M = \tilde{\Theta} \times \tilde{\mathcal{T}}$ with $M = J \cdot K$. Furthermore, $\mathbf{A}_{\tilde{\mu}}$ is given by the discrete Radon transform for view angles $\tilde{\Theta}$ and detector locations $\tilde{\mathcal{T}}$. Many other computational imaging applications such as electron cryotomography (CryoET), magnetic resonance imaging, and optical microscopy can be parameterized similarly.

The parameter mismatch between $\tilde{\mu}$ and μ and its severe impact is illustrated in Figure 6.1. We use a state-of-the-art reconstruction method that computes a filtered backprojection estimate with μ and feeds it into a UNet deep neural network to obtain a reconstruction of \mathbf{x} [128, 89]. The neural network is trained in a supervised manner using training data that is also generated using μ . When there is parameter mismatch, the true view angles differ from the assumed view angles and therefore $\tilde{\mu} \neq \mu$. Figure 6.1 shows

¹We denote by $L^2(\mathcal{M})$ the space of square integrable functions on \mathcal{M} .

that in this case the reconstruction method produces a degraded reconstruction from both a visual and SNR evaluation. The middle column of Figure 6.2 shows this for another ground truth image and number of view angles. The last column of Figure 6.2 also shows that when there is no parameter mismatch ($\tilde{\mu} = \mu$), the same reconstruction method performs strongly.

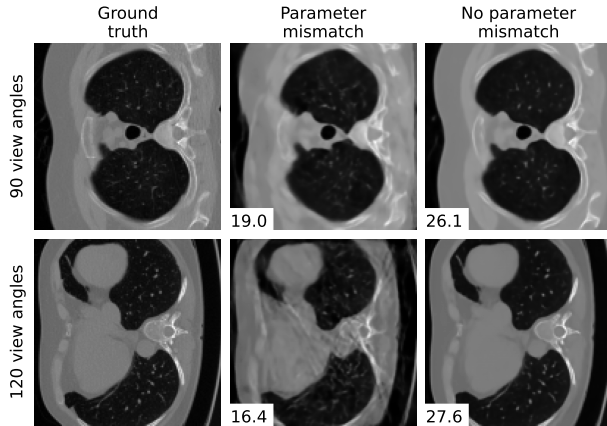


Figure 6.2: 2D CT reconstructions with 90 and 120 view angle. The SNRs (in dB) are written in the bottom-left corners. When there is parameter mismatch because the true and assumed view angles differ, the reconstruction method performs poorly and there is a clear degradation.

6.1.2 Chapter overview and contributions

This chapter is organized as follows. In Section 6.2 we present the optimization problem which models joint calibration and image reconstruction. We first model measurements as continuous functions and then show how to learn these measurement representations. To help better understand measurement representations, we select implicit neural networks and splines and show how they can be used as measurement representations. After this, Section 6.3 experimentally verifies that when there is measurement coordinate uncertainty, our proposed method yields significantly improved reconstructions. The experimental verification is done by solving 2D and 3D CT imaging problems. We show that both implicit neural networks and splines perform comparably as measurement representations. We summarize the chapter in Section 6.4.

6.1.3 Related work

Many of the current state-of-the-art methods for solving imaging inverse problems are based on deep learning [101, 135]. Popular supervised approaches use the forward operator to obtain an initial estimate, which is then enhanced by a neural network (reconstruction method in Figure 6.2 for example) [89, 108, 17, 136]. Unrolling iterative methods or replacing optimization components with deep networks is another established approach [117, 113, 137, 138, 139, 140]. A different direction involves requiring the inverse problem solution to lie in the range of a generative neural network [119, 141, 142]. In general all these methods utilize a forward operator in some way but do not account for measurement coordinate uncertainty. This can severely impact the reconstruction as shown in Figure 6.2. In this chapter we propose to address this problem by reevaluating the measurements at the measurement coordinates that deep neural network reconstruction methods are designed for.

While much less common, there are some recent deep learning approaches which address measurement coordinate uncertainty. Gilton et al. explored fine-tuning a neural network that was trained with measurements sampled at μ to work well with test measurements sampled at unknown $\tilde{\mu}$ [143]. Our method is different: we reevaluate the measurements via a measurement representation for a single example so we do not need to fine-tune. Another appealing approach is to train a neural network on a family of operators with different parameterizations [144]. These methods always induce a tradeoff between the reconstruction quality and the variety of forward maps they are trained on, and they only work well for the distribution of perturbations seen at training time. There is also the challenge of dataset generation, especially in compute-intensive problems. We mitigate this tradeoff by using consistency to identify the true measurement coordinates.

Continuous representations have been used to solve a variety of science and engineering problems. To the best of our knowledge, no prior work has used them for imaging with measurement coordinate uncertainty. Splines have long been used to model surfaces and curves in geometry [125, 126, 145, 146]. Moreover, differentiable splines with parameters that can be fitted using automatic differentiation have also been developed [147]. Recently deep learning methods called neural fields or implicit neural networks have proven to

be extremely good and efficient continuous representations [148]. They have been employed in a broad spectrum of applications ranging from representing geometry via signed distance functions [149, 150] to solving partial differential equations [151]. Implicit neural networks have also been used to solve tomographic imaging inverse problems [152, 153]. In particular, Sun et al. also used them to represent 2D CT measurements [154]. Rather than for calibration, they use implicit networks to upsample measurements for downstream reconstruction by a deep neural network. While upsampling may also be performed by traditional tools such as splines, our framework relies on the differentiability of the used representations—both neural and spline-based—with respect to the input coordinates.

This chapter is also related to the broader theme of inverse problems with “noisy” forward operators. On one end of the spectrum there are inverse problems where the operator (and $\tilde{\boldsymbol{\mu}}$) is perfectly known, and, at the other extreme, there are blind inverse problems where the operator and $\tilde{\boldsymbol{\mu}}$ are completely unknown. Related blind imaging problems are tomography with unknown view angles [155, 156] and cryo-electron microscopy with unknown projection angles [157]. In between the extremes, there are semi-blind inverse problems where the operator and $\tilde{\boldsymbol{\mu}}$ are approximately known. Total least squares approaches that perturb an assumed operator such as the work in Chapter 4 [58, 59, 56] and our proposed measurement coordinate-based framework fall under this category.

6.2 Differentiable framework

We wish to learn a continuous representation of the measurement space so that we can sample it at specific measurement coordinates and obtain the corresponding measurements. We model the measurement representations as continuous functions $r_{\boldsymbol{\varphi}}(\cdot)$ that take measurement coordinates as input and map them to the corresponding sampled measurements. The learnable parameters, $\boldsymbol{\varphi} \in \Phi$ where Φ is the space of feasible parameters, are optimized so that

$$r_{\boldsymbol{\varphi}}(\boldsymbol{\omega}) \approx \mathbf{y}(\boldsymbol{\omega}) \tag{6.3}$$

where $\omega \in \Omega$ is a measurement coordinate and $\mathbf{y}(\omega) \in \mathbb{R}$ is the sample from the measurement space at measurement coordinate ω . For convenience we also denote a batch evaluation of $r_\varphi(\cdot)$ as

$$\begin{aligned} R_\varphi(\boldsymbol{\omega}) &\stackrel{\text{def}}{=} [r_\varphi(\omega_1), \dots, r_\varphi(\omega_Q)]^T \\ &\approx [\mathbf{y}(\omega_1), \dots, \mathbf{y}(\omega_Q)]^T \end{aligned} \quad (6.4)$$

where $\boldsymbol{\omega} = (\omega_1, \dots, \omega_Q) \in \Omega^Q$ and $R_\varphi(\boldsymbol{\omega}) \in \mathbb{R}^Q$. We also require $r_\varphi(\cdot)$ to be differentiable with respect to φ and its input so that we can use gradient-based optimization to estimate φ and the unknown measurement coordinates.

6.2.1 Joint optimization objective

We want $r_\varphi(\cdot)$ to accurately produce samples from the space of measurements. Since we only observe measurements \mathbf{y} at measurement coordinates $\tilde{\boldsymbol{\mu}}$ in (6.1), we require

$$R_\varphi(\tilde{\boldsymbol{\mu}}) \approx \mathbf{y} \quad (6.5)$$

However, since $\tilde{\boldsymbol{\mu}}$ is unknown we cannot use it to verify the accuracy of $r_\varphi(\cdot)$. This motivates us to jointly learn the representation parameters φ and the unknown measurement coordinates $\tilde{\boldsymbol{\mu}}$ by minimizing a measurement fitting loss,

$$\mathcal{L}_{\text{fitting}}(\boldsymbol{\nu}, \varphi) \stackrel{\text{def}}{=} \|\mathbf{y} - R_\varphi(\boldsymbol{\nu})\|_2^2, \quad (6.6)$$

with respect to the input $\boldsymbol{\nu} \in \Omega^M$ and φ . Recall $\mathbf{y} \in \mathbb{R}^M$ is defined by (6.1).

Learning $r_\varphi(\cdot)$ by minimizing only $\mathcal{L}_{\text{fitting}}(\boldsymbol{\nu}, \varphi)$ with respect to both the representation's parameters and input would not in general result in an accurate measurement representation because there are too many degrees of freedom. Therefore, we regularize and control φ by enforcing $r_\varphi(\cdot)$ to be consistent with reconstructions that could be obtained by using its output. This is done by minimizing a consistency loss with respect to φ ,

$$\mathcal{L}_{\text{consistency}}(\varphi) \stackrel{\text{def}}{=} \|R_\varphi(\boldsymbol{\mu}) - \mathbf{A}_\mu G_\mu(R_\varphi(\boldsymbol{\mu}))\|_2^2, \quad (6.7)$$

where $G_{\boldsymbol{\mu}} : \mathbb{R}^M \rightarrow \mathbb{R}^N$ is a differentiable reconstruction method that was designed using measurement coordinates $\boldsymbol{\mu}$. Putting everything together, our complete joint optimization objective is

$$\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\varphi}} = \arg \min_{\boldsymbol{\nu} \in \Omega^M, \boldsymbol{\varphi} \in \Phi} \lambda_1 \mathcal{L}_{\text{fitting}}(\boldsymbol{\nu}, \boldsymbol{\varphi}) + \lambda_2 \mathcal{L}_{\text{consistency}}(\boldsymbol{\varphi}) \quad (6.8)$$

where $\lambda_1, \lambda_2 \in \mathbb{R}_{\geq 0}$ are tunable weights. As the assumed coordinates $\boldsymbol{\mu}$ are close to the true unknown coordinates $\tilde{\boldsymbol{\mu}}$, we initialize $\boldsymbol{\nu}$ to $\boldsymbol{\mu}$. After completing the optimization (6.8), the learned coordinates $\hat{\boldsymbol{\mu}}$ are close to $\tilde{\boldsymbol{\mu}}$ (verified in Section 6.3), and the final reconstruction and estimate of $\boldsymbol{x} \in \mathbb{R}^N$ is given by

$$\hat{\boldsymbol{x}} = G_{\boldsymbol{\mu}}(R_{\boldsymbol{\varphi}}(\boldsymbol{\mu})). \quad (6.9)$$

6.2.2 Leveraging reconstruction methods

A key aspect of our framework is that consistency and the final reconstruction are obtained by using reconstruction method $G_{\boldsymbol{\mu}}(\cdot)$ that was designed using measurement coordinates $\boldsymbol{\mu}$ even though the observations in (6.1) are sampled at measurement coordinates $\tilde{\boldsymbol{\mu}}$. This provides significant flexibility and allows us to incorporate a variety of reconstruction methods. For example, the reconstruction method may be a relatively straightforward adjoint or pseudoinverse operation. Alternatively, it can be a more complex neural network that provides state-of-the-art reconstructions with measurements from $\boldsymbol{\mu}$ (reconstruction method in Figure 6.2 is one example). Our framework is particularly advantageous in this case because it may be cumbersome to retrain a neural network for different measurement coordinates.

6.2.3 Measurement representations

In order to better understand how to use the framework to solve real imaging problems, we now pick implicit neural networks and splines and explain how they are suitable measurement representations. While we consider these, we emphasize that our framework is general and is not restricted to these representation types.

Implicit neural representations. Implicit neural representations are deep feedforward neural networks that represent discrete signals as continuous functions. When used as the measurement representation $r_\varphi(\cdot)$ in (6.3), φ are the trainable network parameters. The input to the network are measurement coordinates and the output are the corresponding measurements. Implicit neural networks have previously been used to represent measurements when there is no measurement uncertainty [154]. In this case the network input was not optimized and consistency (6.7) was not enforced.

As implicit neural representations are neural networks, we can use automatic differentiation to calculate their gradients with respect to φ and their input coordinates to solve (6.8). In this chapter, we use an architecture comprising a Fourier feature mapping layer followed by standard fully-connected layers [150, 154].

Differentiable splines. Splines use locally supported basis functions to represent signals as a continuous surface. In this chapter, we focus on Non-uniform Rational Basis Splines (NURBS) because of their ability to model complex surfaces [145, 158, 159]. To aid understanding we explain NURBS using the 2D CT imaging example that was introduced in Section 6.1.1. The measurement coordinates $\boldsymbol{\mu}$ are the Cartesian product of assumed view angles and assumed detector locations, $\Theta \times \mathcal{T}$ where $\Theta = (\theta_1, \dots, \theta_J)$ and $\mathcal{T} = (t_1, \dots, t_K)$. The NURBS surface $s_\varphi(\cdot)$ with parameters φ evaluated at measurement coordinate $\boldsymbol{\mu}' = [\theta', t']^T \in [0, \pi] \times \mathbb{R}$ is then

$$s_\varphi(\boldsymbol{\mu}') = \sum_{j=0}^{J-1} \sum_{k=0}^{K-1} b_{j,k}(\boldsymbol{\mu}') \mathbf{p}_{j,k} \quad (6.10)$$

where $b_{j,k}(\cdot) \in \mathbb{R}$ are scalar-valued rational basis functions with local support and $\mathbf{p}_{j,k} \in \mathbb{R}^3$ are control point vectors. The NURBS are parameterized by the weight parameters $w_{j,k} \in \mathbb{R}$ of the rational basis functions, and the control point vectors. This gives $\varphi = \{w_{j,k}\}_{j=0, k=0}^{J-1, K-1} \cup \{\mathbf{p}_{j,k}\}_{j=0, k=0}^{J-1, K-1}$. Note that the control point vectors are three-dimensional because for each two-dimensional measurement coordinate $\boldsymbol{\mu}'$, there is a corresponding scalar measurement. Consequently, according to (6.10), $s_\varphi(\boldsymbol{\mu}')$ is also three-dimensional. We then establish the following relationship between NURBS surfaces and our mea-

surement representations (6.3) to get a spline measurement representation,

$$r_{\varphi}(\omega) = s_{\varphi}^{\diamond}(\omega), \quad (6.11)$$

where $s_{\varphi}^{\diamond}(\cdot)$ denotes the value of the measurement dimension of $s_{\varphi}(\cdot)$.

It has recently been shown that automatic differentiation can be used to learn spline parameters [147]. Hence, we develop differentiable spline representations and use gradient-based optimization to learn the NURBS parameters φ and their input measurement coordinates. These differentiable splines fit into our framework straightforwardly as measurement representations (6.11), and we can use them to solve (6.8).

In our experiments, we carefully initialize the spline parameters φ and then learn them: $w_{j,k}$ is initialized to one and control point vectors $\mathbf{p}_{j,k}$ are initialized using the assumed measurement coordinates and their corresponding observed measurements, $\mathbf{p}_{j,k} = \left[\theta_j, t_k, \mathbf{y}([\tilde{\theta}_j, \tilde{t}_k]^T) \right]^T \in \mathbb{R}^3$. Additionally, in our experiments we also extend (6.10) to higher dimensional measurement coordinates (see Section 6.3.2). We provide further details on NURBS and their implementation are provided in Appendices D.2 and D.3.

6.3 Experimental verification

We experimentally verify our framework by solving 2D and 3D CT imaging problems. These experiments demonstrate that our framework can be used to solve imaging problems with different measurement coordinate dimensions. Furthermore, we exhibit the flexibility of our method by using implicit neural networks and splines as measurement representations.²

We use SNR (dB) to quantify the measurement noise and measurement coordinate uncertainty. SNR is calculated by

$$\text{SNR}(\mathbf{c}, \mathbf{d}) = -20 \log_{10} \left(\frac{\|\mathbf{c} - \mathbf{d}\|_2}{\|\mathbf{d}\|_2} \right). \quad (6.12)$$

If we let $\mathbf{y} \in \mathbb{R}^M$ denote the observed measurements as in (6.1) and let $\tilde{\mathbf{y}}$ denote the unobserved noiseless measurements, the measurement noise level is

²Code available at https://github.com/swing-research/differentiable_uncalibrated_imaging.

$\text{SNR}(\mathbf{y}, \tilde{\mathbf{y}})$. For each measurement coordinate dimension, we keep the measurement coordinate uncertainty level the same across all M measurement coordinates. If we let $\tilde{\mu}_{m,d} \in \mathbb{R}$ and $\mu_{m,d} \in \mathbb{R}$ denote the d th dimension of the m th true and assumed measurement coordinates, the measurement coordinate uncertainty level for the d th dimension is $\text{SNR}(\tilde{\mu}_{m,d}, \mu_{m,d})$. The measurement noise and measurement coordinate uncertainty is simulated with zero-mean iid Gaussian noise with variance adjusted to achieve a target SNR level.

Due to their state-of-the art performance when there is no measurement coordinate uncertainty, we use deep neural networks for the reconstruction method, $G_{\boldsymbol{\mu}}(\cdot)$, in (6.8). For 2D CT we use a 2D Unet [89] and for 3D CT we use a 3D Unet [160]. Following standard practice, a preprocessing step applies the pseudoinverse of the imaging operator to the measurements to produce an initial image estimate [89]. These networks are then trained in a supervised manner to map the initial estimates to ground truth images. The training data generation and preprocessing step are done with the assumed measurement coordinates $\boldsymbol{\mu}$. The measurements in the training data are noisy and in each experiment we use a Unet whose training measurement noise level matches the measurement noise level of the obtained measurements.

As mentioned in Section 6.2, the solution, $\hat{\mathbf{x}}$, is given by (6.9). We use SNR to evaluate the solution quality, $\text{SNR}(\hat{\mathbf{x}}, \mathbf{x})$. We compare $\hat{\mathbf{x}}$ against baseline reconstructions that are obtained by directly using the obtained measurements with the reconstruction Unets

$$\mathbf{x}_{\text{baseline}} = G_{\boldsymbol{\mu}}(\mathbf{y}). \quad (6.13)$$

Recall, the obtained measurements correspond to measurement coordinates $\tilde{\boldsymbol{\mu}}$.

Appendix D.3 contains further hyperparameter and implementation details.

6.3.1 Two-dimensional CT imaging

In 2D CT imaging, one-dimensional projections of a two-dimensional object at different view angles are collected and our goal is to reconstruct an im-

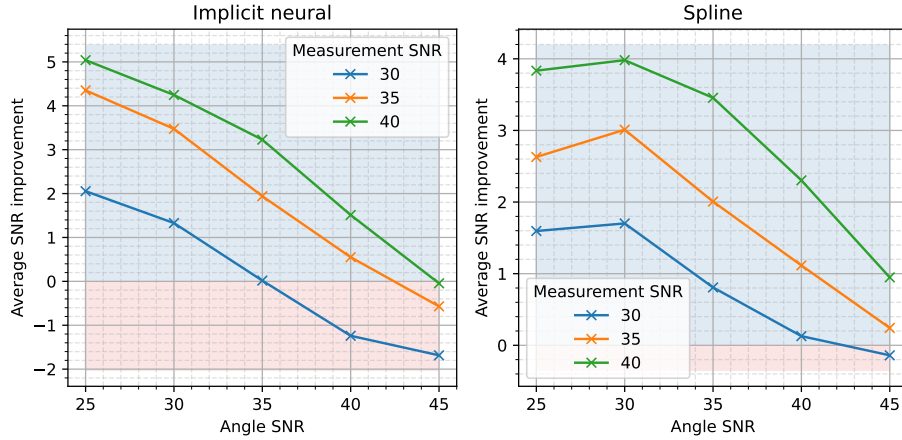
age of the object from these projections. For this set of experiments, the detectors are uniformly spaced on the normalized interval $[0, 1]$ and have no uncertainty. The assumed view angles are uniformly spaced on the interval $[0, \pi]$, and the true view angles have an unknown perturbation from these assumed view angles due to experimental incertitude. The measurement coordinates space is $\Omega = [0, \pi] \times [0, 1]$. As there is only uncertainty in the view angle dimension and not in the detector location dimension, we only optimize the view angle dimension of the measurement coordinates $(\nu_m)_{m=1}^M$ in (6.8).

We explore the performance of our method compared to the baseline which does not account for measurement coordinate uncertainty. We use images from the the LoDoPaB-CT tomography dataset resized to 128×128 [161]. From this dataset, 35,000 samples were used to train the image reconstruction 2D Unet $G_\mu(\cdot)$. Test images from this dataset are used in this section to verify our framework.

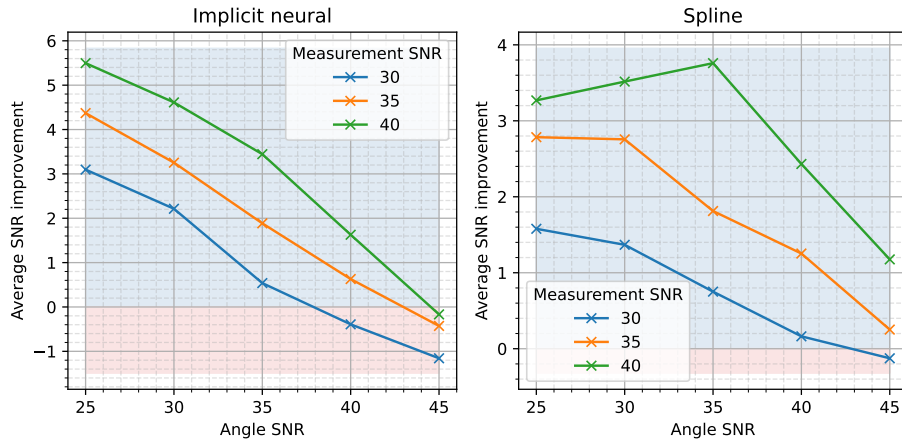
Combinations of measurement and coordinate error. To determine how our framework performs under different settings, we consider different combinations of measurement noise and measurement coordinate uncertainty (in view angles). We do trials over 25 different test images, and in each trial different measurement noise and view angle perturbations are used.

We evaluate the performance of our framework relative to the baseline by calculating the average reconstruction SNR improvement over the baseline. Figure 6.3a shows the performance for 90 view angles. The performance trends are similar for both implicit neural and spline measurement representations. For a given measurement noise SNR, our method shows increasing improvements as the angle SNR decreases (measurement coordinate uncertainty increases). This shows that our method handles measurement coordinate uncertainty well, especially when the uncertainties begin to dominate over measurement noise. We can also see that for a fixed angle SNR, the performance gains decrease as measurement SNR decreases and measurement errors becomes more dominant. Figure 6.3b shows that the same trends hold when there are 120 view angles.

In Figure 6.4, we show some example ground truth, pseudoinverse filtered backprojection (FBP) and baseline reconstructions with their SNRs when there are 90 view angles. The reconstructions using our framework with implicit neural and spline measurement representations are also shown. Com-



(a) 90 view angles



(b) 120 view angles

Figure 6.3: Average SNR improvement (dB) when solving (6.8) for different combinations of measurement noise SNR and view angle uncertainty SNR. There are 90 and 120 view angles.

pared to the baseline, solutions obtained using our framework have fewer artifacts. Figure D.1 in the Appendix shows these same reconstructions when there are 120 view angles instead.

Learned view angles accuracy. In the next experiment we verify that the learned measurement coordinates, $\hat{\boldsymbol{\mu}}$ in (6.8), are close to the true unknown measurement coordinates $\tilde{\boldsymbol{\mu}}$. As the detector locations have no error, we verify the learned view angles only. We denote the set of true unknown view angles and learned view angles as $\tilde{\Theta}$ and $\hat{\Theta}$. We quantitatively measure the

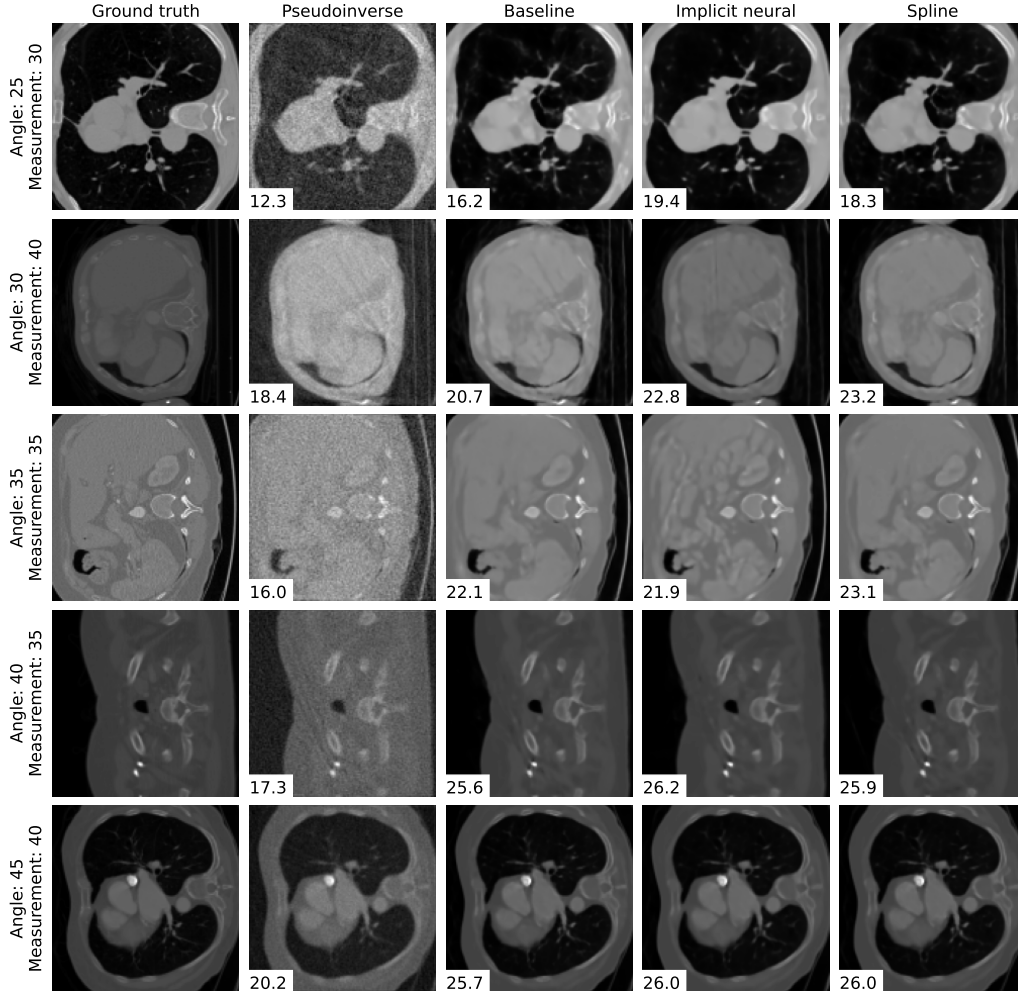


Figure 6.4: Example reconstructions for different measurement noise SNR and view angle uncertainty SNR combinations when there are 90 view angles. The reconstruction SNRs are shown for each reconstruction.

average angle error as

$$\text{Average angle error} = \frac{1}{J} \|\text{sort}(\Theta) - \text{sort}(\hat{\Theta})\|_1, \quad (6.14)$$

where $\text{sort}(\cdot)$ vectorizes and sorts the sets in ascending order and J is the number of view angles.

Figure 6.5 shows how the average angle error changes as the optimization iterations of (6.8) progress. This is shown for one test image with different combinations of measurement noise and measurement coordinate uncertainty when there are 90 view angles. The solid lines are for implicit neural representations and the dashed lines are for spline representations. For both

representation types, the angle error reduces as (6.8) is solved which confirms that our framework learns measurement coordinates that are more accurate than the assumed measurement coordinates which they were initialized with. The average angle error when using the assumed measurement coordinates for reconstruction, as is done in the baseline, is the initial point on the plots. Figure D.2 in the Appendix shows that the same trends hold when there are 120 view angles.

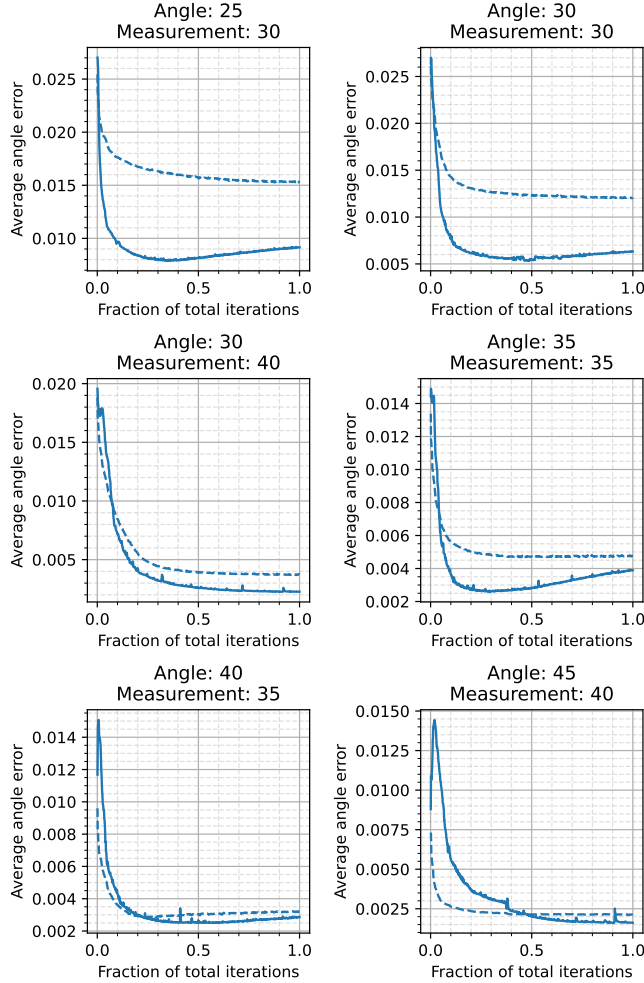


Figure 6.5: Average angle error for one test image with different combinations of measurement noise and measurement coordinate uncertainty when there are 90 view angles. The solid lines are for implicit neural representations and the dashed lines are for spline representations.

Reconstruction with more measurements. Next, we investigate a variant of the main problem considered in this chapter: in addition to the M

true measurement coordinates being unknown, the reconstruction method $G_{\boldsymbol{\mu}}(\cdot)$ is now designed for M' measurements where $M' \geq M$. In this case $\tilde{\boldsymbol{\mu}} = (\tilde{\mu}_1, \dots, \tilde{\mu}_M)$ as before and now $\boldsymbol{\mu} = (\mu_1, \dots, \mu_{M'})$. As the measurement representation $r_{\varphi}(\cdot)$ can be evaluated at any measurement coordinate, we can evaluate (6.7) at M' measurement coordinates.

In this experiment we obtain measurements from 90 view angles. As in the previous experiments, the true view angles are unknown and we initialize the angles of measurements coordinates $\boldsymbol{\nu}$ in (6.6) to be 90 uniformly spaced view angles in the interval $[0, \pi]$. The detector locations have no uncertainty. We consider different values of M' by varying the number of view angles J' . The number of detectors K are not varied which gives $M' = J' \cdot K$. We try two different reconstruction Unets: 1) $G_{\boldsymbol{\mu}}^{J'}(\cdot)$ which was trained with training data having J' view angles uniformly spaced on $[0, \pi]$ and, 2) $G_{\boldsymbol{\mu}}^{90}(\cdot)$ which was trained with training data having 90 view angles uniformly spaced on $[0, \pi]$.³

Table 6.1 shows the average reconstruction SNR over 25 test images. There is 35 dB measurement noise and the unknown true view angles are perturbed by 35 dB from 90 uniformly spaced view angles. With Unet $G_{\boldsymbol{\mu}}^{J'}(\cdot)$, the performance fluctuates. The performance is stable with Unet $G_{\boldsymbol{\mu}}^{90}(\cdot)$. With both reconstruction methods, the best performance is seen when $J' = 90$ ($M' = M$) and evaluating more measurements does not help. This is because the fitting loss (6.6) ensures that $r_{\varphi}(\cdot)$ accurately represents the M obtained measurements which were sampled from the measurement space at coordinates $\tilde{\boldsymbol{\mu}}$. Then, when $M' = M$, because the measurement coordinates used for reconstruction, $\boldsymbol{\mu}$, are a small perturbation away from $\tilde{\boldsymbol{\mu}}$, they are also represented accurately which results in good reconstructions. Furthermore, when $M' = M$, there are enough measurement coordinates that densely cover the measurement space. The accuracy of the measurement coordinates for the extra measurements when $M' > M$ is not enforced by the fitting loss (6.6).

It has been shown that reconstruction with more measurements can help when there is no measurement coordinate uncertainty and the input to the

³The number of Unet training view angles and evaluated view angles J' can be different. This is because the Unet input is computed by a filtered backprojection for J' view angles, which always results in an Unet input with the dimensions of the image being reconstructed.

Table 6.1: Average reconstruction SNR when reconstructing with more measurements.

J'	$G_{\mu}^{J'}(\cdot)$		$G_{\mu}^{90}(\cdot)$	
	Implicit	Spline	Implicit	Spline
90	23.5	23.6	23.5	23.6
110	22.6	23.1	23.7	23.6
120	23.4	23.1	23.8	23.6
130	23.2	22.8	23.7	23.5
150	22.9	22.5	23.8	23.6
180	22.3	22.0	23.9	23.6

reconstruction neural network combines the observed measurements with the measurement representation output [154]. When there is measurement coordinate uncertainty, using the original measurements in the input to the reconstruction neural network can reduce performance as shown by Figure 6.2 and the baseline reconstructions in Figure 6.4.

6.3.2 Three-dimensional CT imaging

Similar to 2D CT imaging, in 3D CT imaging, two-dimensional projections of a three-dimensional volume are collected by tilting it at different tilt angles. The goal is to reconstruct the volume from the the series of projections. Inspired by CryoET imaging, we obtain projections at 60 tilt angles which we assume to be uniformly spaced on the interval $[-\pi/3, \pi/3]$. The true unknown tilt angles are perturbed from these. This setup is challenging because there is a ‘wedge’ of tilt angles for which we do not have projections. The detectors are uniformly spaced on a two-dimensional unit square $[0, 1]^2$ and have no uncertainty. Combining these spaces gives the measurement coordinates space as $\Omega = [-\pi/3, \pi/3] \times [0, 1]^2$. In the same way as the 2D CT experiments, we only optimize the tilt angle dimension of the measurement coordinates $(\nu_m)_{m=1}^M$ in (6.8) because the detector locations have no uncertainty.

To train the reconstruction 3D Unets, we use 3D volumes from the 2019 Brain Tumor Segmentation (BraTS) Challenge dataset [162, 163, 164]. The volumes are resized to be $64 \times 64 \times 64$. We use 478 volumes for training and 15 separate volumes to test and verify our framework.

Combinations of measurement and coordinate error. We consider different combinations of measurement noise and measurement coordinate uncertainty (in tilt angles) in the same way as was done for 2D CT. The trials are done over 15 test volumes and the results are shown in Figure 6.6. We see that the performance trends for both implicit neural and spline representations are similar. Furthermore, the performance is consistent with what was seen for the 2D CT problem (Figure 6.3)—as measurement coordinate uncertainty increases, our framework provides increasing gains.

Figure 6.7 shows two-dimensional slices through some of the test volumes. The ground truth, psuedoinverse, baseline, and our reconstructions are shown with the SNRs of the entire volumes. Similarly, Figure 6.8 shows 3D reconstructions of the same test volumes. Two central orthogonal volume slices have been plotted. Reconstructions using our framework are better than the baseline at recovering the geometric structure of the volumes and have fewer artifacts.

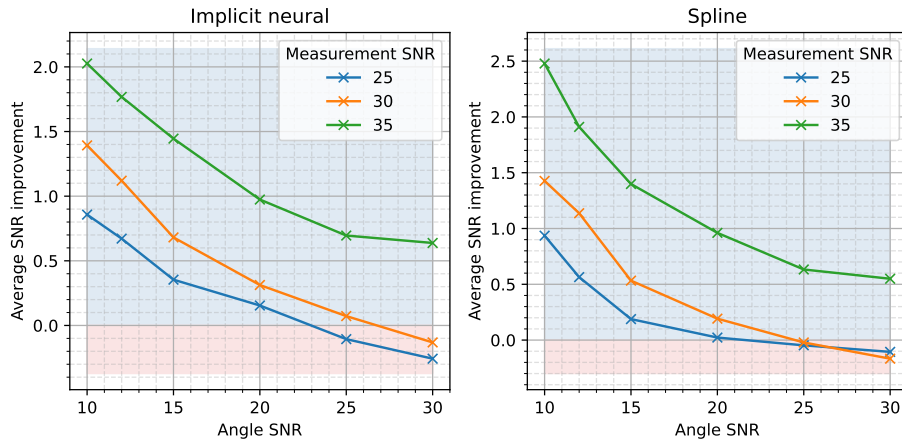


Figure 6.6: Average SNR improvement (dB) when using our framework (6.8) for different combinations of measurement noise SNR and measurement coordinate uncertainty SNR. There are 60 tilt angles on the interval $[-\pi/3, \pi/3]$.

Learned tilt angles accuracy. Next we verify that the learned measurement coordinates are accurate. The uncertainty is only in the tilt angles and we follow the same procedure as when verifying the learned 2D CT view angles in Figure 6.3.1. We use the metric defined in (6.14) and show the results in Figure 6.9 for one test volume with different measurement noise and

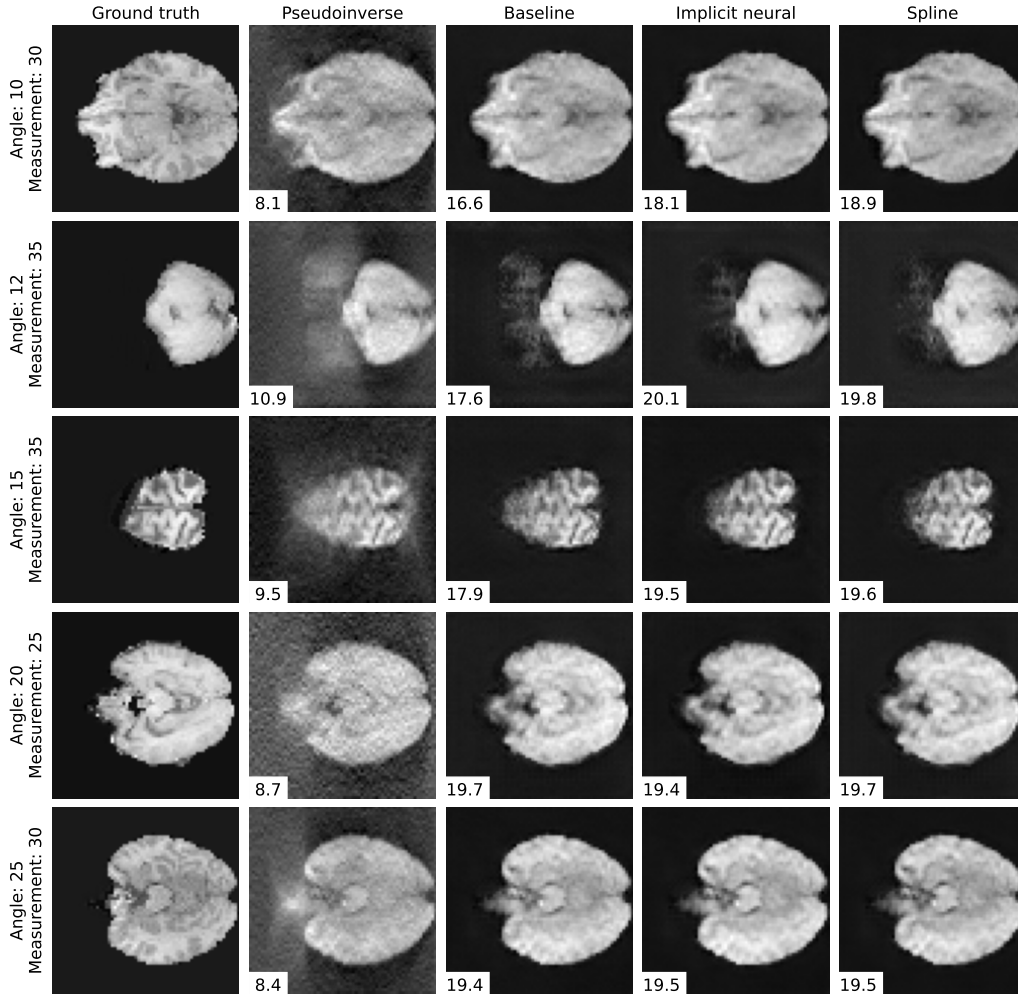


Figure 6.7: Example reconstructed slices through the test volumes. Reconstructions for different measurement noise SNR and measurement coordinate uncertainty SNR are shown. The reconstruction SNRs for each volume are stated.

measurement coordinate uncertainty combinations. Again, the solid lines are for the implicit neural representations and the dashed lines are for the spline representations. For both representation types, the average tilt angle error reduces as the optimization of (6.8) progresses. This demonstrates that our framework learns tilt angles (measurement coordinates) that are more accurate than the assumed ones.

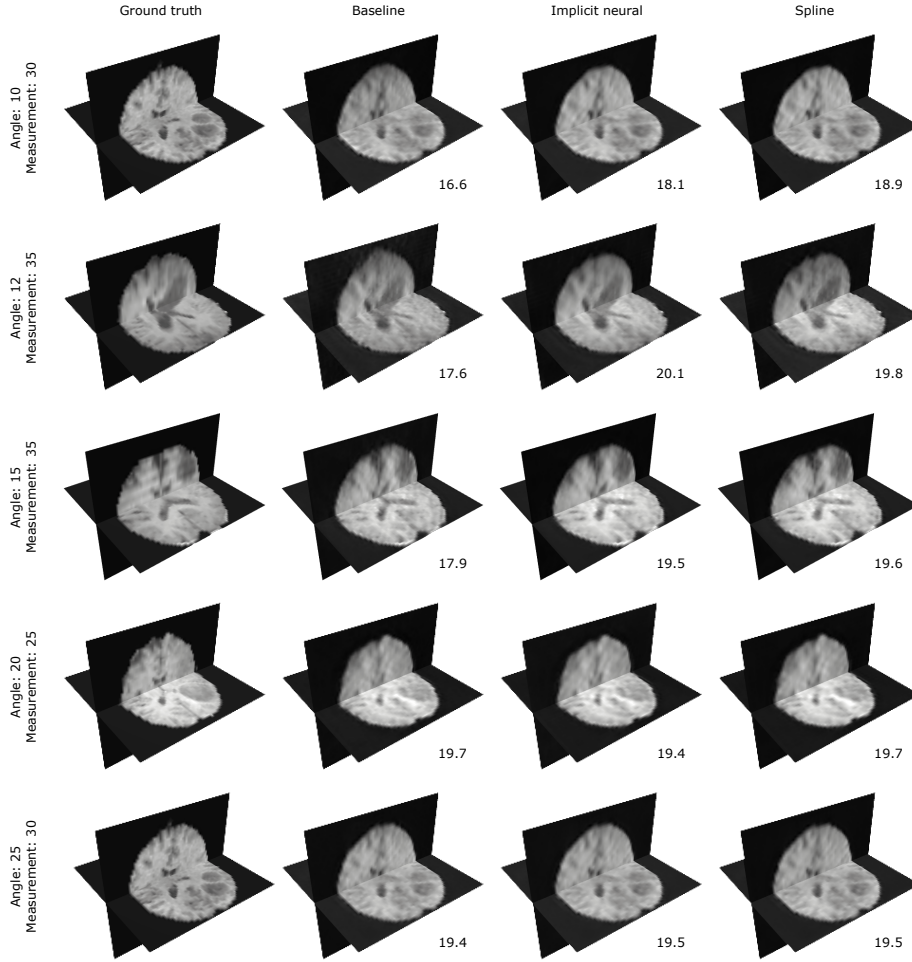


Figure 6.8: Example reconstructions for two of the orthogonal central slices of each of the test volumes in Figure 6.7. The SNRs for the entire volume are stated.

6.4 Summary

We presented a differentiable imaging inverse problem framework to jointly reconstruct the unknown image and learn unknown measurement coordinates when they are approximately known. There are two major elements in our proposed method. Firstly, we learn continuous representations of the measurements whose input are measurement coordinates and output are the corresponding measurements. By optimizing with respect to their parameters and their input, we jointly learn the measurement representation parameters and the unknown measurement coordinates. The second aspect of our method is that because these representations can be evaluated at any in-

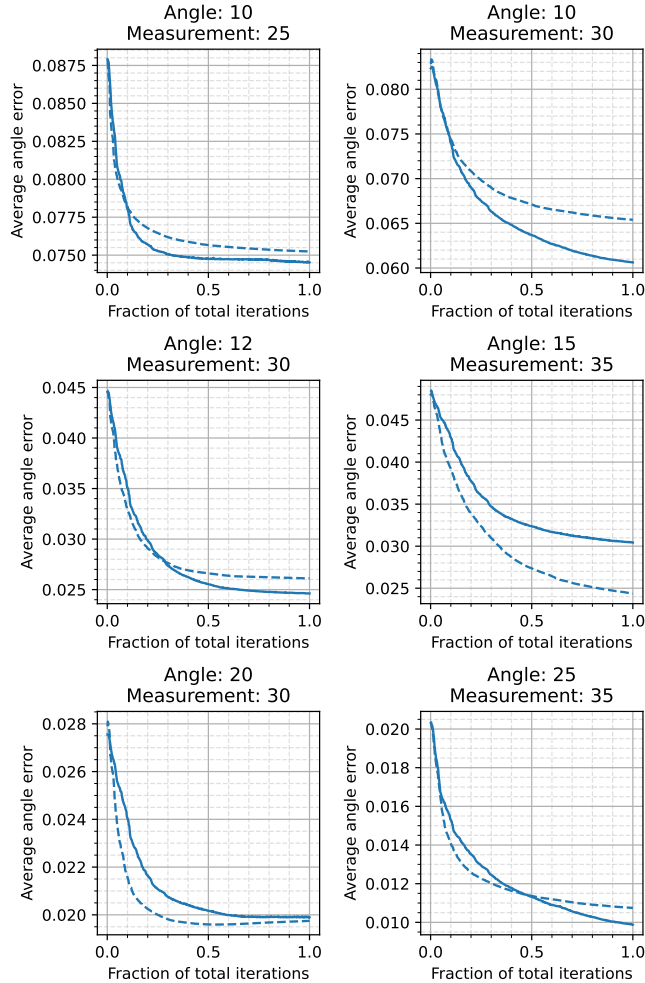


Figure 6.9: Average tilt angle error for one test volume with different measurement noise and measurement coordinate uncertainty combinations. The solid lines are for implicit neural representations and the dashed lines are for spline representations.

put coordinate, we can leverage reconstruction methods that are designed for measurement coordinates that are different from the ones of the observations. Our 2D and 3D CT imaging experiments show that our framework produces superior reconstructions when there is measurement coordinate uncertainty.

As our framework does not assume a particular measurement representation, we use both implicit neural networks and splines to represent measurements. Splines are generally viewed as interpolation tools; however, this chapter demonstrates that they can also be learnable differentiable representations that perform comparably to implicit neural representations. Differentiable splines may provide a viable solution for current research directions

that have been focusing on using implicit neural networks which can have significantly more parameters and complexity [148].

CHAPTER 7

LOOKING FORWARD

In this thesis, we reworked the essential imaging elements—measurements and operators. By doing so we were able to solve various imaging problems with greater accuracy and efficiency. For phase problems, we have developed algorithms to recover all the quantities in the phase problem, $|\mathbf{Y}|^2 = |\mathbf{A}\mathbf{X}|^2$. We also proposed general inverse problem methods to solve problems in practical settings such as scarce measurements with no training data and forward operator uncertainty.

The methods presented in this thesis open up a myriad of opportunities for future research. We briefly outline some of them here.

7.1 Phase problems

Measurement phase retrieval has proven to be a powerful development as shown in Chapters 2 and 3. It enables us to optically perform randomized algorithms and calibrate the TM for imaging through scattering media. However, because we require anchor signals, these benefits come at the expense of a reduction in data throughput for optical computing applications and calibration speed when imaging. Future work should quantify the smallest cost due to allocating a part of acquisition time for anchor measurements. We are optimistic that this will not be limiting because optical data rates are very high to begin with.

Presently, the TLS phase retrieval algorithm in Chapter 4 corrects the sensing vectors based on the one signal that we wish to recover. An interesting future line of work lies in multi-signal batch TLS for sensing vector denoising so that subsequent signals can be recovered without requiring sensing vector corrections if their measurements use the same set of sensing vectors.

Furthermore, there are other applications of phase retrieval with uncertain

sensing vectors. Ptychography, which can be modeled analogously to (4.1), is a prime example [9]. Ptychography has recently been addressed by least squares, comprising spectral initialization followed by gradient descent [165]. It would be interesting to see whether our TLS phase retrieval algorithm from Chapter 4 brings about accuracy improvements. Other ptychography methods use alternating updates to recover the object and the sensing vectors [166]. Our geometric intuition may help reduce the number of unknowns in sensing vector updates and thus improve the overall computational efficiency of these algorithms.

7.2 Imaging inverse problems

A strength of the framework in Chapter 6 is that no extra data is required to learn the measurement representations. However, a drawback is that it can be time consuming if there are multiple test images because we have to learn separate measurement representations and measurement coordinates for each new set of observations. Therefore, similarly to the future work on TLS phase retrieval, extending our framework to jointly recover a batch of images and the shared unknown measurement coordinates is an important step towards helping practitioners adopt our framework. Batch imaging also introduces robustness which can help learn more accurate measurement representations.

Another important endeavor is to adapt our framework in Chapter 6 to account for operator uncertainties due to reasons other than measurement coordinate uncertainty, which this chapter studied. For example, there may be approximation uncertainties if the true operator is approximated to enable faster computations and facilitate analysis. Not accounting for the approximation can lead to degraded solutions [18]. Another source of operator uncertainty arises when the object being imaged is altered in an unknown manner during the measurement acquisition process. For example, in CryoET imaging, the sample can translate and deform during imaging which needs to be taken into account [167, 168, 169].

REFERENCES

- [1] R. E. Blahut, *Theory of remote image formation*. Cambridge University Press, 2004.
- [2] G. H. Stout and L. H. Jensen, *X-ray structure determination: A practical guide*. Wiley, 1989.
- [3] K. Aki, A. Christoffersson, and E. S. Husebye, “Determination of the three-dimensional seismic structure of the lithosphere,” *Journal of Geophysical Research*, vol. 82, no. 2, pp. 277–296, 1977.
- [4] P. C. Hansen, J. G. Nagy, and D. P. O’Leary, *Deblurring Images*. Society for Industrial and Applied Mathematics, 2006.
- [5] S. Farsiu, D. Robinson, M. Elad, and P. Milanfar, “Advances and challenges in super-resolution,” *International Journal of Imaging Systems and Technology*, vol. 14, no. 2, pp. 47–57, 2004.
- [6] C. R. Vogel, *Computational Methods for Inverse Problems*. Society for Industrial and Applied Mathematics, 2002.
- [7] 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*, vol. 32, no. 3, pp. 87–109, 2015.
- [8] R. P. Millane, “Phase retrieval in crystallography and optics,” *JOSA A*, vol. 7, no. 3, pp. 394–411, 1990.
- [9] F. Pfeiffer, “X-ray ptychography,” *Nature Photonics*, vol. 12, no. 1, pp. 9–17, 2018.
- [10] J. R. Fienup, J. C. Marron, T. J. Schulz, and J. H. Seldin, “Hubble space telescope characterized by using phase-retrieval algorithms,” *Applied optics*, vol. 32, no. 10, pp. 1747–1767, 1993.

- [11] A. Fawzi, M. Balog, A. Huang, T. Hubert, B. Romera-Paredes, M. Barekatin, A. Novikov, F. J. R Ruiz, J. Schrittwieser, G. Swirszcz, D. Silver, D. Hassabis, and P. Kohli, “Discovering faster matrix multiplication algorithms with reinforcement learning,” *Nature*, vol. 610, no. 7930, pp. 47–53, 2022.
- [12] J. W. Goodman, *Introduction to Fourier optics*, 2nd ed., ser. McGraw-Hill series in electrical and computer engineering. New York: McGraw-Hill, 1996.
- [13] B. E. A. Saleh and M. C. Teich, *Fundamentals of photonics; 2nd ed.*, ser. Wiley series in pure and applied optics. New York, NY: Wiley, 2007.
- [14] S. Rotter and S. Gigan, “Light fields in complex media: Mesoscopic scattering meets wave control,” *Reviews of Modern Physics*, vol. 89, no. 1, p. 015005, 2017.
- [15] S. Popoff, G. Lerosey, M. Fink, A. C. Boccara, and S. Gigan, “Controlling light through optical disordered media: transmission matrix approach,” *New Journal of Physics*, vol. 13, no. 12, p. 123021, 2011.
- [16] J. Romberg, “Imaging via compressive sampling,” *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 14–20, 2008.
- [17] K. Kothari, S. Gupta, M. v. de Hoop, and I. Dokmanic, “Random mesh projectors for inverse problems,” in *International Conference on Learning Representations*, 2019.
- [18] S. Lunz, A. Hauptmann, T. Tarvainen, C.-B. Schonlieb, and S. Arridge, “On learned operator correction in inverse problems,” *SIAM Journal on Imaging Sciences*, vol. 14, no. 1, pp. 92–127, 2021.
- [19] A. E. Savakis and H. J. Trussell, “On the accuracy of psf representation in image restoration,” *IEEE transactions on image processing*, vol. 2, no. 2, pp. 252–259, 1993.
- [20] S. Gupta, R. Gribonval, L. Daudet, and I. Dokmanić, “Don’t take it lightly: Phasing optical random projections with unknown operators,” *Advances in Neural Information Processing Systems*, vol. 32, 2019.
- [21] A. Rahimi and B. Recht, “Random features for large-scale kernel machines,” in *Advances in Neural Information Processing Systems*, 2008, pp. 1177–1184.
- [22] Y. Yang, M. Pilanci, M. J. Wainwright et al., “Randomized sketches for kernels: Fast and optimal nonparametric regression,” *The Annals of Statistics*, vol. 45, no. 3, pp. 991–1023, 2017.

- [23] J. A. Tropp, A. Yurtsever, M. Udell, and V. Cevher, “Practical sketching algorithms for low-rank matrix approximation,” *SIAM Journal on Matrix Analysis and Applications*, vol. 38, no. 4, pp. 1454–1485, 2017.
- [24] A. Yurtsever, M. Udell, J. Tropp, and V. Cevher, “Sketchy decisions: Convex low-rank matrix optimization with optimal storage,” in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1188–1196.
- [25] F. X. X. Yu, A. T. Suresh, K. M. Choromanski, D. N. Holtmann-Rice, and S. Kumar, “Orthogonal random features,” in *Advances in Neural Information Processing Systems*, 2016, pp. 1975–1983.
- [26] A. Drémeau, A. Liutkus, D. Martina, O. Katz, C. Schülke, F. Krzakala, S. Gigan, and L. Daudet, “Reference-less measurement of the transmission matrix of a highly scattering material using a dmd and phase retrieval techniques,” *Optics express*, vol. 23, no. 9, pp. 11 898–11 911, 2015.
- [27] P. Netrapalli, P. Jain, and S. Sanghavi, “Phase retrieval using alternating minimization,” in *Advances in Neural Information Processing Systems*, 2013, pp. 2796–2804.
- [28] E. J. Candes, X. Li, and M. Soltanolkotabi, “Phase retrieval via wirtinger flow: Theory and algorithms,” *IEEE Transactions on Information Theory*, vol. 61, no. 4, pp. 1985–2007, 2015.
- [29] A. Liutkus, D. Martina, S. Popoff, G. Chardon, O. Katz, G. Lerosey, S. Gigan, L. Daudet, and I. Carron, “Imaging with nature: Compressive imaging using a multiply scattering medium,” *Scientific reports*, vol. 4, p. 5552, 2014.
- [30] A. Saade, F. Caltagirone, I. Carron, L. Daudet, A. Drémeau, S. Gigan, and F. Krzakala, “Random projections through multiple optical scattering: Approximating kernels at the speed of light,” in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 6215–6219.
- [31] W. S. Torgerson, “Multidimensional scaling: I. theory and method,” *Psychometrika*, vol. 17, no. 4, pp. 401–419, 1952.
- [32] I. Dokmanic, R. Parhizkar, J. Ranieri, and M. Vetterli, “Euclidean distance matrices: essential theory, algorithms, and applications,” *IEEE Signal Processing Magazine*, vol. 32, no. 6, pp. 12–30, 2015.
- [33] J. R. Fienup, “Phase retrieval algorithms: a comparison,” *Applied optics*, vol. 21, no. 15, pp. 2758–2769, 1982.

- [34] K. Jaganathan, Y. C. Eldar, and B. Hassibi, “Phase retrieval: An overview of recent developments,” *Optical Compressive Imaging*, pp. 279–312, 2016.
- [35] D. A. Barmherzig, J. Sun, P.-N. Li, T. J. Lane, and E. J. Candes, “Holographic phase retrieval and reference design,” *Inverse Problems*, vol. 35, no. 9, p. 094001, 2019.
- [36] R. Beinert, “One-dimensional phase retrieval with additional interference intensity measurements,” *Results in Mathematics*, vol. 72, no. 1-2, pp. 1–24, 2017.
- [37] W. Kim and M. H. Hayes, “Phase retrieval using two fourier-transform intensities,” *JOSA A*, vol. 7, no. 3, pp. 441–449, 1990.
- [38] G. Satat, M. Tancik, O. Gupta, B. Heshmat, and R. Raskar, “Object classification through scattering media with deep learning on time resolved measurement,” *Optics express*, vol. 25, no. 15, pp. 17 466–17 479, 2017.
- [39] M. Sharma, C. A. Metzler, S. Nagesh, O. Cossairt, R. G. Baraniuk, and A. Veeraraghavan, “Inverse scattering via transmission matrices: Broadband illumination and fast phase retrieval algorithms,” *IEEE Transactions on Computational Imaging*, 2019.
- [40] R. Horisaki, R. Takagi, and J. Tanida, “Learning-based imaging through scattering media,” *Optics express*, vol. 24, no. 13, pp. 13 738–13 743, 2016.
- [41] Q. Le, T. Sarlós, and A. Smola, “Fastfood-approximating kernel expansions in loglinear time,” in *Proceedings of the international conference on machine learning*, vol. 85, 2013.
- [42] P. H. Schoenemann, “A solution of the orthogonal procrustes problem with applications to orthogonal and oblique rotation,” Ph.D. dissertation, University of Illinois at Urbana-Champaign, 1964.
- [43] H. Zhang, Y. Liu, and H. Lei, “Localization from incomplete euclidean distance matrix: Performance analysis for the svd-mds approach,” *IEEE Transactions on Signal Processing*, 2019.
- [44] N. Halko, P.-G. Martinsson, and J. A. Tropp, “Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions,” *SIAM review*, vol. 53, no. 2, pp. 217–288, 2011.
- [45] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner et al., “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

- [46] S. Gupta, R. Gribonval, L. Daudet, and I. Dokmanić, “Fast optical system identification by numerical interferometry,” in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 1474–1478.
- [47] C. W. Hsu, S. F. Liew, A. Goetschy, H. Cao, and A. D. Stone, “Correlation-enhanced control of wave focusing in disordered media,” *Nature Physics*, vol. 13, no. 5, p. 497, 2017.
- [48] B. Rajaei, S. Gigan, F. Krzakala, and L. Daudet, “Robust Phase Retrieval with the Swept Approximate Message Passing (prSAMP) Algorithm,” *Image Processing On Line*, vol. 7, pp. 43–55, 2017.
- [49] S. Popoff, G. Lerosey, R. Carminati, M. Fink, A. Boccara, and S. Gigan, “Measuring the transmission matrix in optics: an approach to the study and control of light propagation in disordered media,” *Physical review letters*, vol. 104, no. 10, p. 100601, 2010.
- [50] J. Yoon, K. Lee, J. Park, and Y. Park, “Measuring optical transmission matrices by wavefront shaping,” *Optics Express*, vol. 23, no. 8, pp. 10 158–10 167, 2015.
- [51] S. Popoff, G. Lerosey, M. Fink, A. C. Boccara, and S. Gigan, “Image transmission through an opaque material,” *Nature communications*, vol. 1, no. 1, pp. 1–5, 2010.
- [52] Y. Choi, T. D. Yang, C. Fang-Yen, P. Kang, K. J. Lee, R. R. Dasari, M. S. Feld, and W. Choi, “Overcoming the diffraction limit using multiple light scattering in a highly disordered medium,” *Physical review letters*, vol. 107, no. 2, p. 023902, 2011.
- [53] B. Rajaei, E. W. Tramel, S. Gigan, F. Krzakala, and L. Daudet, “Intensity-only optical compressive imaging using a multiply scattering material and a double phase retrieval approach,” in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 4054–4058.
- [54] P. Stoica and J. Li, “Lecture notes-source localization from range-difference measurements,” *IEEE Signal Processing Magazine*, vol. 23, no. 6, pp. 63–66, 2006.
- [55] Y. Chen, Y. Chi, J. Fan, and C. Ma, “Gradient descent with random initialization: Fast global convergence for nonconvex phase retrieval,” *Mathematical Programming*, vol. 176, no. 1, pp. 5–37, 2019.
- [56] S. Gupta and I. Dokmanić, “Total least squares phase retrieval,” *IEEE Transactions on Signal Processing*, vol. 70, pp. 536–549, 2022.

- [57] T. Bendory, R. Beinert, and Y. C. Eldar, “Fourier phase retrieval: Uniqueness and algorithms,” in *Compressed Sensing and its Applications*. Springer, 2017, pp. 55–91.
- [58] G. H. Golub and C. F. Van Loan, “An analysis of the total least squares problem,” *SIAM journal on numerical analysis*, vol. 17, no. 6, pp. 883–893, 1980.
- [59] I. Markovsky and S. Van Huffel, “Overview of total least-squares methods,” *Signal processing*, vol. 87, no. 10, pp. 2283–2302, 2007.
- [60] P. T. Boggs, R. H. Byrd, and R. B. Schnabel, “A stable and efficient algorithm for nonlinear orthogonal distance regression,” *SIAM Journal on Scientific and Statistical Computing*, vol. 8, no. 6, pp. 1052–1078, 1987.
- [61] R. W. Gerchberg, “A practical algorithm for the determination of phase from image and diffraction plane pictures,” *Optik*, vol. 35, pp. 237–246, 1972.
- [62] J. R. Fienup, “Reconstruction of an object from the modulus of its fourier transform,” *Optics letters*, vol. 3, no. 1, pp. 27–29, 1978.
- [63] G. Wang, G. B. Giannakis, and Y. C. Eldar, “Solving systems of random quadratic equations via truncated amplitude flow,” *IEEE Transactions on Information Theory*, vol. 64, no. 2, pp. 773–794, 2017.
- [64] Y. Chen and E. Candes, “Solving random quadratic systems of equations is nearly as easy as solving linear systems,” in *Advances in Neural Information Processing Systems*, 2015, pp. 739–747.
- [65] S. Huang, S. Gupta, and I. Dokmanić, “Solving complex quadratic systems with full-rank random matrices,” *IEEE Transactions on Signal Processing*, vol. 68, pp. 4782–4796, 2020.
- [66] S. Huang and I. Dokmanić, “Reconstructing point sets from distance distributions,” *IEEE Transactions on Signal Processing*, vol. 69, pp. 1811–1827, 2021.
- [67] T. T. Cai, X. Li, Z. Ma et al., “Optimal rates of convergence for noisy sparse phase retrieval via thresholded wirtinger flow,” *The Annals of Statistics*, vol. 44, no. 5, pp. 2221–2251, 2016.
- [68] G. Wang, L. Zhang, G. B. Giannakis, M. Akçakaya, and J. Chen, “Sparse phase retrieval via truncated amplitude flow,” *IEEE Transactions on Signal Processing*, vol. 66, no. 2, pp. 479–491, 2017.

- [69] M. A. Davenport and J. Romberg, “An overview of low-rank matrix recovery from incomplete observations,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 4, pp. 608–622, 2016.
- [70] E. J. Candes, Y. C. Eldar, T. Strohmer, and V. Voroninski, “Phase retrieval via matrix completion,” *SIAM review*, vol. 57, no. 2, pp. 225–251, 2015.
- [71] E. J. Candes, T. Strohmer, and V. Voroninski, “Phaselift: Exact and stable signal recovery from magnitude measurements via convex programming,” *Communications on Pure and Applied Mathematics*, vol. 66, no. 8, pp. 1241–1274, 2013.
- [72] T. Goldstein and C. Studer, “Phasemax: Convex phase retrieval via basis pursuit,” *IEEE Transactions on Information Theory*, vol. 64, no. 4, pp. 2675–2689, 2018.
- [73] S. Van Huffel and J. Vandewalle, “On the accuracy of total least squares and least squares techniques in the presence of errors on all data,” *Automatica*, vol. 25, no. 5, pp. 765–769, 1989.
- [74] I. Markovsky, J. C. Willems, S. Van Huffel, B. De Moor, and R. Pintelon, “Application of structured total least squares for system identification and model reduction,” *IEEE Transactions on Automatic Control*, vol. 50, no. 10, pp. 1490–1500, 2005.
- [75] D. Malioutov and N. Slavov, “Convex total least squares,” in *International Conference on Machine Learning*, 2014, pp. 109–117.
- [76] H. Zhu, G. Leus, and G. B. Giannakis, “Sparsity-cognizant total least-squares for perturbed compressive sampling,” *IEEE Transactions on Signal Processing*, vol. 59, no. 5, pp. 2002–2016, 2011.
- [77] D. M. Sima, S. Van Huffel, and G. H. Golub, “Regularized total least squares based on quadratic eigenvalue problem solvers,” *BIT Numerical Mathematics*, vol. 44, no. 4, pp. 793–812, 2004.
- [78] A. E. Yagle and A. E. Bell, “One-and two-dimensional minimum and nonminimum phase retrieval by solving linear systems of equations,” *IEEE Transactions on Signal Processing*, vol. 47, no. 11, pp. 2978–2989, 1999.
- [79] H. Schwetlick and V. Tiller, “Numerical methods for estimating parameters in nonlinear models with errors in the variables,” *Technometrics*, vol. 27, no. 1, pp. 17–24, 1985.
- [80] D. Powell and J. Macdonald, “A rapidly convergent iterative method for the solution of the generalised nonlinear least squares problem,” *The Computer Journal*, vol. 15, no. 2, pp. 148–155, 1972.

- [81] A. Wiesel, Y. C. Eldar, and A. Beck, “Maximum likelihood estimation in linear models with a gaussian model matrix,” *IEEE Signal Processing Letters*, vol. 13, no. 5, pp. 292–295, 2006.
- [82] S. Gould, B. Fernando, A. Cherian, P. Anderson, R. S. Cruz, and E. Guo, “On differentiating parameterized argmin and argmax problems with application to bi-level optimization,” *arXiv preprint arXiv:1607.05447*, 2016.
- [83] R. Balan, P. Casazza, and D. Edidin, “On signal reconstruction without phase,” *Applied and Computational Harmonic Analysis*, vol. 20, no. 3, pp. 345–356, 2006.
- [84] A. Conca, D. Edidin, M. Hering, and C. Vinzant, “An algebraic characterization of injectivity in phase retrieval,” *Applied and Computational Harmonic Analysis*, vol. 38, no. 2, pp. 346–356, 2015.
- [85] P. Sprechmann, R. Litman, T. B. Yakar, A. M. Bronstein, and G. Sapiro, “Supervised sparse analysis and synthesis operators,” in *Advances in Neural Information Processing Systems*, 2013, pp. 908–916.
- [86] E. Galetti, A. Curtis, B. Baptie, D. Jenkins, and H. Nicolson, “Transdimensional Love-wave tomography of the British Isles and shear-velocity structure of the East Irish Sea Basin from ambient-noise interferometry,” *Geophys. J. Int.*, vol. 208, no. 1, pp. 36–58, Jan. 2017.
- [87] A. Bora, E. Price, and A. G. Dimakis, “Ambientgan: Generative models from lossy measurements,” in *International Conference on Learning Representations (ICLR)*, 2018.
- [88] S. Lutz, O. Öktem, and C.-B. Schönlieb, “Adversarial regularizers in inverse problems,” *Advances in neural information processing systems*, vol. 31, 2018.
- [89] K. H. Jin, M. T. McCann, E. Froustey, and M. Unser, “Deep convolutional neural network for inverse problems in imaging,” *IEEE Transactions on Image Processing*, vol. 26, no. 9, pp. 4509–4522, 2017.
- [90] R. P. Bording, A. Gersztenkorn, L. R. Lines, J. A. Scales, and S. Treitel, “Applications of seismic travel-time tomography,” *Geophysical Journal International*, vol. 90, no. 2, pp. 285–303, 1987.
- [91] J. Hole, “Nonlinear high-resolution three-dimensional seismic travel time tomography,” *Journal of Geophysical Research: Solid Earth*, vol. 97, no. B5, pp. 6553–6562, 1992.

- [92] T. Ogawa, Y. Kosugi, and H. Kanada, “Neural network based solution to inverse problems,” in *Neural Networks Proceedings, 1998. IEEE World Congress on Computational Intelligence. The 1998 IEEE International Joint Conference on*, vol. 3. IEEE, 1998, pp. 2471–2476.
- [93] S. R. H. Hoole, “Artificial neural networks in the solution of inverse electromagnetic field problems,” *IEEE Trans. Magn.*, vol. 29, no. 2, pp. 1931–1934, Mar. 1993.
- [94] H. Schiller and R. Doerffer, “Neural network for emulation of an inverse model operational derivation of Case II water properties from MERIS data,” *International Journal of Remote Sensing*, Nov. 2010.
- [95] İ. Güler and E. D. Übeyli, “ECG beat classifier designed by combined neural network model,” *Pattern Recognition*, vol. 38, no. 2, pp. 199–208, 2005.
- [96] D. L. Hudson and M. E. Cohen, *Neural networks and artificial intelligence for biomedical engineering*. Wiley Online Library, 2000.
- [97] H. Greenspan, B. van Ginneken, and R. M. Summers, “Deep Learning in Medical Imaging: Overview and Future Promise of an Exciting New Technique,” *IEEE Trans. Med. Imag.*, vol. 35, no. 5, pp. 1153–1159, may 2016.
- [98] F. Porikli, S. Shan, C. Snoek, R. Sukthankar, and X. Wang, “Deep Learning for Visual Understanding [From the Guest Editors],” *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 24–25, Nov 2017.
- [99] F. Porikli, S. Shan, C. Snoek, R. Sukthankar, and X. Wang, “Deep Learning for Visual Understanding: Part 2 [From the Guest Editors],” *IEEE Signal Process. Mag.*, vol. 35, no. 1, pp. 17–19, Jan 2018.
- [100] A. Lucas, M. Iliadis, R. Molina, and A. K. Katsaggelos, “Using Deep Neural Networks for Inverse Problems in Imaging: Beyond Analytical Methods,” *IEEE Signal Process. Mag.*, vol. 35, no. 1, pp. 20–36, 2018.
- [101] M. T. McCann, K. H. Jin, and M. Unser, “Convolutional neural networks for inverse problems in imaging: A review,” *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 85–95, 2017.
- [102] M. Araya-Polo, J. Jennings, A. Adler, and T. Dahlke, “Deep-learning tomography,” *The Leading Edge*, Dec. 2017.
- [103] W. Lewis and D. Vigh, “Deep learning prior models from seismic images for full-waveform inversion,” in *SEG International Exposition and Annual Meeting*. Society of Exploration Geophysicists, 2017.

- [104] M. J. Bianco and P. Gerstoft, “Travel time tomography with adaptive dictionaries,” *IEEE Transactions on Computational Imaging*, vol. 4, no. 4, pp. 499–511, 2018.
- [105] D. M. Pelt and K. J. Batenburg, “Fast tomographic reconstruction from limited data using artificial neural networks,” *IEEE Trans. on Image Process.*, vol. 22, no. 12, pp. 5238–5251, 2013.
- [106] B. Zhu, J. Z. Liu, S. F. Cauley, B. R. Rosen, and M. S. Rosen, “Image reconstruction by domain-transform manifold learning,” *Nature*, vol. 555, no. 7697, p. 487, Mar. 2018.
- [107] G. Wang, “A perspective on deep imaging,” *IEEE Access*, vol. 4, pp. 8914–8924, 2016.
- [108] S. Antholzer, M. Haltmeier, and J. Schwab, “Deep learning for photoacoustic tomography from sparse data,” *Inverse problems in science and engineering*, vol. 27, no. 7, pp. 987–1005, 2019.
- [109] Y. S. Han, J. Yoo, and J. C. Ye, “Deep Residual Learning for Compressed Sensing CT Reconstruction via Persistent Homology Analysis,” *arXiv preprint arXiv:1611.06391*, Nov. 2016.
- [110] H. Zhang, L. Li, K. Qiao, L. Wang, B. Yan, L. Li, and G. Hu, “Image Prediction for Limited-angle Tomography via Deep Learning with Convolutional Neural Network,” *arXiv preprint arXiv:1607.08707v1*, July 2016.
- [111] H. Li, J. Schwab, S. Antholzer, and M. Haltmeier, “Nett: Solving inverse problems with deep neural networks,” *Inverse Problems*, vol. 36, no. 6, p. 065005, 2020.
- [112] B. Kelly, T. P. Matthews, and M. A. Anastasio, “Deep Learning-Guided Image Reconstruction from Incomplete Data,” *arXiv preprint arXiv:1709.00584*, Sep. 2017.
- [113] J. Adler and O. Öktem, “Learned primal-dual reconstruction,” *IEEE transactions on medical imaging*, vol. 37, no. 6, pp. 1322–1332, 2018.
- [114] J. Adler and O. Öktem, “Solving ill-posed inverse problems using iterative deep neural networks,” *Inverse Problems*, vol. 33, no. 12, p. 124007, 2017.
- [115] J.-H. R. Chang, C.-L. Li, B. Póczos, 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 Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5888–5897.

- [116] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg, “Plug-and-play priors for model based reconstruction,” in *Global Conference on Signal and Information Processing (GlobalSIP), 2013 IEEE*. IEEE, 2013, pp. 945–948.
- [117] K. Gregor and Y. LeCun, “Learning fast approximations of sparse coding,” in *Proceedings of the 27th International Conference on International Conference on Machine Learning*. Omnipress, 2010, pp. 399–406.
- [118] B. Xin, Y. Wang, W. Gao, D. Wipf, and B. Wang, “Maximal sparsity with deep networks?” in *Advances in Neural Information Processing Systems*, 2016, pp. 4340–4348.
- [119] A. Bora, A. Jalal, E. Price, and A. G. Dimakis, “Compressed sensing using generative models,” in *International Conference on Machine Learning*. PMLR, 2017, pp. 537–546.
- [120] R. Gribonval, G. Blanchard, N. Keriven, and Y. Traonmilin, “Compressive statistical learning with random feature moments,” *Mathematical Statistics and Learning*, vol. 3, no. 2, pp. 113–164, 2021.
- [121] M. Pilanci and M. J. Wainwright, “Iterative hessian sketch: Fast and accurate solution approximation for constrained least-squares,” *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1842–1879, 2016.
- [122] A. Rahimi and B. Recht, “Random features for large-scale kernel machines,” *Advances in Neural Information and Processing (NIPS)*, 2008.
- [123] A. Rahimi and B. Recht, “Weighted Sums of Random Kitchen Sinks: Replacing minimization with randomization in learning,” *Advances in Neural Information and Processing (NIPS)*, pp. 1313–1320, 2009.
- [124] M. A. Ganaie, M. Hu et al., “Ensemble deep learning: A review,” *arXiv preprint arXiv:2104.02395*, 2021.
- [125] R. H. Bartels, J. C. Beatty, and B. A. Barsky, *An introduction to splines for use in computer graphics and geometric modeling*. Morgan Kaufmann, 1995.
- [126] D. F. Rogers and J. A. Adams, *Mathematical elements for computer graphics*. McGraw-Hill, Inc., 1989.
- [127] M. Vetterli, J. Kovačević, and V. K. Goyal, *Foundations of signal processing*. Cambridge University Press, 2014.

- [128] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2015, pp. 234–241.
- [129] C. K. Sønderby, J. Caballero, L. Theis, W. Shi, and F. Huszár, “Amortised MAP inference for image super-resolution,” in *International Conference on Learning Representations*, 2017.
- [130] J. C. Ye, Y. Han, and E. Cha, “Deep convolutional framelets: A general deep learning framework for inverse problems,” *SIAM Journal on Imaging Sciences*, vol. 11, no. 2, pp. 991–1048, 2018.
- [131] D. Mery, V. Rizzo, U. Zscherpel, G. Mondragón, I. Lillo, I. Zuccar, H. Lobel, and M. Carrasco, “GDXray: The Database of X-ray Images for Nondestructive Testing,” *Journal of Nondestructive Evaluation*, vol. 34, 11 2015.
- [132] F. Yu, Y. Zhang, S. Song, A. Seff, and J. Xiao, “LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop,” *arXiv preprint arXiv:1506.03365*, 2015.
- [133] Z. Liu, P. Luo, X. Wang, and X. Tang, “Deep learning face attributes in the wild,” in *Proceedings of International Conference on Computer Vision (ICCV)*, Dec. 2015.
- [134] S. Gupta, K. Kothari, V. Debarnot, and I. Dokmanić, “Differentiable uncalibrated imaging,” *arXiv preprint arXiv:2211.10525*, 2022.
- [135] 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*, vol. 1, no. 1, pp. 39–56, 2020.
- [136] D. Ulyanov, A. Vedaldi, and V. Lempitsky, “Deep image prior,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9446–9454.
- [137] H. Gupta, K. H. Jin, H. Q. Nguyen, M. T. McCann, and M. Unser, “Cnn-based projected gradient descent for consistent ct image reconstruction,” *IEEE transactions on medical imaging*, vol. 37, no. 6, pp. 1440–1453, 2018.
- [138] J. Rick Chang, C.-L. Li, B. Póczos, 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*, 2017, pp. 5888–5897.

- [139] D. Gilton, G. Ongie, and R. Willett, “Neumann networks for linear inverse problems in imaging,” *IEEE Transactions on Computational Imaging*, vol. 6, pp. 328–343, 2019.
- [140] K. Zhang, Y. Li, W. Zuo, L. Zhang, L. Van Gool, and R. Timofte, “Plug-and-play image restoration with deep denoiser prior,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.
- [141] K. Kothari, A. Khorashadizadeh, M. de Hoop, and I. Dokmanić, “Trumpets: Injective flows for inference and inverse problems,” in *Uncertainty in Artificial Intelligence*. PMLR, 2021, pp. 1269–1278.
- [142] Y. Song, L. Shen, L. Xing, and S. Ermon, “Solving inverse problems in medical imaging with score-based generative models,” *arXiv preprint arXiv:2111.08005*, 2021.
- [143] D. Gilton, G. Ongie, and R. Willett, “Model adaptation for inverse problems in imaging,” *IEEE Transactions on Computational Imaging*, vol. 7, pp. 661–674, 2021.
- [144] A. Gossard and P. Weiss, “Training adaptive reconstruction networks for inverse problems,” *arXiv preprint arXiv:2202.11342*, 2022.
- [145] L. Piegl, “On nurbs: a survey,” *IEEE Computer Graphics and Applications*, vol. 11, no. 01, pp. 55–71, 1991.
- [146] E. Cohen, T. Lyche, and R. Riesenfeld, “Discrete b-splines and subdivision techniques in computer-aided geometric design and computer graphics,” *Computer graphics and image processing*, vol. 14, no. 2, pp. 87–111, 1980.
- [147] A. D. Prasad, A. Balu, H. Shah, S. Sarkar, C. Hegde, and A. Krishnamurthy, “Nurbs-diff: A differentiable programming module for nurbs,” *Computer-Aided Design*, vol. 146, p. 103199, 2022.
- [148] Y. Xie, T. Takikawa, S. Saito, O. Litany, S. Yan, N. Khan, F. Tombari, J. Tompkin, V. Sitzmann, and S. Sridhar, “Neural fields in visual computing and beyond,” in *Computer Graphics Forum*, vol. 41, no. 2. Wiley Online Library, 2022, pp. 641–676.
- [149] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, “DeepSDF: Learning continuous signed distance functions for shape representation,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 165–174.
- [150] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, “Nerf: Representing scenes as neural radiance fields for view synthesis,” in *European conference on computer vision*. Springer, 2020, pp. 405–421.

- [151] V. Sitzmann, J. Martel, A. Bergman, D. Lindell, and G. Wetzstein, “Implicit neural representations with periodic activation functions,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 7462–7473, 2020.
- [152] A. W. Reed, H. Kim, R. Anirudh, K. A. Mohan, K. Champley, J. Kang, and S. Jayasuriya, “Dynamic ct reconstruction from limited views with implicit neural representations and parametric motion fields,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 2258–2268.
- [153] L. Lozanski, M. Anastasio, and U. Villa, “Neural fields for dynamic imaging,” in *Medical Imaging 2022: Physics of Medical Imaging*, vol. 12031. SPIE, 2022, pp. 231–238.
- [154] Y. Sun, J. Liu, M. Xie, B. Wohlberg, and U. S. Kamilov, “Coil: Coordinate-based internal learning for tomographic imaging,” *IEEE Transactions on Computational Imaging*, vol. 7, pp. 1400–1412, 2021.
- [155] S. Basu and Y. Bresler, “Uniqueness of tomography with unknown view angles,” *IEEE Transactions on Image Processing*, vol. 9, no. 6, pp. 1094–1106, 2000.
- [156] R. R. Coifman, Y. Shkolnisky, F. J. Sigworth, and A. Singer, “Graph laplacian tomography from unknown random projections,” *IEEE Transactions on Image Processing*, vol. 17, no. 10, pp. 1891–1899, 2008.
- [157] T. Bendory, A. Bartesaghi, and A. Singer, “Single-particle cryo-electron microscopy: Mathematical theory, computational challenges, and opportunities,” *IEEE signal processing magazine*, vol. 37, no. 2, pp. 58–76, 2020.
- [158] L. Piegl and W. Tiller, *The NURBS book*. Springer Science & Business Media, 1996.
- [159] D. F. Rogers, *An introduction to NURBS: with historical perspective*. Morgan Kaufmann, 2001.
- [160] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, “3d u-net: learning dense volumetric segmentation from sparse annotation,” in *International conference on medical image computing and computer-assisted intervention*. Springer, 2016, pp. 424–432.
- [161] J. Leuschner, M. Schmidt, D. O. Baguer, and P. Maaß, “The lodopab-ct dataset: A benchmark dataset for low-dose ct reconstruction methods,” *arXiv preprint arXiv:1910.01113*, 2019.

- [162] B. H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, J. Kirby, Y. Burren, N. Porz, J. Slotboom, R. Wiest et al., “The multimodal brain tumor image segmentation benchmark (brats),” *IEEE transactions on medical imaging*, vol. 34, no. 10, pp. 1993–2024, 2014.
- [163] S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, J. S. Kirby, J. B. Freymann, K. Farahani, and C. Davatzikos, “Advancing the cancer genome atlas glioma mri collections with expert segmentation labels and radiomic features,” *Scientific data*, vol. 4, no. 1, pp. 1–13, 2017.
- [164] S. Bakas, M. Reyes, A. Jakab, S. Bauer, M. Rempfler, A. Crimi, R. T. Shinohara, C. Berger, S. M. Ha, M. Rozycki et al., “Identifying the best machine learning algorithms for brain tumor segmentation, progression assessment, and overall survival prediction in the brats challenge,” *arXiv preprint arXiv:1811.02629*, 2018.
- [165] L. Valzania, J. Dong, and S. Gigan, “Accelerating ptychographic reconstructions using spectral initializations,” *Optics Letters*, vol. 46, no. 6, pp. 1357–1360, 2021.
- [166] A. Maiden, D. Johnson, and P. Li, “Further improvements to the ptychographical iterative engine,” *Optica*, vol. 4, no. 7, pp. 736–745, 2017.
- [167] D. Tegunov and P. Cramer, “Real-time cryo-electron microscopy data preprocessing with warp,” *Nature methods*, vol. 16, no. 11, pp. 1146–1152, 2019.
- [168] K. Naydenova, P. Jia, and C. J. Russo, “Cryo-em with sub-1 Å specimen movement,” *Science*, vol. 370, no. 6513, pp. 223–226, 2020.
- [169] J.-J. Fernandez and S. Li, “Tomoalign: A novel approach to correcting sample motion and 3d ctf in cryoet,” *Journal of Structural Biology*, vol. 213, no. 4, p. 107778, 2021.
- [170] A. Beck, P. Stoica, and J. Li, “Exact and approximate solutions of source localization problems,” *IEEE Transactions on Signal Processing*, vol. 56, no. 5, pp. 1770–1778, 2008.
- [171] E. J. Candes, X. Li, and M. Soltanolkotabi, “Phase retrieval from coded diffraction patterns,” *Applied and Computational Harmonic Analysis*, vol. 39, no. 2, pp. 277–299, 2015.
- [172] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the Conference on Computer Vision and Pattern Recognition*. IEEE, 2016, pp. 770–778.
- [173] K. He, X. Zhang, S. Ren, and J. Sun, “Identity mappings in deep residual networks,” in *European Conference on Computer Vision*. Springer, 2016, pp. 630–645.

- [174] M. Buda, A. Saha, and M. A. Mazurowski, “Association of genomic subtypes of lower-grade gliomas with shape features automatically extracted by a deep learning algorithm,” *Computers in Biology and Medicine*, vol. 109, 2019.
- [175] J. L. Prince and A. S. Willsky, “Constrained sinogram restoration for limited-angle tomography,” *Optical Engineering*, vol. 29, no. 5, pp. 535–544, 1990.

APPENDIX A

MEASUREMENT PHASE RETRIEVAL

A.1 Practical considerations with hardware

Reference design. On the OPU system described in Figure 2.1, the DMD only lets us encode and randomly project binary signals. Therefore, all pairwise differences $(\mathbf{x}_q - \mathbf{x}_r)$ between the columns of $\mathbf{X} \in \mathbb{R}^{N \times Q}$ must be binary. To design reference signals we first collect all frames $\{\boldsymbol{\xi}_s\}_{s=1}^S$ and sum them $\sum_{s=1}^S \boldsymbol{\xi}_s$. The first reference \mathbf{r}_1 is initialized to ones at the indices where $\sum_{s=1}^S \boldsymbol{\xi}_s$ is nonzero. Next, some of \mathbf{r}_1 's zero-valued entries are flipped to one with probability α . A similar process is used for all subsequent references. In general, a reference \mathbf{r}_q is initialized by assigning ones to the nonzero support of $(\sum_{s=1}^S \boldsymbol{\xi}_s + \sum_{k=1}^{q-1} \mathbf{r}_k)$ and then flipping some of its zero entries with probability α .

There is a tradeoff between large and small α . If α is too large, a reference may become all-ones before all subsequent references are generated. On the other hand if α is too small, $\mathbf{r}_{q+1} - \mathbf{r}_q$ will have many zeros and so $|\mathbf{A}(\mathbf{r}_{q+1} - \mathbf{r}_q)|^2$ may not be high enough to be detected by the camera sensor. The consequence of this tradeoff is that in practice the number of anchors is limited. Furthermore, in general a larger N makes it easier to make K good anchors as α can be larger which keeps $|\mathbf{A}(\mathbf{r}_{q+1} - \mathbf{r}_q)|^2$ away from the sensitivity threshold.

Figure A.1 shows binary references reshaped into squares which were used for the linearity experiment on the OPU in Figure 2.2c. Here, $N = 64^2$ and $\alpha = 0.2$. The number on top of each reference is the difference in the number of ones between itself and the previously generated reference.

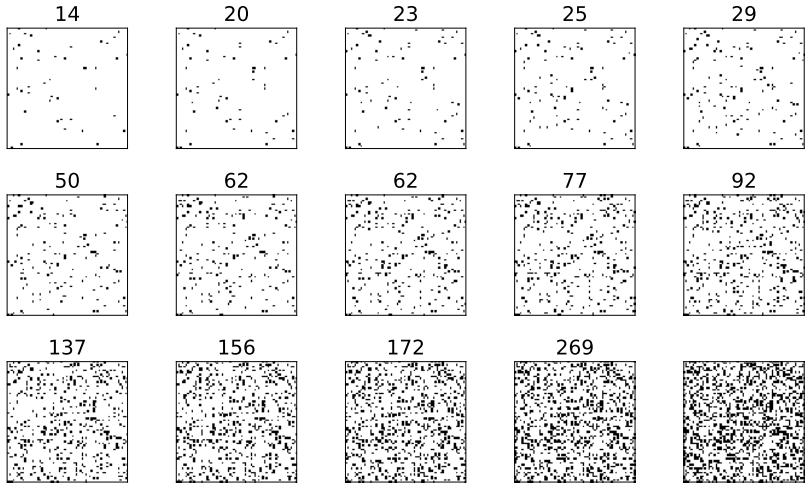


Figure A.1: Binary references reshaped into squares which were used for the linearity experiment on the OPU in Figure 2.2c. Here $N = 64^2$ and $\alpha = 0.2$. The number on top of each anchor is the difference in the number of ones between itself and the previously generated reference.

Minimum attainable measurement. To determine the sensitivity threshold, τ , we randomly project an all-zero signal a few times and record the output. The minimum, mode or mean of these measurements can be taken to be the minimum that can be measured. Once τ is estimated, we apply a mask and zero any measurements which are equal to or less than the minimum.

Camera sensor saturation. It is possible for the signal reaching the camera to saturate the sensor. In a b -bit system we can detect this if many measurements are equal to $2^b - 1$. In all experiments, we ensure that there is no saturation by adjusting the camera exposure. This again involves a tradeoff: if exposure is too high, we saturate the sensor; if it is too low, measurements may be too small to be detected and we are not exploiting the full dynamic range.

A.2 Randomized singular value decomposition (RSVD) details

Algorithm 3 is the prototype randomized SVD algorithm given by [44]. To implement this on hardware we replace Step 1 and 2 to formulate Algorithm 4. As \mathbf{A} in Algorithm 4 has iid entries following a standard complex Gaussian, calculating \mathbf{P} in Algorithm 4 is the same as doing step 2 in Algorithm 3. We only need to do half the number of projections because we use an iid complex random matrix. The real and imaginary parts are two random projections.

Algorithm 3 Prototype randomized SVD algorithm [44].

Input: Matrix, $\mathbf{B} \in \mathbb{R}^{M \times N}$ whose SVD is required, a target number of K singular vectors

Output: The SVD \mathbf{U} , $\mathbf{\Sigma}$ and \mathbf{V}^*

- 1: Generate an $N \times 2K$ random Gaussian matrix \mathbf{A} .
 - 2: Form $\mathbf{Y} = \mathbf{B}\mathbf{A}$.
 - 3: Construct a matrix \mathbf{Q} whose columns form an orthonormal basis for the range of \mathbf{Y} .
 - 4: Form $\mathbf{C} = \mathbf{Q}^*\mathbf{B}$.
 - 5: Compute the SVD of the smaller $\mathbf{C} = \tilde{\mathbf{U}}\mathbf{\Sigma}\mathbf{V}^*$.
 - 6: $\mathbf{U} = \mathbf{Q}\tilde{\mathbf{U}}$.
-

Algorithm 4 Randomized SVD algorithm on the OPU.

Input: Matrix, $\mathbf{B} \in \mathbb{R}^{M \times N}$ whose SVD is required, a target number of K singular vectors

Output: The SVD \mathbf{U} , $\mathbf{\Sigma}$ and \mathbf{V}^*

- 1: Solve $|\mathbf{Y}|^2 = |\mathbf{A}\mathbf{B}^*|^2$ by treating each column of \mathbf{B}^* as a frame and using Algorithm 1, where $\mathbf{A} \in \mathbb{C}^{K \times N}$ is as in the MPR problem and has K rows.
 - 2: Horizontally stack the real and imaginary parts of $\mathbf{Y}^* \in \mathbb{C}^{M \times K}$ as $\mathbf{P} = [\text{Re}(\mathbf{Y}^*) \quad \text{Im}(\mathbf{Y}^*)] \in \mathbb{R}^{M \times 2K}$.
 - 3: Construct a matrix \mathbf{Q} whose columns form an orthonormal basis for the range of \mathbf{P} .
 - 4: Form $\mathbf{C} = \mathbf{Q}^*\mathbf{B}$.
 - 5: Compute the SVD of the smaller $\mathbf{C} = \tilde{\mathbf{U}}\mathbf{\Sigma}\mathbf{V}^*$.
 - 6: $\mathbf{U} = \mathbf{Q}\tilde{\mathbf{U}}$.
-

A.3 Localization with known anchor positions

If we have perfect knowledge of the anchor locations in the complex plane, we do not need to localize them for each frame s . The localization problem then boils down to multilateration, which can be formulated by minimizing the square-range-based least squares (SR-LS) objective [170],

$$\hat{\mathbf{v}}_1 = \arg \min_{\mathbf{v}_1} \sum_{q=2}^Q (\|\mathbf{v}_1 - \mathbf{v}_q\|_2^2 - d_q^2)^2 \quad (\text{A.1})$$

where \mathbf{v}_1 and \mathbf{v}_q are as defined in Section 2.2.3 and d_q is the noisy measured distance. There exists efficient algorithms which solve (A.1) to global optimality [170], as well as suboptimal solutions based on solving a small linear system [54].

APPENDIX B

TOTAL LEAST SQUARES PHASE RETRIEVAL

B.1 Roots of cubic equations

Consider finding the roots of the following cubic equation

$$ax^3 + bx^2 + cx + d = 0. \quad (\text{B.1})$$

Denote

$$\psi_0 = b^2 - 3ac \quad (\text{B.2})$$

$$\psi_1 = 2b^3 - 9abc + 27a^2d \quad (\text{B.3})$$

$$\psi_3 = \sqrt[3]{\frac{\psi_1 + \sqrt{\psi_1^2 - 4\psi_0^3}}{2}}. \quad (\text{B.4})$$

Then for $k \in \{0, 1, 2\}$ the three roots, x_k , are

$$x_k = -\frac{1}{3a} \left(b + \theta^k \psi_3 + \frac{\psi_0}{\theta^k \psi_3} \right) \quad (\text{B.5})$$

where θ is the cube root of unity, $\theta = \frac{-1 + \sqrt{-3}}{2}$.

Note that the cubic equations in Chapter 4 are with $b = 0$ which simplifies the above expressions.

B.2 Proof of Proposition 3

The ML estimator estimates both $\tilde{\mathbf{x}}$ and $\{\tilde{\mathbf{a}}_m\}_{m=1}^M$ from the data $\{y_m\}_{m=1}^M$ and $\{\mathbf{a}_m\}_{m=1}^M$ by minimizing the negative conditional log-likelihood

$$\arg \min_{\mathbf{x}, \mathbf{e}_1, \dots, \mathbf{e}_M} - \ln \left(\prod_{m=1}^M \Pr\{y_m, \mathbf{a}_m | \tilde{\mathbf{x}} = \mathbf{x}, \tilde{\mathbf{a}}_m = \mathbf{a}_m + \mathbf{e}_m\} \right)$$

With $\tilde{\mathbf{x}}$ and $\{\tilde{\mathbf{a}}_m\}_{m=1}^M$ given, the only randomness in each y_m and \mathbf{a}_m are due to η_m and $\boldsymbol{\delta}_m$. Furthermore as $\{\eta_m\}_{m=1}^M$ and $\{\boldsymbol{\delta}_m\}_{m=1}^M$ are independent, the negative conditional log-likelihood is,

$$\sum_{m=1}^M - \ln(\Pr\{\eta_m = |\langle \mathbf{a}_m + \mathbf{e}_m, \mathbf{x} \rangle|^2 - y_m\}) - \ln(\Pr\{\boldsymbol{\delta}_m = \mathbf{e}_m\}). \quad (\text{B.6})$$

Using the assumptions on the error distributions,

$$\ln(\Pr\{\eta_m = |\langle \mathbf{a}_m + \mathbf{e}_m, \mathbf{x} \rangle|^2 - y_m\}) = K_\eta - \frac{1}{2\sigma_\eta^2} (y_m - |\langle \mathbf{a}_m + \mathbf{e}_m, \mathbf{x} \rangle|^2)^2 \quad (\text{B.7})$$

and

$$\ln(\Pr\{\boldsymbol{\delta}_m = \mathbf{e}_m\}) = K_\delta - \frac{1}{2\sigma_\delta^2} \|\mathbf{e}_m\|_2^2 \quad (\text{B.8})$$

where K_η and K_δ are constants independent of \mathbf{x} and $\{\mathbf{e}_m\}_{m=1}^M$. Substituting these into (B.6) gives

$$\arg \min_{\mathbf{x}, \mathbf{e}_1, \dots, \mathbf{e}_M} \sum_{m=1}^M \frac{1}{\sigma_\delta^2} \|\mathbf{e}_m\|_2^2 + \frac{1}{\sigma_\eta^2} (y_m - |\langle \mathbf{a}_m + \mathbf{e}_m, \mathbf{x} \rangle|^2)^2.$$

B.3 Taylor series expansions

Lemma 1 states the Taylor series expansions around the no error point $\tilde{\mathbf{t}}$ for the TLS and LS solutions. The notation defined in Section 4.3 is used.

Lemma 1. *The Taylor series expansions for the solution $\mathbf{x}_{\text{TLS}}^\dagger$ to the TLS optimization problem (4.32), and, the solution $\mathbf{x}_{\text{LS}}^\dagger$ to the LS optimization problem (4.38) at the no error point $\mathbf{t} = \tilde{\mathbf{t}}$ with perturbation $\boldsymbol{\gamma}$ are*

$$\mathbf{x}_{\text{TLS}}^\dagger = \mathbf{x}^\# + \left(\tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \mathbf{D} \tilde{\mathbf{A}} \right)^{-1} \tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \mathbf{D} \boldsymbol{\gamma} + \mathcal{O}(\|\boldsymbol{\gamma}\|_2^2) \quad (\text{B.9})$$

$$\mathbf{x}_{\text{LS}}^\dagger = \mathbf{x}^\# + \left(\tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \tilde{\mathbf{A}} \right)^{-1} \tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \boldsymbol{\gamma} + \mathcal{O}(\|\boldsymbol{\gamma}\|_2^2). \quad (\text{B.10})$$

Proof. To compute the Taylor series expansions we require $\mathbf{x}_{\text{TLS}}^\dagger(\tilde{\mathbf{t}})$ and $\nabla_{\mathbf{t}} \mathbf{x}_{\text{TLS}}^\dagger(\mathbf{t})|_{\mathbf{t}=\tilde{\mathbf{t}}} \boldsymbol{\gamma}$ for the TLS problem (4.29) and the corresponding terms for the LS problem, $\mathbf{x}_{\text{LS}}^\dagger(\tilde{\mathbf{t}})$ and $\nabla_{\mathbf{t}} \mathbf{x}_{\text{LS}}^\dagger(\mathbf{t})|_{\mathbf{t}=\tilde{\mathbf{t}}} \boldsymbol{\gamma}$

In the no error setting, when $\boldsymbol{\gamma} = 0$, $\mathbf{x}_{\text{TLS}}^\dagger(\tilde{\mathbf{t}}) = \mathbf{x}^\#$. This is because with no error the minimum objective function (4.32) value of zero is achievable with $\mathbf{e}_m = 0$ for all m and $\mathbf{x}_{\text{TLS}}^\dagger(\mathbf{t}) = \mathbf{x}^\#$. Similarly for the LS problem, when $\boldsymbol{\gamma} = 0$, the LS solution, $\mathbf{x}_{\text{LS}}^\dagger(\tilde{\mathbf{t}})$, is also $\mathbf{x}^\#$, as this achieves the minimum LS objective function (4.38) value of zero.

The full derivations for the derivatives $\nabla_{\mathbf{t}} \mathbf{x}_{\text{TLS}}^\dagger(\mathbf{t})$ and $\nabla_{\mathbf{t}} \mathbf{x}_{\text{LS}}^\dagger(\mathbf{t})$ using (4.37) are contained in Appendices B.3.1 and B.3.2. Appendices B.3.3 and B.3.4 then evaluate these derivatives at $\tilde{\mathbf{t}}$ and multiply them by $\boldsymbol{\gamma}$. We again use the fact that at $\tilde{\mathbf{t}}$ the solutions are $\mathbf{x}^\#$ and that the TLS sensing vector corrections, \mathbf{e}_m , are zero for all m . \square

B.3.1 Gradients for TLS problem

For convenience, we restate the optimization problem (4.32),

$$\begin{aligned} \mathbf{q}^\dagger = \arg \min_{\mathbf{q}} \underbrace{\sum_{m=1}^M \lambda_a \|\mathbf{e}_m\|_2^2 + \lambda_y (y_m - |\langle \mathbf{a}_m + \mathbf{e}_m, \mathbf{x} \rangle|)^2}_{f(\mathbf{q}, \mathbf{t})} \\ \text{s.t. } \mathbf{q} = \begin{bmatrix} \mathbf{e}_1^T & \cdots & \mathbf{e}_M^T & \mathbf{x}^T \end{bmatrix}^T \in \mathbb{R}^{MN+N}. \end{aligned} \quad (\text{B.11})$$

We denote the quantities

$$\hat{\mathbf{a}}_m = \mathbf{a}_m + \mathbf{e}_m \in \mathbb{R}^N \quad (\text{B.12})$$

$$d_m = (|\langle \hat{\mathbf{a}}_m, \mathbf{x} \rangle|^2 - y_m) \langle \hat{\mathbf{a}}_m, \mathbf{x} \rangle \in \mathbb{R} \quad (\text{B.13})$$

$$l_m = |\langle \hat{\mathbf{a}}_m, \mathbf{x} \rangle|^2 - y_m \in \mathbb{R} \quad (\text{B.14})$$

$$m_m = 2|\langle \hat{\mathbf{a}}_m, \mathbf{x} \rangle|^2 \in \mathbb{R} \quad (\text{B.15})$$

$$h_m = l_m + m_m = 3|\langle \hat{\mathbf{a}}_m, \mathbf{x} \rangle|^2 - y_m \in \mathbb{R} \quad (\text{B.16})$$

$$p_m = \frac{2\lambda_y}{\lambda_a} d_m \in \mathbb{R} \quad (\text{B.17})$$

$$\phi_m = \frac{h_m}{1 + \frac{2\lambda_y}{\lambda_a} h_m \|\mathbf{x}\|_2^2} \in \mathbb{R} \quad (\text{B.18})$$

which are used to denote

$$\mathbf{B} = \frac{\lambda_a}{2\lambda_y} \mathbf{I}_{MN} + \begin{bmatrix} h_1 \mathbf{x} \mathbf{x}^T & & \\ & \ddots & \\ & & h_M \mathbf{x} \mathbf{x}^T \end{bmatrix} \in \mathbb{R}^{MN \times MN} \quad (\text{B.19})$$

$$\mathbf{C} = \begin{bmatrix} d_1 \mathbf{I}_N + h_1 \mathbf{x} (\mathbf{a}_1 + \mathbf{e}_1)^T \\ \vdots \\ d_M \mathbf{I}_N + h_M \mathbf{x} (\mathbf{a}_M + \mathbf{e}_M)^T \end{bmatrix} \in \mathbb{R}^{MN \times N} \quad (\text{B.20})$$

$$\mathbf{T} = \sum_{m=1}^M h_m \hat{\mathbf{a}}_m \hat{\mathbf{a}}_m^T \in \mathbb{R}^{N \times N} \quad (\text{B.21})$$

$$\hat{\mathbf{A}} = (\mathbf{A} + \mathbf{E})^T \in \mathbb{R}^{N \times M} \quad (\text{B.22})$$

$$\mathbf{p} = [p_1 \ \cdots \ p_M]^T \in \mathbb{R}^M \quad (\text{B.23})$$

$$\Phi = \text{diag}(\phi_1, \dots, \phi_M) \in \mathbb{R}^{M \times M}. \quad (\text{B.24})$$

The first derivative of the objective function with respect to \mathbf{q} , $\nabla_{\mathbf{q}} f(\mathbf{q}, \mathbf{t}) \in \mathbb{R}^{MN+N}$, is

$$\nabla_{\mathbf{q}} f(\mathbf{q}, \mathbf{t}) = 2\lambda_a \begin{bmatrix} \mathbf{e}_1 \\ \vdots \\ \mathbf{e}_M \\ \mathbf{0} \end{bmatrix} + 4\lambda_y \begin{bmatrix} d_1 \mathbf{x} \\ \vdots \\ d_M \mathbf{x} \\ \sum_{m=1}^M d_m \hat{\mathbf{a}}_m \end{bmatrix}. \quad (\text{B.25})$$

The second derivative of the objective function with respect to \mathbf{q} , $\nabla_{\mathbf{q}\mathbf{q}}^2 f(\mathbf{q}, \mathbf{t}) \in \mathbb{R}^{(MN+N) \times (MN+N)}$, is

$$\begin{aligned}
& \nabla_{\mathbf{q}\mathbf{q}}^2 f(\mathbf{q}, \mathbf{t}) \\
&= 2\lambda_a \begin{bmatrix} \mathbf{I}_N & & & \vdots \\ & \ddots & & \mathbf{0}_{MN \times N} \\ & & \mathbf{I}_N & \vdots \\ \cdots & \mathbf{0}_{N \times MN} & \cdots & \mathbf{0}_{N \times N} \end{bmatrix} \\
&+ 4\lambda_y \begin{bmatrix} \ddots & & & d_1 \mathbf{I}_N \\ & \mathbf{0}_{MN \times MN} & & \vdots \\ & & \ddots & d_M \mathbf{I}_N \\ d_1 \mathbf{I}_N & \cdots & d_M \mathbf{I}_N & \mathbf{0}_{N \times N} \end{bmatrix} \\
&+ 4\lambda_y \begin{bmatrix} l_1 \mathbf{x}\mathbf{x}^T & & & l_1 \mathbf{x}\hat{\mathbf{a}}_1^T \\ & \ddots & & \vdots \\ & & l_M \mathbf{x}\mathbf{x}^T & l_M \mathbf{x}\hat{\mathbf{a}}_M^T \\ l_1 \hat{\mathbf{a}}_1 \mathbf{x}^T & \cdots & l_M \hat{\mathbf{a}}_M \mathbf{x}^T & \sum_{i=1}^M l_i \hat{\mathbf{a}}_i \hat{\mathbf{a}}_i^T \end{bmatrix} \\
&+ 4\lambda_y \begin{bmatrix} m_1 \mathbf{x}\mathbf{x}^T & & & m_1 \mathbf{x}\hat{\mathbf{a}}_1^T \\ & \ddots & & \vdots \\ & & m_M \mathbf{x}\mathbf{x}^T & m_M \mathbf{x}\hat{\mathbf{a}}_M^T \\ m_1 \hat{\mathbf{a}}_1 \mathbf{x}^T & \cdots & m_M \hat{\mathbf{a}}_M \mathbf{x}^T & \sum_{i=1}^M m_i \hat{\mathbf{a}}_i \hat{\mathbf{a}}_i^T \end{bmatrix} \\
&= 4\lambda_y \begin{bmatrix} \mathbf{B} & \mathbf{C} \\ \mathbf{C}^T & \mathbf{T} \end{bmatrix}. \tag{B.26}
\end{aligned}$$

The second derivative with respect to y_k , $\frac{d}{dy_k} \nabla_{\mathbf{q}} f(\mathbf{q}, \mathbf{t}) \in \mathbb{R}^{MN+N}$, is

$$\frac{d}{dy_k} \nabla_{\mathbf{q}} f(\mathbf{q}, \mathbf{t}) = -4\lambda_y \begin{bmatrix} \mathbf{0}_{(k-1)N \times N} \\ \mathbf{x}\hat{\mathbf{a}}_k^T \mathbf{x} \\ \mathbf{0}_{(M-k)N \times N} \\ \hat{\mathbf{a}}_k \hat{\mathbf{a}}_k^T \mathbf{x} \end{bmatrix}. \tag{B.27}$$

The second derivative with respect to \mathbf{a}_k , $\nabla_{\mathbf{a}_k \mathbf{q}}^2 f(\mathbf{q}, \mathbf{t}) \in \mathbb{R}^{(MN+N) \times N}$, is

$$\begin{aligned}
& \nabla_{\mathbf{a}_k \mathbf{q}}^2 f(\mathbf{q}, \mathbf{t}) \\
&= 4\lambda_y l_k \begin{bmatrix} \mathbf{0}_{(k-1)N \times N} \\ \mathbf{x}\mathbf{x}^T \\ \mathbf{0}_{(M-k)N \times N} \\ \widehat{\mathbf{a}}_k \mathbf{x}^T + \langle \widehat{\mathbf{a}}_k, \mathbf{x} \rangle \mathbf{I}_N \end{bmatrix} + 4\lambda_y m_k \begin{bmatrix} \mathbf{0}_{(k-1)N \times N} \\ \mathbf{x}\mathbf{x}^T \\ \mathbf{0}_{(M-k)N \times N} \\ \widehat{\mathbf{a}}_k \mathbf{x}^T \end{bmatrix} \\
&= 4\lambda_y \begin{bmatrix} \mathbf{0}_{(k-1)N \times N} \\ h_k \mathbf{x}\mathbf{x}^T \\ \mathbf{0}_{(M-k)N \times N} \\ h_k \widehat{\mathbf{a}}_k \mathbf{x}^T + d_k \mathbf{I}_N \end{bmatrix}. \tag{B.28}
\end{aligned}$$

We will require the inverse of the second derivative (B.26), $(\nabla_{\mathbf{q}\mathbf{q}}^2 f(\mathbf{q}, \mathbf{t}))^{-1} \in \mathbb{R}^{(MN+N) \times (MN+N)}$, in our calculations (4.35) (4.36). We can use blockwise matrix inversion to invert the block matrix (B.26),

$$\begin{aligned}
& (\nabla_{\mathbf{q}\mathbf{q}}^2 f(\mathbf{q}, \mathbf{t}))^{-1} \\
&= \frac{1}{4\lambda_y} \begin{bmatrix} \mathbf{B}^{-1} + \mathbf{Q}_{CB}^T \mathbf{Q}_S^{-1} \mathbf{Q}_{CB} & -\mathbf{Q}_{CB}^T \mathbf{Q}_S^{-1} \\ -\mathbf{Q}_S^{-1} \mathbf{Q}_{CB} & \mathbf{Q}_S^{-1} \end{bmatrix}, \tag{B.29}
\end{aligned}$$

where

$$\mathbf{Q}_{CB} = \mathbf{C}^T \mathbf{B}^{-1} \in \mathbb{R}^{N \times MN} \tag{B.30}$$

$$\begin{aligned}
\mathbf{Q}_S &= \mathbf{T} - \mathbf{C}^T \mathbf{B}^{-1} \mathbf{C} \\
&= \mathbf{T} - \mathbf{Q}_{CB} \mathbf{C} \in \mathbb{R}^{N \times N} \tag{B.31}
\end{aligned}$$

and $\mathbf{Q}_S = \mathbf{T} - \mathbf{C}^T \mathbf{B}^{-1} \mathbf{C}$ is the Schur complement of \mathbf{B} . Furthermore because \mathbf{B} is a block diagonal matrix, \mathbf{B}^{-1} is also block diagonal with each block being the inverse of its counterpart block in \mathbf{B} . Each block in \mathbf{B} has the same structure and, due to this structure, the Sherman-Morrison formula can be used to invert each block,

$$\left(\frac{\lambda_a}{2\lambda_y} \mathbf{I}_N + h_m \mathbf{x}\mathbf{x}^T \right)^{-1} = \frac{2\lambda_y}{\lambda_a} \mathbf{I}_N - \frac{\frac{4\lambda_y^2}{\lambda_a^2} h_m \mathbf{x}\mathbf{x}^T}{1 + \frac{2\lambda_y}{\lambda_a} h_m \|\mathbf{x}\|_2^2} \tag{B.32}$$

and $(\mathbf{B}^{-1})^T = \mathbf{B}^{-1}$.

As we wish to understand the sensitivity of $\mathbf{x}_{\text{TLS}}^\dagger$ (4.34), we only require the final N rows of the inverse of (B.26). More precisely we will only require the submatrix

$$\begin{aligned} & (\nabla_{\mathbf{q}\mathbf{q}}^2 f(\mathbf{q}, \mathbf{t}))_{-N}^{-1} \\ &= \frac{1}{4\lambda_y} \mathbf{Q}_S^{-1} \begin{bmatrix} -\mathbf{Q}_{CB} & \mathbf{I}_N \end{bmatrix} \in \mathbb{R}^{N \times (MN+N)}. \end{aligned} \quad (\text{B.33})$$

To calculate (B.33) we require \mathbf{Q}_{CB} which is a block matrix with M matrices horizontally stacked. The m th block is

$$\begin{aligned} & (d_m \mathbf{I}_N + h_m \widehat{\mathbf{a}}_m \mathbf{x}^T) \left(\frac{2\lambda_y}{\lambda_a} \mathbf{I}_N - \frac{4\lambda_y^2}{\lambda_a^2} \phi_m \mathbf{x} \mathbf{x}^T \right) \\ &= p_m \mathbf{I}_N - \frac{2\lambda_y}{\lambda_a} \phi_m p_m \mathbf{x} \mathbf{x}^T \\ &\quad + \frac{2\lambda_y}{\lambda_a} \phi_m \left(\frac{h_m}{\phi_m} - \frac{2\lambda_y}{\lambda_a} h_m \|\mathbf{x}\|_2^2 \right) \widehat{\mathbf{a}}_m \mathbf{x}^T \\ &= p_m \mathbf{I}_N + \frac{2\lambda_y}{\lambda_a} \phi_m (\widehat{\mathbf{a}}_m - p_m \mathbf{x}) \mathbf{x}^T. \end{aligned} \quad (\text{B.34})$$

To obtain \mathbf{Q}_S in (B.33) we can use (B.34) and the block matrix structure of \mathbf{C} to calculate $\mathbf{C}^T \mathbf{B}^{-1} \mathbf{C} = \mathbf{Q}_{CB} \mathbf{C}$,

$$\begin{aligned} & \mathbf{C}^T \mathbf{B}^{-1} \mathbf{C} = \mathbf{Q}_{CB} \mathbf{C} \\ &= \sum_{m=1}^M \left(p_m \mathbf{I}_N + \frac{2\lambda_y}{\lambda_a} \phi_m (\widehat{\mathbf{a}}_m - p_m \mathbf{x}) \mathbf{x}^T \right) \\ &\quad \left(d_m \mathbf{I}_N + h_m \mathbf{x} \widehat{\mathbf{a}}_m^T \right) \\ &= \sum_{m=1}^M \frac{\lambda_a}{2\lambda_y} p_m^2 \mathbf{I}_N + p_m \phi_m \widehat{\mathbf{a}}_m \mathbf{x}^T - p_m^2 \phi_m \mathbf{x} \mathbf{x}^T \\ &\quad + p_m \phi_m \left(\frac{h_m}{\phi_m} - \frac{2\lambda_y}{\lambda_a} h_m \|\mathbf{x}\|_2^2 \right) \mathbf{x} \widehat{\mathbf{a}}_m^T \\ &\quad + \frac{2\lambda_y}{\lambda_a} h_m \phi_m \|\mathbf{x}\|_2^2 \widehat{\mathbf{a}}_m \widehat{\mathbf{a}}_m^T \\ &= \frac{\lambda_a}{2\lambda_y} \|\mathbf{p}\|^2 \mathbf{I}_N + \widehat{\mathbf{A}} \Phi \mathbf{p} \mathbf{x}^T - \mathbf{x} \mathbf{p}^T \Phi \mathbf{p} \mathbf{x}^T + \mathbf{x} \mathbf{p}^T \Phi \widehat{\mathbf{A}}^T \\ &\quad + \sum_{m=1}^M \frac{2\lambda_y}{\lambda_a} h_m \phi_m \|\mathbf{x}\|_2^2 \widehat{\mathbf{a}}_m \widehat{\mathbf{a}}_m^T. \end{aligned} \quad (\text{B.35})$$

Then the Schur complement of \mathbf{B} is

$$\begin{aligned}
\mathbf{Q}_S &= \mathbf{T} - \mathbf{C}^T \mathbf{B}^{-1} \mathbf{C} \\
&= -\frac{\lambda_a}{2\lambda_y} \|\mathbf{p}\|^2 \mathbf{I}_N - \widehat{\mathbf{A}} \Phi \mathbf{p} \mathbf{x}^T + \mathbf{x} \mathbf{p}^T \Phi \mathbf{p} \mathbf{x}^T - \mathbf{x} \mathbf{p}^T \Phi \widehat{\mathbf{A}}^T \\
&\quad + \sum_{m=1}^M \phi_m \left(\frac{h_m}{\phi_m} - \frac{2\lambda_y}{\lambda_a} h_m \|\mathbf{x}\|_2^2 \right) \widehat{\mathbf{a}}_m \widehat{\mathbf{a}}_m^T \\
&= -\frac{\lambda_a}{2\lambda_y} \|\mathbf{p}\|^2 \mathbf{I}_N - \widehat{\mathbf{A}} \Phi \mathbf{p} \mathbf{x}^T + \mathbf{x} \mathbf{p}^T \Phi \mathbf{p} \mathbf{x}^T - \mathbf{x} \mathbf{p}^T \Phi \widehat{\mathbf{A}}^T \\
&\quad + \widehat{\mathbf{A}} \Phi \widehat{\mathbf{A}}^T \\
&= -\frac{\lambda_a}{2\lambda_y} \|\mathbf{p}\|_2^2 \mathbf{I}_N + (\widehat{\mathbf{A}} - \mathbf{x} \mathbf{p}^T) \Phi (\widehat{\mathbf{A}} - \mathbf{x} \mathbf{p}^T)^T. \tag{B.36}
\end{aligned}$$

Using (B.33) with (B.36) and (B.34) we can compute the last N rows of (4.36), $\frac{d}{dy_k} \mathbf{x}_{\text{TLS}}^\dagger(\mathbf{t}) \in \mathbb{R}^N$. First,

$$\begin{aligned}
& -\frac{1}{4\lambda_y} \begin{bmatrix} -\mathbf{Q}_{CB} & \mathbf{I}_N \end{bmatrix} \frac{d}{dy_k} \nabla_{\mathbf{q}} f(\mathbf{q}, \mathbf{t}) \\
&= \begin{bmatrix} -p_k \mathbf{I}_N - \frac{2\lambda_y}{\lambda_a} \phi_k (\widehat{\mathbf{a}}_k - p_k \mathbf{x}) \mathbf{x}^T & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{x} \widehat{\mathbf{a}}_k^T \mathbf{x} \\ \widehat{\mathbf{a}}_k \widehat{\mathbf{a}}_k^T \mathbf{x} \end{bmatrix} \\
&= \widehat{\mathbf{a}}_k^T \mathbf{x} \begin{bmatrix} -p_k \mathbf{I}_N - \frac{2\lambda_y}{\lambda_a} \phi_k (\widehat{\mathbf{a}}_k - p_k \mathbf{x}) \mathbf{x}^T & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \widehat{\mathbf{a}}_k \end{bmatrix} \\
&= \widehat{\mathbf{a}}_k^T \mathbf{x} \left(-p_k \mathbf{x} - \frac{2\lambda_y}{\lambda_a} \phi_k (\widehat{\mathbf{a}}_k - p_k \mathbf{x}) \|\mathbf{x}\|_2^2 + \widehat{\mathbf{a}}_k \right) \\
&= \widehat{\mathbf{a}}_k^T \mathbf{x} (\widehat{\mathbf{a}}_k - p_k \mathbf{x}) \frac{\phi_k}{h_k}, \tag{B.37}
\end{aligned}$$

and therefore,

$$\begin{aligned}
& \frac{d}{dy_k} \mathbf{x}_{\text{TLS}}^\dagger(\mathbf{t}) \\
&= \left(-\frac{\lambda_a}{2\lambda_y} \|\mathbf{p}\|_2^2 \mathbf{I}_N + (\widehat{\mathbf{A}} - \mathbf{x} \mathbf{p}^T) \Phi (\widehat{\mathbf{A}} - \mathbf{x} \mathbf{p}^T)^T \right)^{-1} \\
&\quad \widehat{\mathbf{a}}_k^T \mathbf{x} (\widehat{\mathbf{a}}_k - p_k \mathbf{x}) \frac{\phi_k}{h_k}. \tag{B.38}
\end{aligned}$$

Similarly using (B.33) with (B.36) and (B.34) we can compute the last N rows of (4.35), $\nabla_{\mathbf{a}_k} \mathbf{x}_{\text{TLS}}^\dagger(\mathbf{t}) \in \mathbb{R}^{N \times N}$. Again first,

$$\begin{aligned}
& -\frac{1}{4\lambda_y} \begin{bmatrix} -\mathbf{Q}_{CB} & \mathbf{I}_N \end{bmatrix} \nabla_{\mathbf{a}_k \mathbf{q}}^2 f(\mathbf{q}, \mathbf{t}) \\
& = -\begin{bmatrix} -p_k \mathbf{I}_N - \frac{2\lambda_y}{\lambda_a} \phi_k (\widehat{\mathbf{a}}_k - p_k \mathbf{x}) \mathbf{x}^T & \mathbf{I} \end{bmatrix} \\
& \quad \begin{bmatrix} h_k \mathbf{x} \mathbf{x}^T \\ h_k \widehat{\mathbf{a}}_k \mathbf{x}^T + \frac{\lambda_a}{2\lambda_y} p_k \mathbf{I}_N \end{bmatrix} \\
& = -\left(-p_k h_k \mathbf{x} \mathbf{x}^T - \frac{2\lambda_y}{\lambda_a} h_k \phi_k \|\mathbf{x}\|_2^2 (\widehat{\mathbf{a}}_k - p_k \mathbf{x}) \mathbf{x}^T \right. \\
& \quad \left. + h_k \widehat{\mathbf{a}}_k \mathbf{x}^T + \frac{\lambda_a}{2\lambda_y} p_k \mathbf{I}_N \right) \\
& = -\left(\frac{\lambda_a}{2\lambda_y} p_k \mathbf{I}_N + h_k (\widehat{\mathbf{a}}_k - p_k \mathbf{x}) \mathbf{x}^T \right. \\
& \quad \left. - \frac{2\lambda_y}{\lambda_a} h_k \phi_k \|\mathbf{x}\|_2^2 (\widehat{\mathbf{a}}_k - p_k \mathbf{x}) \mathbf{x}^T \right) \\
& = -\left(\frac{\lambda_a}{2\lambda_y} p_k \mathbf{I}_N \right. \\
& \quad \left. + \phi_k \left(\frac{h_k}{\phi_k} - \frac{2\lambda_y}{\lambda_a} h_k \|\mathbf{x}\|_2^2 \right) (\widehat{\mathbf{a}}_k - p_k \mathbf{x}) \mathbf{x}^T \right) \\
& = -\left(\frac{\lambda_a}{2\lambda_y} p_k \mathbf{I}_N + \phi_k (\widehat{\mathbf{a}}_k - p_k \mathbf{x}) \mathbf{x}^T \right), \tag{B.39}
\end{aligned}$$

and therefore,

$$\begin{aligned}
& \nabla_{\mathbf{a}_k} \mathbf{x}_{\text{TLS}}^\dagger(\mathbf{t}) \\
& = -\left(-\frac{\lambda_a}{2\lambda_y} \|\mathbf{p}\|_2^2 \mathbf{I}_N + (\widehat{\mathbf{A}} - \mathbf{x} \mathbf{p}^T) \Phi (\widehat{\mathbf{A}} - \mathbf{x} \mathbf{p}^T)^T \right)^{-1} \\
& \quad \left(\frac{\lambda_a}{2\lambda_y} p_k \mathbf{I}_N + \phi_k (\widehat{\mathbf{a}}_k - p_k \mathbf{x}) \mathbf{x}^T \right). \tag{B.40}
\end{aligned}$$

B.3.2 Gradients for LS problem

We restate the optimization problem (4.38)

$$\mathbf{x}_{\text{LS}}^\dagger(\mathbf{t}) = \arg \min_{\mathbf{x}} \underbrace{\sum_{m=1}^M (y_m - |\langle \mathbf{a}_m, \mathbf{x} \rangle|^2)^2}_{s(\mathbf{x}, \mathbf{t})}. \quad (\text{B.41})$$

We denote similar quantities to those in the TLS derivation. The main differences are that there are no \mathbf{e}_m , λ_y and λ_a in the LS approach,

$$d_m^- = (|\langle \mathbf{a}_m, \mathbf{x} \rangle|^2 - y_m) \langle \mathbf{a}_m, \mathbf{x} \rangle \in \mathbb{R} \quad (\text{B.42})$$

$$h_m^- = 3|\langle \mathbf{a}_m, \mathbf{x} \rangle|^2 - y_m \in \mathbb{R} \quad (\text{B.43})$$

$$\mathbf{A}^- = \mathbf{A}^T \in \mathbb{R}^{N \times M} \quad (\text{B.44})$$

$$\mathbf{p}^- = 2 \left[d_1^-, \dots, d_M^- \right]^T \in \mathbb{R}^M \quad (\text{B.45})$$

$$\mathbf{\Phi}^- = \text{diag}(h_1^-, \dots, h_M^-) \in \mathbb{R}^{M \times M}. \quad (\text{B.46})$$

To derive $\frac{d}{dy_k} \mathbf{x}_{\text{LS}}^\dagger(\mathbf{t}) \in \mathbb{R}^N$ and $\nabla_{\mathbf{a}_k} \mathbf{x}_{\text{LS}}^\dagger(\mathbf{t}) \in \mathbb{R}^{N \times N}$ for LS, the expressions that were derived for TLS can be used. Set $\{\mathbf{e}_m\}_{m=1}^M = 0$, $\lambda_y = \lambda_a = 1$ in the TLS expressions (B.26), (B.27) and (B.28). Then take the bottom right $N \times N$ block of (B.26) and the bottom N rows of (B.27) and (B.28) to get for LS

$$\begin{aligned} \nabla_{\mathbf{x}\mathbf{x}}^2 s(\mathbf{x}, \mathbf{t}) &= 4 \sum_{i=1}^M h_i^- \mathbf{a}_i \mathbf{a}_i^T \\ &= 4 \mathbf{A}^- \mathbf{\Phi}^- (\mathbf{A}^-)^T \in \mathbb{R}^{N \times N} \end{aligned} \quad (\text{B.47})$$

$$\frac{d}{dy_k} \nabla_{\mathbf{x}} s(\mathbf{x}, \mathbf{t}) = -4 \mathbf{a}_k \mathbf{a}_k^T \mathbf{x} = -4 \mathbf{a}_k^- (\mathbf{a}_k^-)^T \mathbf{x} \in \mathbb{R}^N \quad (\text{B.48})$$

$$\begin{aligned} \nabla_{\mathbf{a}_k \mathbf{x}}^2 s(\mathbf{x}, \mathbf{t}) &= 4(h_k^- \mathbf{a}_k \mathbf{x}^T + d_k^- \mathbf{I}_N) \\ &= 4 \left(\frac{1}{2} p_k^- \mathbf{I}_N + \phi_k^- \mathbf{a}_k^- \mathbf{x}^T \right) \in \mathbb{R}^{N \times N}. \end{aligned} \quad (\text{B.49})$$

Therefore

$$\frac{d}{dy_k} \mathbf{x}_{\text{LS}}^\dagger(\mathbf{t}) = (\mathbf{A}^- \Phi^- (\mathbf{A}^-)^T)^{-1} (\mathbf{a}_k^-)^T \mathbf{x} \mathbf{a}_k^- \quad (\text{B.50})$$

$$\begin{aligned} \nabla_{\mathbf{a}_k} \mathbf{x}_{\text{LS}}^\dagger(\mathbf{t}) &= - (\mathbf{A}^- \Phi^- (\mathbf{A}^-)^T)^{-1} \\ &\quad \left(\frac{1}{2} p_k^- \mathbf{I}_N + \phi_k^- \mathbf{a}_k^- \mathbf{x}^T \right). \end{aligned} \quad (\text{B.51})$$

B.3.3 Derivation of TLS solution Taylor series expansion

To calculate $\nabla_{\mathbf{t}} \mathbf{x}_{\text{TLS}}^\dagger(\mathbf{t})|_{\mathbf{t}=\tilde{\mathbf{t}}}$ we need the last N rows of $\nabla_{\mathbf{a}_k} g(\mathbf{t})|_{\mathbf{t}=\tilde{\mathbf{t}}}$ and $\frac{d}{dy_k} g(\mathbf{t})|_{\mathbf{t}=\tilde{\mathbf{t}}}$ for $1 \leq k \leq M$ as in (4.37). With $\mathbf{t} = \tilde{\mathbf{t}}$, $\{\mathbf{e}_m\}_{m=1}^M = 0$ and $\mathbf{x} = \mathbf{x}^\#$, the quantities defined when deriving the gradients in Appendix B.3.1 become

$$\hat{\mathbf{a}}_m = \mathbf{a}_m + \mathbf{e}_m = \tilde{\mathbf{a}}_m \in \mathbb{R}^N \quad (\text{B.52})$$

$$d_m = (|\langle \hat{\mathbf{a}}_m, \mathbf{x} \rangle|^2 - y_m) \langle \hat{\mathbf{a}}_m, \mathbf{x} \rangle = 0 \in \mathbb{R} \quad (\text{B.53})$$

$$l_m = |\langle \hat{\mathbf{a}}_m, \mathbf{x} \rangle|^2 - y_m = 0 \in \mathbb{R} \quad (\text{B.54})$$

$$m_m = 2|\langle \hat{\mathbf{a}}_m, \mathbf{x} \rangle|^2 = 2\tilde{y}_m \in \mathbb{R} \quad (\text{B.55})$$

$$h_m = l_m + m_m = 3|\langle \hat{\mathbf{a}}_m, \mathbf{x} \rangle|^2 - y_m = 2\tilde{y}_m \in \mathbb{R} \quad (\text{B.56})$$

$$p_m = \frac{2\lambda_y}{\lambda_a} d_m = 0 \in \mathbb{R} \quad (\text{B.57})$$

$$\phi_m = \frac{2\tilde{y}_m}{1 + \frac{4\lambda_y}{\lambda_a} \tilde{y}_m \|\mathbf{x}^\#\|_2^2} \in \mathbb{R} \quad (\text{B.58})$$

$$\hat{\mathbf{A}} = (\mathbf{A} + \mathbf{E})^T = \tilde{\mathbf{A}}^T \in \mathbb{R}^{N \times M} \quad (\text{B.59})$$

$$\mathbf{p} = [p_1 \ \cdots \ p_M]^T = [\mathbf{0} \ \cdots \ \mathbf{0}]^T \in \mathbb{R}^M \quad (\text{B.60})$$

$$\Phi = \text{diag}(\phi_1, \dots, \phi_M) = 2\tilde{\mathbf{Y}}\mathbf{D} \in \mathbb{R}^{M \times M}. \quad (\text{B.61})$$

Using these quantities with (B.38) and (B.40) in Appendix B.3.1,

$$\nabla_{\mathbf{a}_k} \mathbf{x}_{\text{TLS}}^\dagger(\mathbf{t})|_{\mathbf{t}=\tilde{\mathbf{t}}} = - \left(\tilde{\mathbf{A}}^T \Phi \tilde{\mathbf{A}} \right)^{-1} (\phi_k \tilde{\mathbf{a}}_k (\mathbf{x}^\#)^T) \quad (\text{B.62})$$

$$\frac{d}{dy_k} \mathbf{x}_{\text{TLS}}^\dagger(\mathbf{t})|_{\mathbf{t}=\tilde{\mathbf{t}}} = \left(\tilde{\mathbf{A}}^T \Phi \tilde{\mathbf{A}} \right)^{-1} \tilde{\mathbf{a}}_k^T \mathbf{x}^\# \tilde{\mathbf{a}}_k \frac{\phi_k}{h_k}, \quad (\text{B.63})$$

Therefore

$$\begin{aligned}
& \nabla_t \mathbf{x}_{\text{TLS}}^\dagger(t) \Big|_{t=\tilde{t}} \gamma \\
&= \left(\tilde{\mathbf{A}}^T \Phi \tilde{\mathbf{A}} \right)^{-1} \\
& \quad \left(\sum_{m=1}^M -\phi_m \tilde{\mathbf{a}}_m (\mathbf{x}^\#)^T \boldsymbol{\delta}_m + \tilde{\mathbf{a}}_m^T \mathbf{x}^\# \tilde{\mathbf{a}}_m \frac{\phi_m}{h_m} \eta_m \right) \\
&= \left(\tilde{\mathbf{A}}^T \Phi \tilde{\mathbf{A}} \right)^{-1} \left(\sum_{m=1}^M \frac{\phi_m}{h_m} \eta_m \tilde{\mathbf{a}}_m \tilde{\mathbf{a}}_m^T - \phi_m \tilde{\mathbf{a}}_m \boldsymbol{\delta}_m^T \right) \mathbf{x}^\# \\
&= \left(\tilde{\mathbf{A}}^T \Phi \tilde{\mathbf{A}} \right)^{-1} \left(\tilde{\mathbf{A}}^T \Phi (2\tilde{\mathbf{Y}})^{-1} \mathbf{E}_Y \tilde{\mathbf{A}} - \tilde{\mathbf{A}}^T \Phi \mathbf{E}_A \right) \mathbf{x}^\# \\
&= \left(\tilde{\mathbf{A}}^T \Phi \tilde{\mathbf{A}} \right)^{-1} \tilde{\mathbf{A}}^T \Phi \left((2\tilde{\mathbf{Y}})^{-1} \mathbf{E}_Y \tilde{\mathbf{A}} - \mathbf{E}_A \right) \mathbf{x}^\# \\
&= \left(\tilde{\mathbf{A}}^T 2\tilde{\mathbf{Y}} D \tilde{\mathbf{A}} \right)^{-1} \tilde{\mathbf{A}}^T 2\tilde{\mathbf{Y}} D \left((2\tilde{\mathbf{Y}})^{-1} \mathbf{E}_Y \tilde{\mathbf{A}} - \mathbf{E}_A \right) \mathbf{x}^\# \\
&= \left(\tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} D \tilde{\mathbf{A}} \right)^{-1} \tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} D \mathbf{w}. \tag{B.64}
\end{aligned}$$

B.3.4 Derivation of LS solution Taylor series expansion

Following the same procedure as in Appendix B.3.3 and using (B.50) and (B.51) in Appendix B.3.2,

$$\nabla_{\mathbf{a}_k} \mathbf{x}_{\text{LS}}^\dagger(t) \Big|_{t=\tilde{t}} = - \left(\tilde{\mathbf{A}}^T 2\tilde{\mathbf{Y}} \tilde{\mathbf{A}} \right)^{-1} (2\tilde{y}_k \tilde{\mathbf{a}}_k (\mathbf{x}^\#)^T) \tag{B.65}$$

$$\frac{d}{dy_k} \mathbf{x}_{\text{LS}}^\dagger(t) \Big|_{t=\tilde{t}} = \left(\tilde{\mathbf{A}}^T 2\tilde{\mathbf{Y}} \tilde{\mathbf{A}} \right)^{-1} \tilde{\mathbf{a}}_k^T \mathbf{x}^\# \tilde{\mathbf{a}}_k. \tag{B.66}$$

Therefore for LS

$$\begin{aligned}
& \nabla_t \mathbf{x}_{\text{LS}}^\dagger(t) \Big|_{t=\tilde{t}} \gamma \\
&= \left(\tilde{\mathbf{A}}^T 2\tilde{\mathbf{Y}} \tilde{\mathbf{A}} \right)^{-1} \left(\sum_{m=1}^M \eta_m \tilde{\mathbf{a}}_m \tilde{\mathbf{a}}_m^T - 2\tilde{y}_m \tilde{\mathbf{a}}_m \boldsymbol{\delta}_m^T \right) \mathbf{x}^\# \\
&= \left(\tilde{\mathbf{A}}^T 2\tilde{\mathbf{Y}} \tilde{\mathbf{A}} \right)^{-1} \left(\tilde{\mathbf{A}}^T 2\tilde{\mathbf{Y}} (2\tilde{\mathbf{Y}})^{-1} \mathbf{E}_Y \tilde{\mathbf{A}} - \tilde{\mathbf{A}}^T 2\tilde{\mathbf{Y}} \mathbf{E}_A \right) \mathbf{x}^\# \\
&= \left(\tilde{\mathbf{A}}^T 2\tilde{\mathbf{Y}} \tilde{\mathbf{A}} \right)^{-1} \tilde{\mathbf{A}}^T 2\tilde{\mathbf{Y}} \left((2\tilde{\mathbf{Y}})^{-1} \mathbf{E}_Y \tilde{\mathbf{A}} - \mathbf{E}_A \right) \mathbf{x}^\# \\
&= \left(\tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \tilde{\mathbf{A}} \right)^{-1} \tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \mathbf{w}. \tag{B.67}
\end{aligned}$$

B.4 Proof of Proposition 5

We begin by noting that e_{TLS}^2 and e_{LS}^2 can both be written in the form $e^2 = (\mathbf{R}\mathbf{w})^T(\mathbf{R}\mathbf{w})$ where $\mathbf{R} \in \mathbb{R}^{N \times M}$ is $(\tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \mathbf{D} \tilde{\mathbf{A}})^{-1} \tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \mathbf{D}$ and $(\tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \tilde{\mathbf{A}})^{-1} \tilde{\mathbf{A}}^T \tilde{\mathbf{Y}}$ for TLS and LS. The vector $\mathbf{w} \in \mathbb{R}^M$ is as defined in Proposition 4 and contains all the random quantities. Let r_{ij} be the i, j th entry of \mathbf{R} and let w_i be the i th entry of \mathbf{w} . Then

$$e^2 = \begin{bmatrix} \sum_{i=1}^M r_{1,i} w_i & \cdots & \sum_{i=1}^M r_{N,i} w_i \end{bmatrix} \begin{bmatrix} \sum_{j=1}^M r_{1,j} w_j \\ \vdots \\ \sum_{j=1}^M r_{N,j} w_j \end{bmatrix} \quad (\text{B.68})$$

$$= \sum_{i=1}^M \sum_{j=1}^M r_{1,i} r_{1,j} w_i w_j + \dots + \sum_{i=1}^M \sum_{j=1}^M r_{N,i} r_{N,j} w_i w_j. \quad (\text{B.69})$$

Further, $w_i = \frac{\eta_i}{2\tilde{y}_i} \langle \tilde{\mathbf{a}}_i, \mathbf{x}^\# \rangle - \langle \boldsymbol{\delta}_i, \mathbf{x}^\# \rangle$ and so all the entries of \mathbf{w} are independent of each other. As a result, $\mathbb{E}[w_i w_j] = 0$ if $i \neq j$ and

$$\mathbb{W}_i := \mathbb{E}[w_i^2] = \mathbb{E}[\eta_i^2] \frac{\langle \tilde{\mathbf{a}}_i, \mathbf{x}^\# \rangle^2}{4\tilde{y}_i^2} + (\mathbf{x}^\#)^T \mathbb{E}[\boldsymbol{\delta}_i \boldsymbol{\delta}_i^T] \mathbf{x}^\# \quad (\text{B.70})$$

$$= \frac{\sigma_\eta^2}{4\tilde{y}_i} + \sigma_\delta^2 \|\mathbf{x}^\#\|_2^2. \quad (\text{B.71})$$

Denoting \mathbf{r}_m as the m th column of \mathbf{R} and using \mathbb{W}_i ,

$$\mathbb{E}[e^2] = \sum_{i=1}^M r_{1,i}^2 \mathbb{W}_i + \dots + \sum_{i=1}^M r_{N,i}^2 \mathbb{W}_i \quad (\text{B.72})$$

$$= \sum_{i=1}^M \mathbb{W}_i (r_{1,i}^2 + \dots + r_{N,i}^2) \quad (\text{B.73})$$

$$= \sum_{i=1}^M \mathbb{W}_i \|\mathbf{r}_i\|_2^2 \quad (\text{B.74})$$

$$= \sigma_\delta^2 \|\mathbf{x}^\#\|_2^2 \sum_{i=1}^M \|\mathbf{r}_i\|_2^2 + \frac{\sigma_\eta^2}{4} \sum_{i=1}^M \frac{1}{\tilde{y}_i} \|\mathbf{r}_i\|_2^2 \quad (\text{B.75})$$

$$= \sigma_\delta^2 \|\mathbf{x}^\#\|_2^2 \|\mathbf{R}\|_F^2 + \frac{\sigma_\eta^2}{4} \left\| \mathbf{R} \tilde{\mathbf{Y}}^{-\frac{1}{2}} \right\|_F^2. \quad (\text{B.76})$$

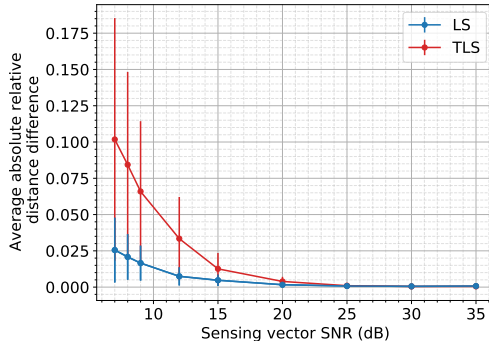


Figure B.1: The average of the absolute value difference between the actual relative distance and the relative error reconstruction from Proposition 4 when $\frac{M}{N} = 8$. Measurement SNR is twice the sensing vector SNR.

The result in Proposition 5 then follows by substituting the TLS and LS values for \mathbf{R} . We also use the fact that the matrix multiplication of $\tilde{\mathbf{Y}}$ and \mathbf{D} in the TLS expression is commutative because both matrices are diagonal.

B.5 Experimental verification of Proposition 4

We verify the derived reconstruction errors in Proposition 4. As the results are for a first-order approximation, we expect their accuracy to reduce as $\|\boldsymbol{\gamma}\|$ increases. To vary $\|\boldsymbol{\gamma}\|$ we vary the sensing vector SNR and pin the measurement SNR to be twice the sensing vector SNR. For each SNR combination we perform 100 trials. In each trial, we generate new real-valued ground truth signals, Gaussian sensing vectors and Gaussian errors for sensing vectors and measurements. The ground truth signals are iid standard real Gaussian with $N = 100$ and $\frac{M}{N} = 8$.

We plot the average of the absolute difference between the relative distance from the solution of Algorithm 2 and the relative reconstruction error, $\left| \text{rel.dist}(\mathbf{x}^\#, \mathbf{x}_{\text{LS}}^\dagger) - \text{rel.}e_{\text{LS}} \right|$ for LS and $\left| \text{rel.dist}(\mathbf{x}^\#, \mathbf{x}_{\text{TLS}}^\dagger) - \text{rel.}e_{\text{TLS}} \right|$ for TLS in Figure B.1. The step sizes are $\frac{1.0}{\lambda_a}$ for TLS and 0.05 for LS. We set $\lambda_a = \frac{1}{N}$ and $\lambda_y = \frac{1}{\|\mathbf{x}^{(0)}\|_2^4}$. As expected the first-order approximations are accurate for high SNR and decrease in accuracy with decreasing SNR. The high accuracy for the moderate to high SNR values also confirms that Algorithm 2 can optimize (TLS-PR2).

B.6 Handcrafted errors

Using the results and notation of Section 4.3, we show that there exist error models which can significantly change the relative performance of TLS and LS. With scalars $k_a, k_y \in \mathbb{R}^+$ to control the SNR of \mathbf{E}_A and \mathbf{E}_Y , we create errors $\mathbf{E}_A = k_a \mathbf{D}^{-1} \mathbf{E}'_A$ for some \mathbf{E}'_A and $\mathbf{E}_Y = k_y \mathbf{D}^{-1} \mathbf{E}'_Y$ for some diagonal \mathbf{E}'_Y . With these errors, the expressions from Proposition 4 become

$$e_{\text{TLS}} = \left\| \left(\tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \mathbf{D} \tilde{\mathbf{A}} \right)^{-1} \tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \mathbf{w}' \right\|_2 \quad (\text{B.77})$$

$$e_{\text{LS}} = \left\| \left(\tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \tilde{\mathbf{A}} \right)^{-1} \tilde{\mathbf{A}}^T \tilde{\mathbf{Y}} \mathbf{D}^{-1} \mathbf{w}' \right\|_2 \quad (\text{B.78})$$

where $\mathbf{w}' = ((2\tilde{\mathbf{Y}})^{-1} k_y \mathbf{E}'_Y \tilde{\mathbf{A}} - k_a \mathbf{E}'_A) \mathbf{x}^\#$. Compared to (4.45), e_{TLS} in (B.77) does not multiply \mathbf{w}' by \mathbf{D} . Additionally, compared to (4.46), e_{LS} in (B.78) multiplies \mathbf{w}' by \mathbf{D}^{-1} . The elements of diagonal matrix \mathbf{D}^{-1} are greater than one, $d_{mm}^{-1} = \left(1 + \frac{4\lambda_y}{\lambda_a} \|\mathbf{x}^\#\|_2^2 \tilde{y}_m \right) > 1$, and we investigate how this alters performance when the sensing vectors follow the iid standard real Gaussian measurement model. Appendix B.7.1 contains experiments using the coded diffraction pattern model.

B.6.1 First-order reconstruction error numerical experiments

In the next set of experiments, $\mathbf{E}_A = k_a \mathbf{D}_1^{-1} \mathbf{E}'_A$ and $\mathbf{E}_Y = k_y \mathbf{D}_1^{-1} \mathbf{E}'_Y$, where $\mathbf{D}_1 = (\mathbf{I}_M + 4 \|\mathbf{x}^\#\|_2^2 \tilde{\mathbf{Y}})^{-1}$ is free of the regularization parameters. Matrix \mathbf{E}'_A and diagonal matrix \mathbf{E}'_Y are iid zero-mean Gaussian. We repeat the experiment of Figure 4.2b using these created errors in Figure B.2. All other experimental details are unchanged. We see that now TLS outperforms LS with this error model.

Next we fix $\frac{M}{N} = 8$ and the sensing vector SNR to 100 dB so there is virtually no sensing vector error. We vary the measurement SNR over 100 trials and use the handcrafted errors. The sensing vectors and ground truth signals are generated in the same way as in the numerical experiments of Section 4.3. In Figure B.3 we plot the average first-order relative reconstruction error and see that with this setting TLS outperforms LS. This occurs despite there only being measurement error, a setting where we may expect LS to outperform TLS.

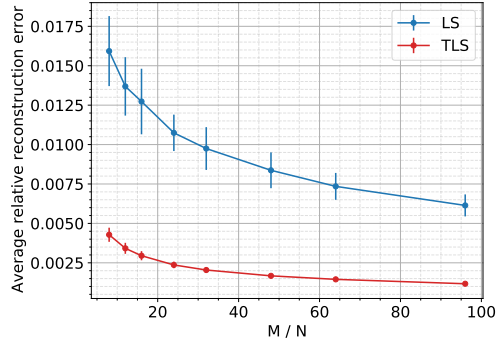


Figure B.2: Relative reconstruction errors, (4.45) and (4.46) for different values of $\frac{M}{N}$ when handcrafted errors are used. Measurement and sensing vector SNRs are 40 dB.

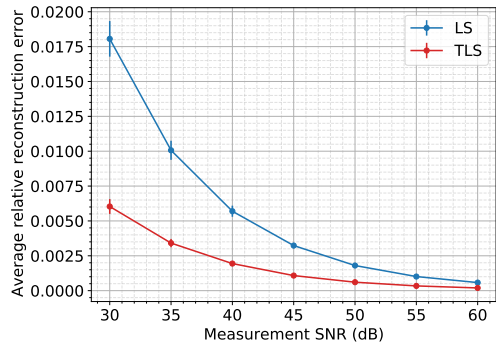


Figure B.3: Relative reconstruction errors with handcrafted errors when there is only error in measurements. Here $\frac{M}{N} = 8$.

The results in Figures B.2 and B.3 show that the type of measurement and sensing vector error can impact performance.

B.6.2 Simulations with actual reconstruction error

To investigate the impact of this error model on the actual reconstruction error, we design handcrafted errors in the same manner as above and calculate the actual reconstruction error. Despite the earlier analysis using real-valued errors, we show that the ideas carry through when we use the complex Gaussian measurement model with \mathbf{E}_A being complex Gaussian.

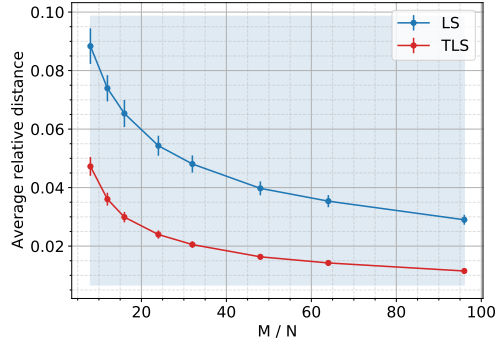


Figure B.4: Relative distance of reconstructions using TLS and LS for the Gaussian measurement model for different values of $\frac{M}{N}$ when handcrafted errors are used. Measurement SNR is 20 dB and sensing vector SNR is 30 dB.

In the first simulation, we use a step size of $\frac{0.5}{\lambda_a}$ for TLS and 0.02 for LS and repeat the experiment of Figure 4.5b with handcrafted errors instead. Figure B.4 shows that in this setting TLS outperforms LS, the opposite of what is seen with Gaussian errors in Figure 4.5b. This is consistent with the first-order reconstruction error numerical experiment of Figure B.2.

Next we use $\frac{M}{N} = 8$ and a step size of $\frac{0.2}{\lambda_a}$ for TLS and 0.02 for LS. Following the experiment of Figure B.3, in Figure B.5 the sensing vector SNR is 100 dB and there is virtually no sensing vector error. The measurement SNR is varied and the performance of TLS and LS with handcrafted errors is compared to TLS and LS with iid Gaussian errors. We do 100 trials at each measurement SNR. Even though there is significant error only in the measurements, TLS with handcrafted errors outperforms LS as was suggested by Figure B.3. With Gaussian errors, LS outperforms TLS when there are only measurement errors.

With $\frac{M}{N} = 8$ and handcrafted errors, Figure B.6 shows an identical experiment to that of Figure 4.4. We see that the relative performance of TLS over LS improves with handcrafted errors compared to Figure 4.4 where random Gaussian errors were used.

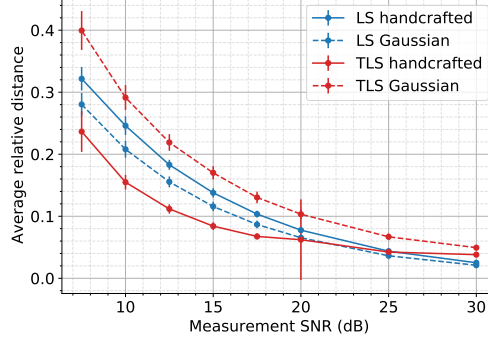


Figure B.5: Performance of TLS and LS with handcrafted errors when there is only error in measurements for the Gaussian measurement model. Here $\frac{M}{N} = 8$.

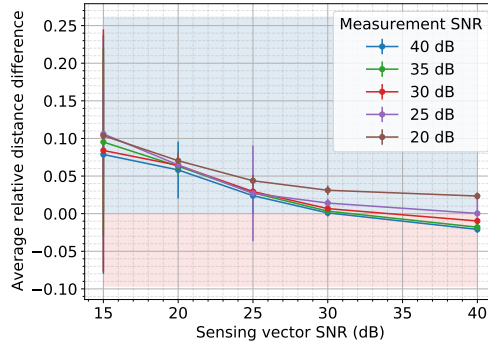


Figure B.6: Average difference in relative distance between TLS and LS solutions for the Gaussian measurement model when $\frac{M}{N} = 8$ for different measurement and sensing vector SNR combinations when the errors are handcrafted. This can be compared to Figure 4.4 when $\frac{M}{N} = 8$.

B.7 Coded diffraction pattern (CDP) measurement model

Denoting row n of the N -point DFT matrix as \mathbf{f}_n^* and L modulation patterns $\{\mathbf{p}_l\}_{l=1}^L \in \mathbb{C}^N$, the $M = LN$ quadratic coded diffraction pattern measurements are then

$$y_m \approx \left| \underbrace{\mathbf{f}_n^* \text{diag}(\mathbf{p}_l)^*}_{\mathbf{a}_m^*} \mathbf{x} \right|^2, \quad m = (n, l) \quad (\text{B.79})$$

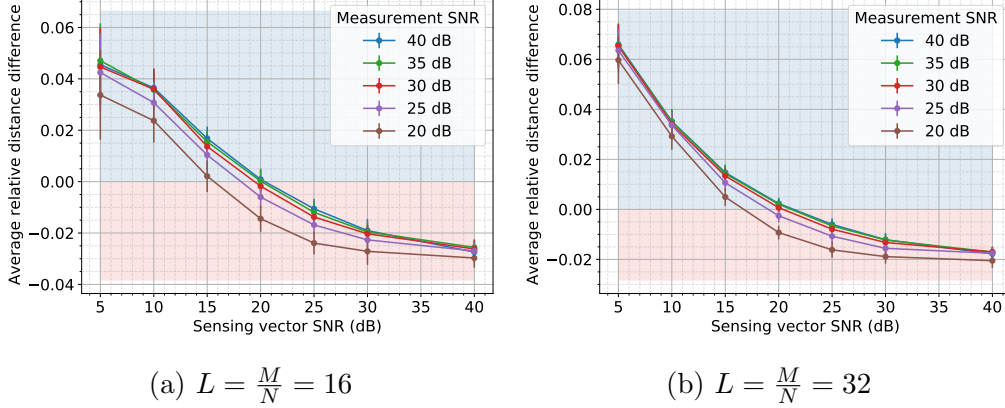


Figure B.7: Average difference in relative distance of TLS and LS solutions, $\text{rel.dist}(\mathbf{x}^\#, \mathbf{x}_{\text{LS}}^\dagger) - \text{rel.dist}(\mathbf{x}^\#, \mathbf{x}_{\text{TLS}}^\dagger)$, for the octanary CDP measurement model for different measurement and sensing vector SNR combinations when the number of patterns is $L = \frac{M}{N} \in \{16, 32\}$.

where $1 \leq n \leq N$, $1 \leq l \leq L$ [171]. The modulation patterns $\mathbf{p}_l \in \mathbb{C}^N$ follow the octanary pattern which means its entries are iid and follow the distribution of p where $p = q_1 q_2$. The random variable q_1 is one of $\{-1, 1, -j, j\}$ with equal probability and $q_2 = \frac{\sqrt{2}}{2}$ with probability 0.8 or $q_2 = \sqrt{3}$ with probability 0.2. Note that M can only be an integer multiple of N and depends on the number of patterns used.

B.7.1 Experiments

In this section, we repeat the experiments that were done for the Gaussian measurement model in Section 4.4 for the CDP measurement model. The experimental setup such as the number of trials, type of ground truth signal, step sizes, and iteration stopping criteria are the same as those used for the equivalent simulation with the Gaussian model.

Random errors The experiments in Figures B.7, B.8a, B.8b are for the CDP measurement model and are the same as Figures 4.4, 4.5a and 4.5b for the Gaussian measurement model.

Handcrafted errors Figures B.8c, B.9 and B.10 show the performance with handcrafted errors for the CDP measurement model. With only handcrafted measurement error and 100 dB SNR sensing vector error, the performance of TLS is better than LS for low measurement SNR in Figure B.9 compared to when there are random Gaussian errors. The performance with different error combinations for handcrafted errors in Figure B.10 should be compared against Figure B.7.

Sensing vector correction verification The experiments done to verify the sensing vector corrections for the Gaussian measurement model in Figures 4.6 and 4.7 are done for the CDP measurement model in Figures B.11 and B.12.

B.8 Additional OPU experiment information

B.8.1 Sensing vector calibration

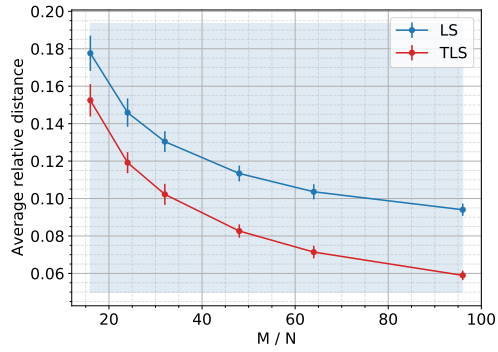
We use the method described in Chapter 3 of this thesis to calibrate the sensing vectors of the OPU's TM. We implemented this method with $1.5N$ calibration signals and 20 anchor signals. We use the same procedure as Chapter 3 to design the calibration signals.

B.8.2 Experiment details

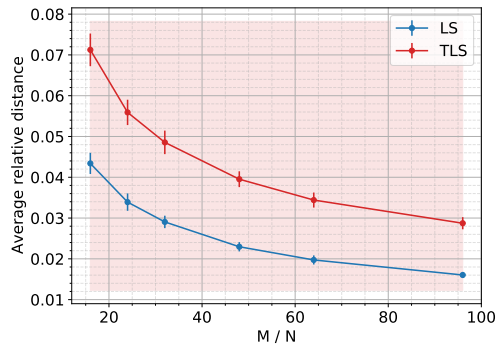
When doing the experiments with random images on the OPU, we set the camera exposure time to $700 \mu s$ to utilize and not saturate the full zero to 255 8-bit measurement range of the camera. The input display framerate is set to $1200 \mu s$. For the experiments with the real images, the camera exposure is $400 \mu s$ and the framerate is $500 \mu s$.

To use a new set of sensing vectors in each trial, we calibrate a complex-valued transmission matrix with 2^{17} rows. In each trial in Figure 4.9a, we then do phase retrieval by choosing a new set of M rows. The optical measurements corresponding to the chosen M rows are used. As previously, the TLS and LS iterations in Algorithm 2 are stopped when the objective function value between successive iterates changes by less than 10^{-6} and the initialization is done using 50 iterations of the power method.

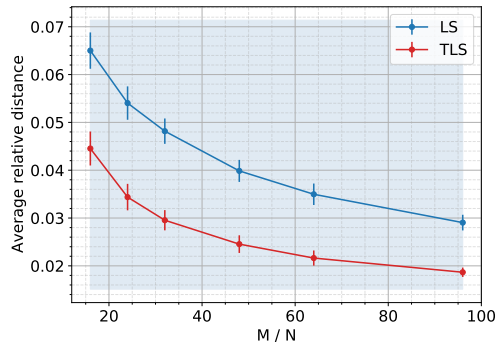
Because the output device exposure time controls the range of the measurements, the entries of the calibrated OPU transmission matrix are iid complex Gaussian and the calibrated Gaussian sensing vectors are scaled versions of the sensing vectors from the complex Gaussian measurement model. This does not impact the sensing vector updates in Section 4.2.2 because the procedure does not assume a measurement model. However, the initialization scaling and signal gradient descent updates (4.7) for both TLS and LS require minor changes. Instead of altering these steps we estimate the standard deviation and variance of the calibrated transmission matrix from its entries and divide the calibrated matrix by the estimated standard deviation. Correspondingly, we also divide the measurements by the estimated variance.



(a) Sensing vector SNR is 10 dB.
Gaussian errors.



(b) Sensing vector SNR is 30 dB.
Gaussian errors.



(c) Sensing vector SNR is 30 dB.
Handcrafted errors.

Figure B.8: Relative distance of reconstructions using TLS and LS for the CDP measurement model for different number of patterns $L = \frac{M}{N}$ when measurement SNR is 20 dB.

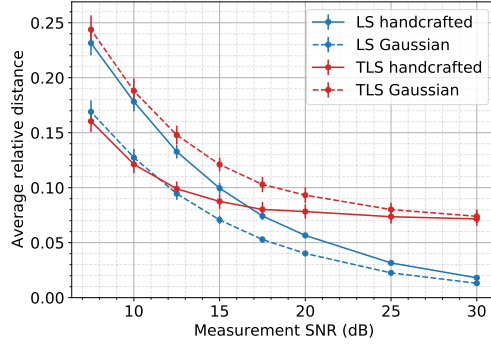


Figure B.9: Performance of TLS and LS with handcrafted errors when there is only error in measurements for the CDP measurement model. Here $L = \frac{M}{N} = 16$ octanary patterns are used.

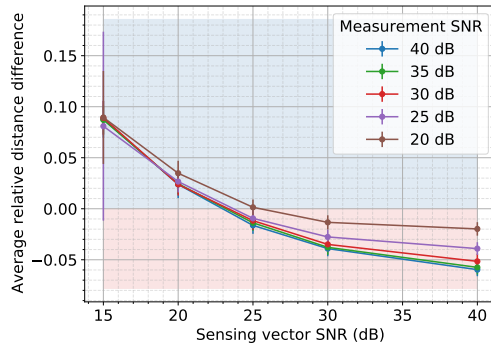


Figure B.10: Average difference in relative distance between TLS and LS solutions for the CDP measurement model when $L = \frac{M}{N} = 16$ octanary patterns are used. Different measurement and sensing vector SNR combinations are used and the errors are handcrafted. This should be compared to Figure B.7 when $L = \frac{M}{N} = 16$.

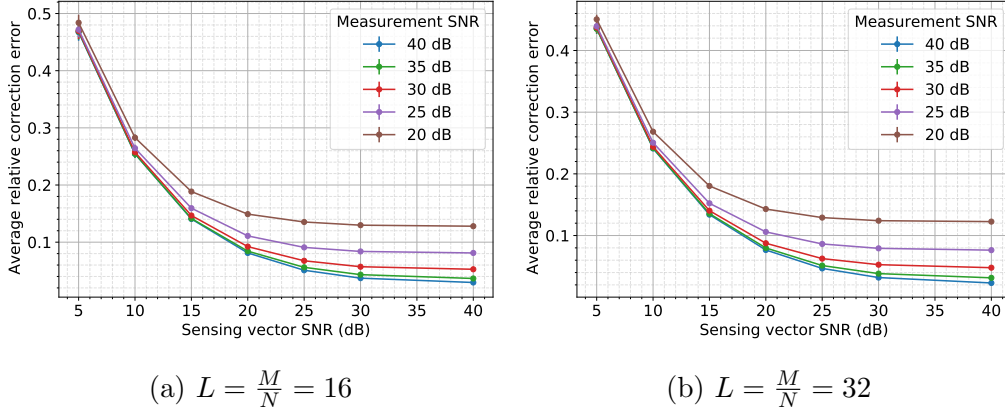


Figure B.11: Average relative sensing vector correction error when using TLS, $\text{rel.corr}(\{\tilde{\mathbf{a}}, \mathbf{x}^\#\}, \{\hat{\mathbf{a}}^\dagger, e^{j\varphi} \mathbf{x}_{\text{TLS}}^\dagger\})$, for the CDP measurement model for different measurement and sensing vector SNR combinations when the number of patterns is $L = \frac{M}{N} \in \{16, 32\}$.

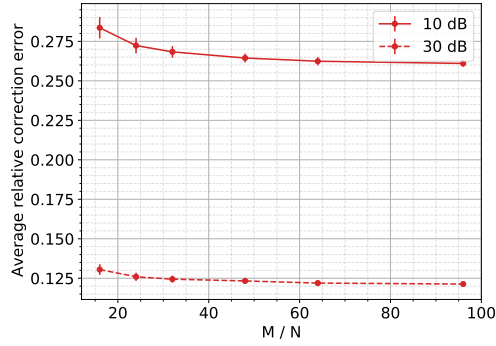


Figure B.12: Relative sensing vector correction error when using TLS for the CDP measurement model for different number of patterns $L = \frac{M}{N}$ when measurement SNR is 20 dB. The sensing vector SNR is 10 dB or 30 dB.

APPENDIX C

RANDOM MESH PROJECTORS

C.1 Oblique projection experiments

We illustrate the performance difference between linear oblique projectors and our non-linear learned operator in Figures C.1 and C.2. We refer the reader to the captions below each figure for more details.

C.2 Proof of Proposition 6

Proof. The reconstruction of the new inverse problem can be written as $\hat{\mathbf{x}} = \tilde{\mathbf{B}}\mathbf{B}^T \mathbf{x}$ where the columns of $\tilde{\mathbf{B}} = (\mathbf{B}^T)^\dagger$ form a biorthogonal basis to the columns of \mathbf{B} . Thus

$$\hat{\mathbf{x}} = \sum_{p=1}^{K\Lambda} \langle \mathbf{x}, \mathbf{b}_p \rangle \tilde{\mathbf{b}}_p. \quad (\text{C.1})$$

Using the definition of the inner product and rearranging, we get

$$\hat{\mathbf{x}}(u) = \left\langle \sum_{p=1}^{K\Lambda} \mathbf{b}_p(\cdot) \tilde{\mathbf{b}}_p(u), \mathbf{x} \right\rangle \stackrel{\text{def}}{=} \langle \boldsymbol{\kappa}(u, \cdot), \mathbf{x} \rangle \quad (\text{C.2})$$

where $\boldsymbol{\kappa}(u, v) \stackrel{\text{def}}{=} \sum_{p=1}^{K\Lambda} \mathbf{b}_p(v) \tilde{\mathbf{b}}_p(u)$. Now, the probability distribution of triangles around any point u is both shift- and rotation-invariant because a Poisson process in the plane is shift- and rotation-invariant. It follows that $\mathbb{E} \boldsymbol{\kappa}(u, v) = \tilde{\boldsymbol{\kappa}}(\|u - v\|)$ for some $\tilde{\boldsymbol{\kappa}}$, meaning that

$$(\mathbb{E} \hat{\mathbf{x}})(u) = \mathbb{E} \langle \boldsymbol{\kappa}(u, \cdot), \mathbf{x} \rangle = \langle \tilde{\boldsymbol{\kappa}}(\|u - \cdot\|), \mathbf{x} \rangle = (\mathbf{x} * \tilde{\boldsymbol{\kappa}})(u) \quad (\text{C.3})$$

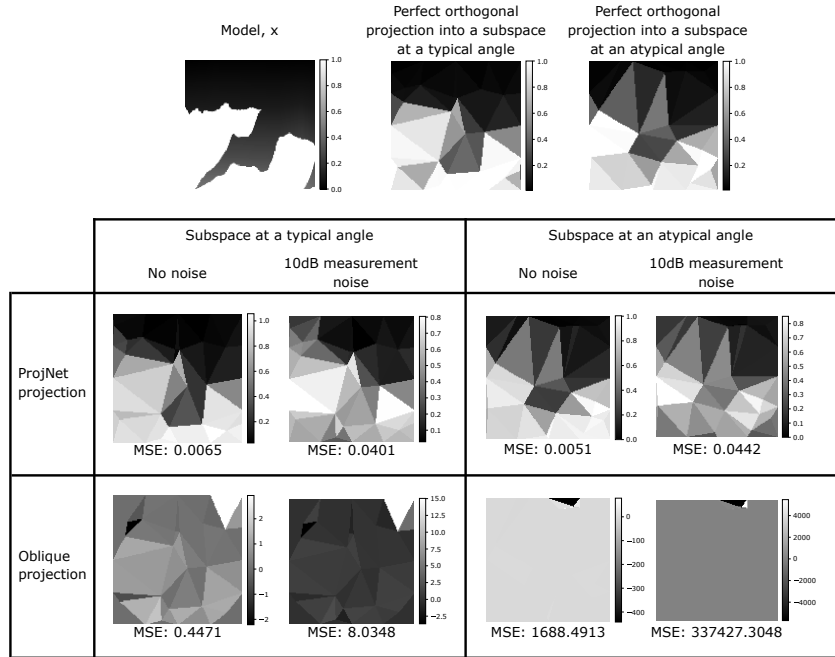


Figure C.1: Comparison between perfect orthogonal projection, ProjNet projections and oblique projection. The projections of an image, \mathbf{x} , (same as Figure 5.6) are obtained using ProjNet and the linear oblique projection method. The mean-squared errors (MSE) between the obtained projections and the perfect projections are stated. The subspaces used in this figure were used in the ProjNet reconstructions.

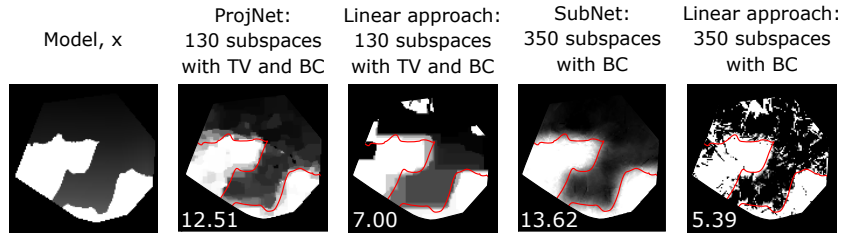


Figure C.2: We try hard to get the best reconstruction from the linear approach. SNRs are indicated in the bottom-left of each reconstruction. In the linear approach, coefficients are obtained using the linear oblique projection method. Once coefficients are obtained, they are non-linearly reconstructed according to (5.4). Both linear approach reconstructions use the box-constraint (BC) mentioned in (5.4). For the 130 subspace reconstruction, total-variation (TV) regularization is also used. Therefore, once the coefficients are obtained using the linear approach, the reconstruction of the final image is done in an identical manner as ProjNet for 130 subspaces and SubNet for 350 subspaces. To give the linear approach the best chance, we also optimized hyperparameters such as the regularization parameter to give the highest SNR.

which is a convolution of the original model with a rotationally invariant (isotropic) kernel. \square

C.3 Additional training information

C.3.1 Network architectures

Figure C.3 explains the network architecture used for ProjNet and SubNet. The network consists of a sequence of downsampling layers followed by upsampling layers, with skip connections [172, 173] between the downsampling and upsampling layers. Each ProjNet output is constrained to a single subspace by applying a subspace projection operator, \mathbf{P}_{S_λ} . We train 130 such networks and reconstruct from the projection estimate using (5.4). SubNet is a single network that is trained over multiple subspaces. To do this, we change its input to be $[\tilde{\mathbf{y}} \ \mathbf{B}_\lambda]$. Moreover, we apply the same projection operator as ProjNet to the output of the SubNet. Each SubNet is trained to give projection estimates over 350 random subspaces. This approach allows us to scale to any number of subspaces without training new networks for each. Moreover, this allows us to build an over-constrained system $\mathbf{q} = \mathbf{B}\mathbf{x}$ to solve. Even though SubNet has almost as many parameters as the direct net, reconstructing via the projection estimates allows SubNet to get higher SNR and more importantly, get better estimates of the coarse geometry than the direct inversion. All networks are trained with the Adam optimizer.

C.3.2 Shapes dataset

The shapes dataset was generated using random ellipses, circle and rectangle patches. See Figure C.4 for examples. This dataset was used in Figure 5.7.

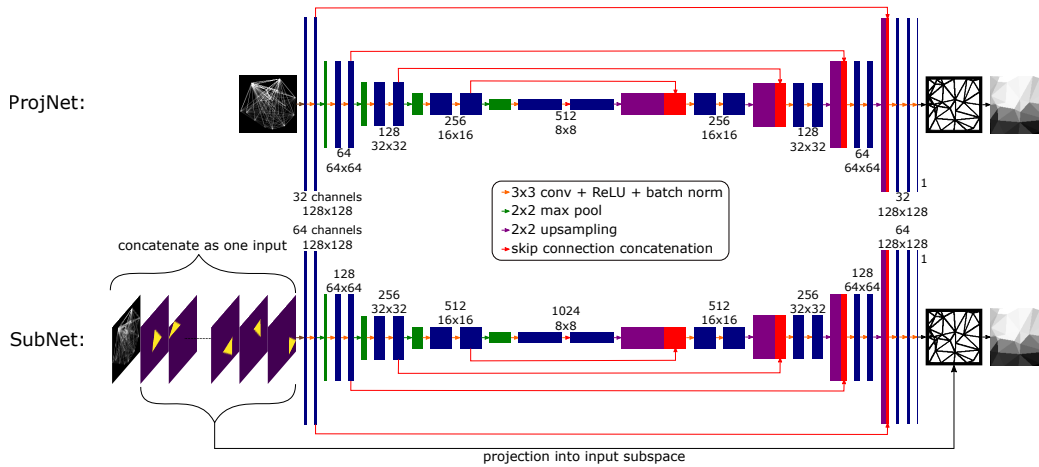


Figure C.3: a) ProjNet architecture; b) SubNet architecture. In both cases, the input is a non-negative least squares reconstruction and the network is trained to reconstruct a projection into one subspace. In SubNet, the subspace basis is concatenated to the non-negative least squares reconstruction.

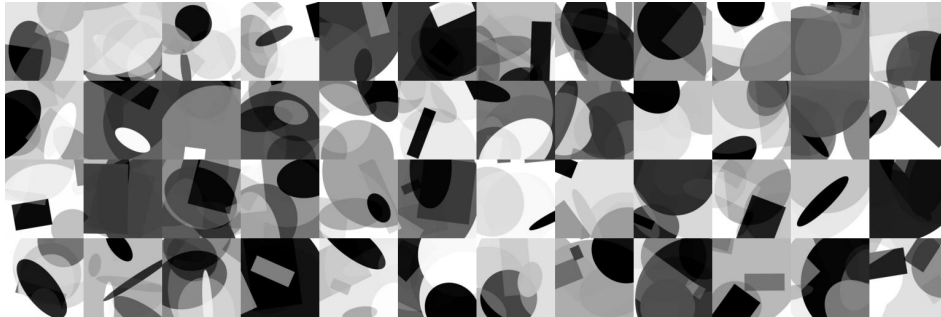


Figure C.4: Examples from the random shapes dataset which is used in Figure 5.7.

C.4 Further reconstructions

C.4.1 Geo images

We showcase more reconstructions on actual geophysics images taken from the BP2004 dataset in Figure C.5. Note that all networks were trained on the LSUN bridges dataset.

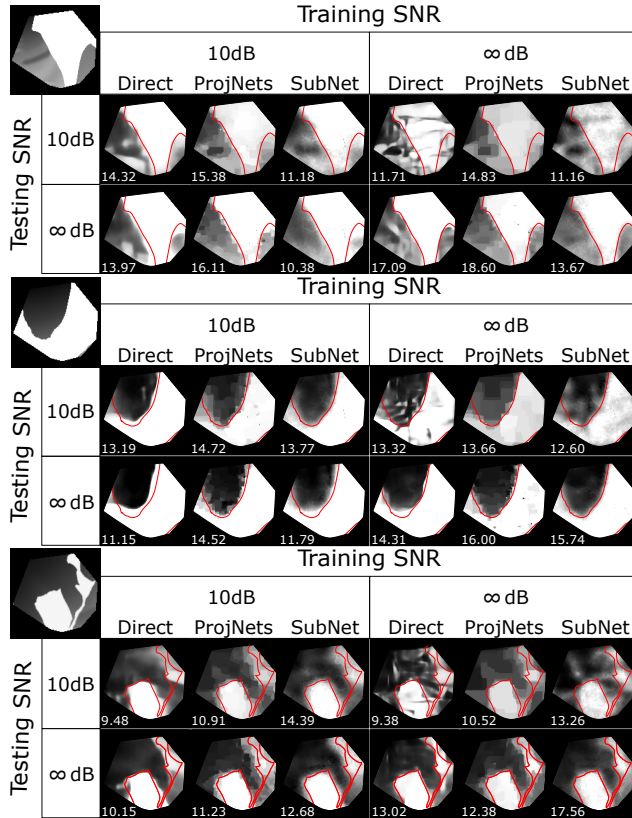


Figure C.5: Geophysics image patches taken from BP2004 dataset. Our method especially gets correct global shapes with better accuracy even when tested on noise levels different from training.

C.4.2 Erasure reconstructions

We show additional reconstructions for the largest corruption case, $p = \frac{1}{8}$, for x-ray images (Figure C.6) and geo images (Figure C.7). Our method consistently has better SNR. More importantly, we note that there is not a single instance where the direct reconstruction gets a feature that our methods do not. In a majority of instances, the direct network misses a feature of the image. This is highly undesirable in settings such as geophysical imaging.

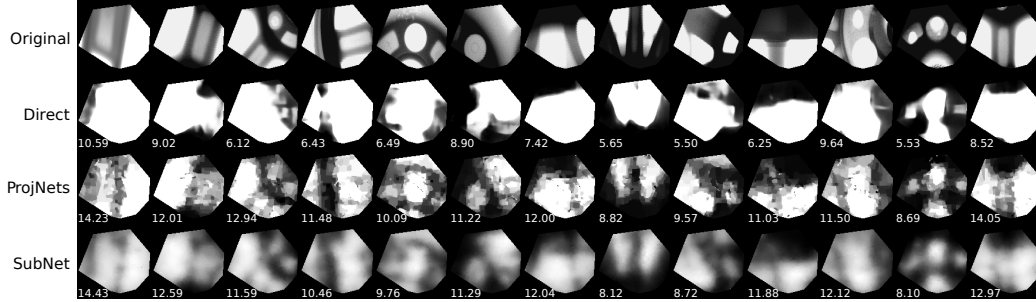


Figure C.6: Reconstructions from erasures on x-ray images with erasure probability $p = \frac{1}{8}$.

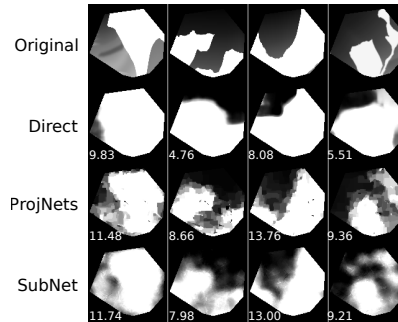


Figure C.7: Reconstructions from erasures on geo images with erasure probability $p = \frac{1}{8}$.

C.5 Comparison of proposed method with ensemble of direct nets

In Section 5.3, we train multiple ProjNets, each focusing on a different low-dimensional subspace. Here we train an ensemble of direct networks where each network is as described in Section 5.3.2 and evaluate the robustness of a method where the outputs of these networks are averaged to give a final reconstruction. Once again, we consider scenarios where the model is trained with data at a particular noise level and then tested with data at a different noise level and with erasures that were unseen during training time. We show that our proposed method is more robust to changes in the test scenario.

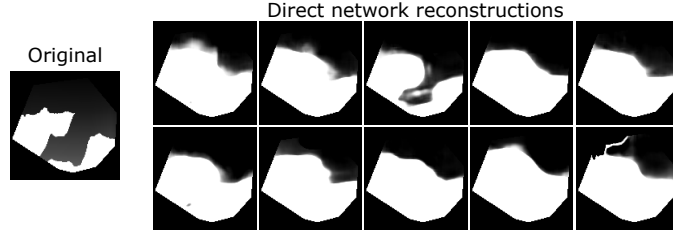


Figure C.8: Reconstructions of the original image from 10 individually trained direct inversion networks for 10dB noise under the $p = \frac{1}{8}$ erasure corruptions model (described in Figure 5.6b). 9 out of the 10 reconstructions fail to capture the key structure of the original image.

Scenario	50-Ensemble	ProjNets	SubNet
10 dB train and 10 db test	13.404	13.15	12.72
10 dB train and ∞ dB test	10.70	13.36	11.51
10 dB train and erasures with $p = \frac{1}{8}$ test	8.70	10.64	10.48

Table C.1: Comparison of SNRs in different scenarios. 50-Ensemble refers to reconstructions obtained by averaging the output of 50 direct networks.

In Figure C.8, we consider the erasure model with $p = \frac{1}{8}$ (described in Figure 5.6b). Nine out of 10 randomly chosen direct network reconstructions fail to capture the key structure of the original image under this corruption mechanism. In Table C.1, we summarize this with the SNRs of reconstructions from the erasure corruption mechanism. In that table we also report SNRs when reconstructing from measurements at different noise levels. The ensemble of direct networks performs well when the training and test data have the same measurement noise level. However, our method is more robust to changes in the test noise level. This further illustrates that direct networks are highly tuned to the training scenario and therefore not as stable as our proposed method.

APPENDIX D

DIFFERENTIABLE UNCALIBRATED IMAGING

D.1 2D CT experiments with 120 view angles

In Figure D.1, we show 2D CT imaging sample reconstructions for different combinations of measurement noise and measurement coordinate uncertainty when there are 120 view angles. The experiment is performed as described in Section 6.3.1. The reconstructions for when there are 90 view angles is shown in Figure 6.4.

Figure D.2 shows average angle error plots for 2D CT imaging when there are 120 view angles. The experimental details are in Section 6.3.1, and the equivalent results for 90 view angles are shown in Figure 6.5.

D.2 Additional spline information

We continue using 2D CT as an example and build on Section 6.2.3 to provide further information on NURBS. Besides the learnable weights and learnable control point vectors explained in Section 6.2.3, there is a knot vector for each dimension of the measurement coordinates that is not learnable. The NURBS surface also has degrees $d_\Theta, d_\mathcal{T} \in \mathbb{Z}^+$ for the view angle and detector location measurement coordinate dimensions. The knot vector for the view angle dimension has $(J + d_\Theta + 1)$ elements which are arranged in ascending order. We design its u th element, k_u , to be zero when $0 \leq u < d_\Theta + 1$, uniformly spaced between zero and one when $d_\Theta + 1 \leq u \leq J$, and one when $J < u \leq J + d_\Theta$ [147]. The knot vector for the detector location dimension has $(J + d_\mathcal{T} + 1)$ elements and is made in a similar manner by using its degree $d_\mathcal{T}$.

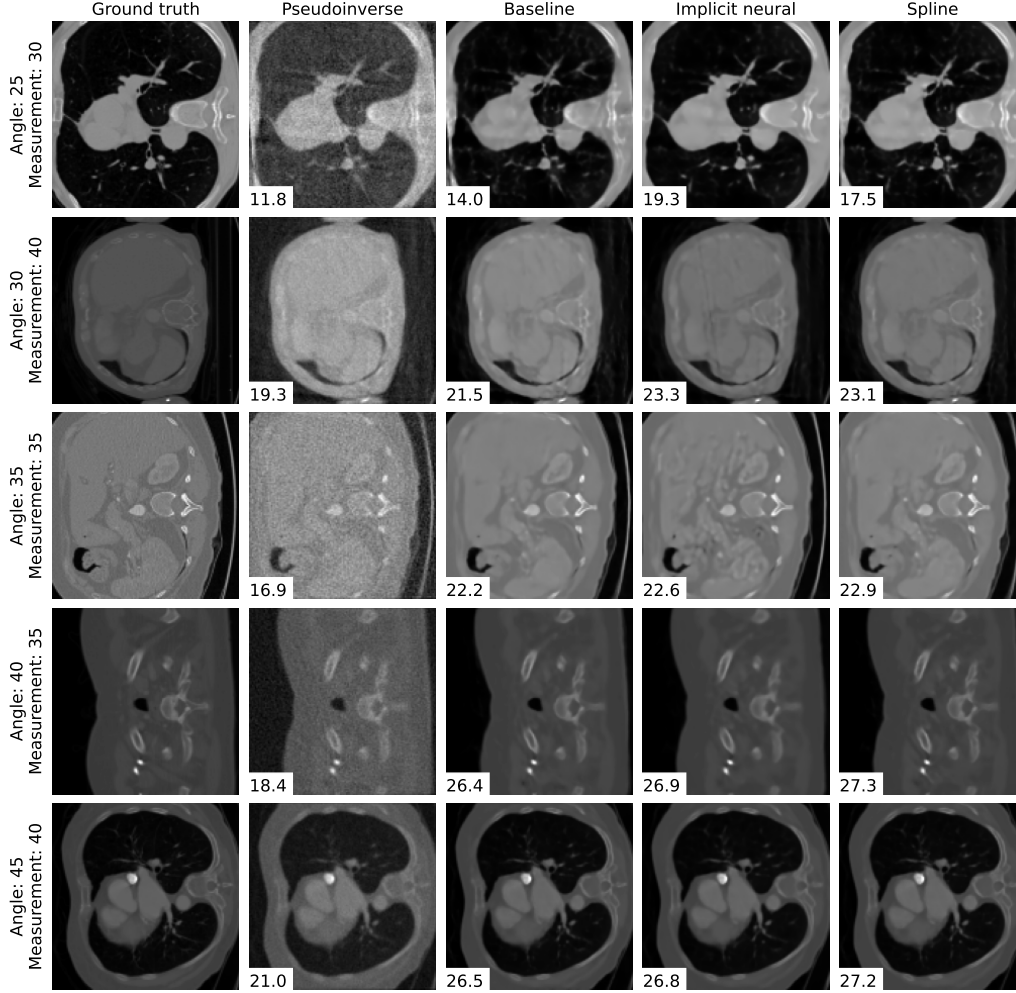


Figure D.1: Example reconstructions for different measurement noise and measurement coordinate uncertainty when there are 120 view angles. The SNRs are shown for each reconstruction

The rational basis functions defined in (6.10) are

$$b_{j,k}([\theta', t']^T) = \frac{w_{j,k} Q_{j,d_\Theta}(\theta') Q_{k,d_\mathcal{T}}(t')}{\sum_{u=0}^{J-1} \sum_{v=0}^{K-1} w_{u,v} Q_{u,d_\Theta}(\theta') Q_{v,d_\mathcal{T}}(t')}. \quad (\text{D.1})$$

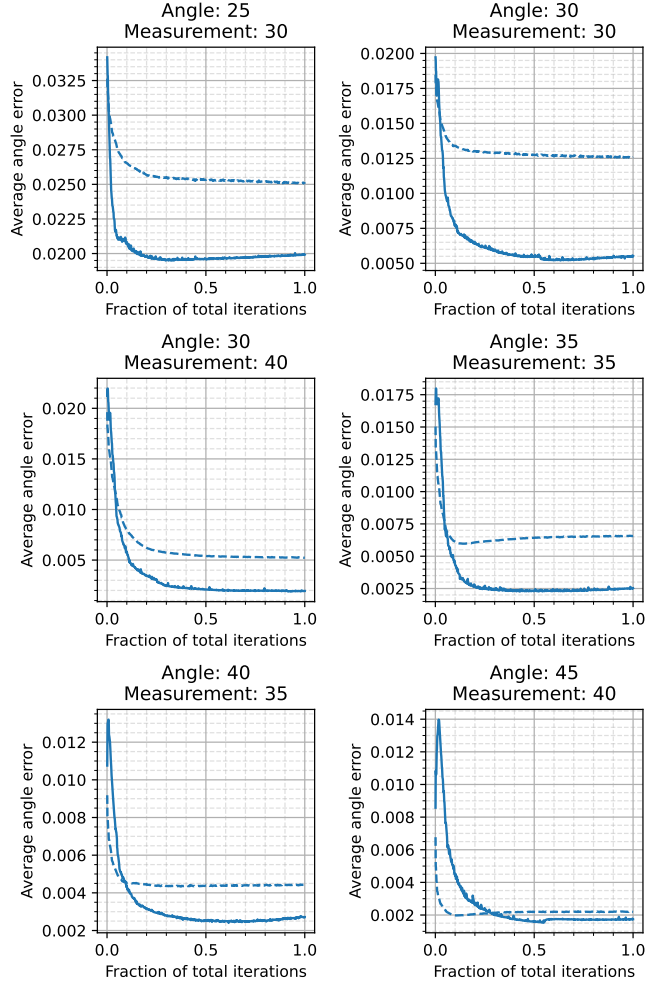


Figure D.2: Average angle error for one test image with different measurement noise and measurement coordinate uncertainty combinations when there are 120 view angles. The solid lines are for implicit neural representations and the dashed lines are for spline representations.

The function $Q_{u,d}(\cdot)$ is the u th B-spline basis function of degree d . If k_u denotes the u th element of a knot vector, each basis function can be obtained using the Cox-de Boor recursion method,

$$Q_{u,0}(k) = \begin{cases} 1 & \text{if } k_u \leq k \leq k_{u+1} \\ 0 & \text{otherwise} \end{cases} \quad (\text{D.2})$$

$$Q_{u,d}(k) = \frac{k - k_u}{k_{u+d} - k_u} Q_{u,d-1}(k) + \frac{k_{u+d+1} - k}{k_{u+d+1} - k_{u+1}} Q_{u+1,d-1}(k). \quad (\text{D.3})$$

From (D.2), we can see that a degree zero NURBS is constructed from piecewise constant basis functions. The recursion (D.3) is then used to create higher degree basis functions with larger support. If the denominator in any term of (D.3) is zero, that term is taken to be zero.

The NURBS surface for two-dimensional measurement coordinates in (6.10) is the tensor product of two one-dimensional NURBS curves as shown in (D.1). To create NURBS surfaces for D -dimensional measurement coordinates, we take the tensor product of D one-dimensional NURBS curves. This is done for $D = 3$ in Section 6.3.2.

D.3 Implementation details

D.3.1 Framework optimization

In this section, we provide implementation details for solving our optimization problem (6.8). All parameters were tuned on a held out set of images for three randomly chosen measurement and operator error combinations.

To implement NURBS, we modified and extended the PyTorch source code released by the NURBS-Diff module authors [147]. The implicit neural representation and neural network for $G_{\mu}(\cdot)$ in (6.8) are also implemented in PyTorch. This enables us to conveniently optimize the objective function (6.8) using automatic differentiation and the Adam optimizer.

When implicit neural networks are used, the optimization is run for at least 8,000 iterations and at most 20,000 iterations. The optimization is terminated when the loss value between successive iterations is below 1×10^{-10} for 2D CT imaging and 1×10^{-11} for 3D CT imaging. When splines are used the optimization runs for at least 2,000 iterations and at most 5,000 iterations. The optimization for all imaging problems is terminated when the loss value between successive iterations is below 1×10^{-11} .

2D CT imaging. When implicit neural representations are used, $\lambda_1 = 10$ and $\lambda_2 = 1$ in (6.8). The learning rate for both the neural network parameters and the input coordinates is 5×10^{-4} . When splines are used, $\lambda_1 = 1$ and $\lambda_2 = 0.025$. The learning rate for the neural network parameters is 5×10^{-2} and for the input coordinates is 2×10^{-4} .

For the implicit neural representation, we use the cosine of the view angle rather than the angle when creating the measurement coordinate. This encodes the circular nature of angular data and results in improved performance.

3D CT imaging. When implicit neural representations are used, $\lambda_1 = 10$ and $\lambda_2 = 1$. The learning rate for the neural network parameters is 1×10^{-3} and for the input coordinates is 1×10^{-4} . When splines are used, $\lambda_1 = 1$ and $\lambda_2 = 0.6$. The learning rate for the neural network parameters is 5×10^{-3} and for the input coordinates is 1×10^{-4} .

D.3.2 Reconstruction Unet training

In this section we describe the implementation details for the Unet neural networks used for the reconstruction method $G_{\mu}(\cdot)$ in (6.8).

A mean-squared error loss function is minimized using Adam during training. We train the Unets for 100 epochs where one epoch is a full pass through the training dataset.

For the 2D Unet, a batch size of 128 and learning rate of 1×10^{-3} is used. For the 3D Unet, a batch size of 16 and learning rate of 1×10^{-3} is used.

Publicly available Unet architectures were downloaded and trained. Unless mentioned here, the default parameters from the download sources were used. The 2D Unet model is from <https://github.com/mateuszbeda/brain-segmentation-pytorch> [174]. We used one input channel, one output channel, and 16 features in the first layer. The 3D Unet model is from <https://github.com/ELEKTRONN/elektronn3>. We used one input channel, one output channel, 16 features in the first layer and a depth of four blocks.

Due to the limited size of the 3D volume training dataset for 3D CT imaging, we use a data augmentation strategy. We perform random horizontal flips, random vertical flips, and random rotations by 90, 180 or 270 degrees.

D.3.3 NURBS

2D CT imaging. For 2D CT imaging, the NURBS degree along the view angle measurement coordinate dimension is 18. It is two in the detector location dimension. Furthermore, we create additional control points to ensure the spline measurement representations satisfy the Radon transform measurement consistency conditions [175]

$$r_{\varphi}([\theta + \pi, t]^T) = r_{\varphi}([\theta, -t]^T), \quad (\text{D.4})$$

and

$$r_{\varphi}([\theta - \pi, t]^T) = r_{\varphi}([\theta, -t]^T), \quad (\text{D.5})$$

where a detector location of $-t$ refers to the t th last detector. This gives the locally supported NURBS basis functions (D.1) a sufficient number of control points around 0 and π radians and ensures the NURBS is accurate in the interval $[0, \pi]$ radians. Specifically, if d_{Θ} is the degree of the spline in the view angle dimension, we use the consistency condition to create new control points for (6.10)

$$\mathbf{p}_{j+J-1,k} = \left[\theta_j + \pi, t_k, \mathbf{y}([\tilde{\theta}_j, -\tilde{t}_k]^T) \right]^T \quad (\text{D.6})$$

for $1 \leq j \leq d_{\Theta}$ and

$$\mathbf{p}_{j-J-1,k} = \left[\theta_j - \pi, t_k, \mathbf{y}([\tilde{\theta}_j, -\tilde{t}_k]^T) \right]^T \quad (\text{D.7})$$

for $J - d_{\Theta} < j \leq J$ where J is the number of view angles in the observed measurements. Essentially, if there are K detectors, we create $2d_{\Theta}K$ additional control points.

3D CT imaging. For 3D CT, the NURBS degree along the tilt angle measurement coordinate dimension is 15. It is two in the two detector location dimensions. Additionally, for this imaging technique we fix the basis function weights to one and do not optimize them. This makes the NURBS surface a B-spline surface.