Foundations of Model-Based Deep Learning: Applications, Interpretability and Performance Guarantees

BY

SHAHIN KHOBAHI
B.Sc., Isfahan University of Technology, 2012
M.Sc., University of Illinois at Chicago, 2021

THESIS

Submitted as 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 at Chicago, 2022

Chicago, Illinois

Defense Committee:

Mojtaba Soltanalian, Chair and Advisor
Natasha Devroye
Ahmet Enis Cetin
Rashid Ansari
Arnold Lee Swindlehurst, University of California, Irvine

ACKNOWLEDGMENTS

This thesis would not have been possible without the close collaboration I have had with my advisor Prof. Mojtaba Soltanalian over the past few years. Prof. Soltanalian's creativity and passion have profoundly shaped the way I think about scientific problems and pursue research directions. Among the numerious skills I started to pick up from Prof. Soltanalian, my favorite one is asking the right question and formalizing it. I will not forget our countless long meetings that did not seem long to me at all. Looking back through all these years, I strongly believe that Prof. Soltanalian is the best mentor I could have ever hoped for. I am incredibly grateful for the tremendous amount of time and energy Prof. Soltanalian has put into shaping the direction of my academic direction and for molding me into the person I am today. Since the beginning of my studies, Prof. Soltanalian has provided his full academic and emotional support for more than almost six years through the tough path towards this thesis. I am forever grateful for all I have learned from him.

I wish to express my deepest gratitude to all of my committee members: Prof. Natasha Devroye, Prof. Ahmet E. Çetin, Prof. Rashid Ansari, and Prof. Arnold Lee Swindlehurst, for their insightful comments and encouragements on this thesis.

I would like to sincerely thank all my collaborators and coauthors including Prof. Yonina C. Eldar (Weizmann Institute of Science), Prof. Nir Shlezinger (Ben-Gurion University of the Negev), Prof. A. Lee Swindlehrst (University of California at Irvine), Prof. Dan Schonfeld (University of Illinois at Chicago), Dr. Feng Jiang (University of California at Irvine), Dr.

Chirag Agarwal (Harvard University), Arindam Bose (University of Illinois at Chicago), and Naveed Naimipour (University of Illinois at Chicago). I enjoyed collaborating with these smart people.

My deepest gratitude goes to my parents and my brother who have shaped who I am today. Words cannot capture how grateful I am to my mother, Dr. Elham Payervand, without whom everything in my life would have been drastically different.

Finally, I am dedicating this thesis to my mother and the memory of my father, Dr. Khosrow Khobahi, who raised me to be an achiever, to push myself to the limits, and be dedicated to finish the job through determination and hard work.

Finally, but not least, I am dedicating this thesis to my mother and my wife Shakiba Ghanemzadeh for their sacrifices and unconditional support.

SK

PREFACE

This dissertation is an original intellectual product of the author Shahin Khobahi. The research presented here has been partially supported by the National Science Foundation (NSF) under grants CCF-1704401 and ECCS-1809225. Parts of the thesis are also partially supported by the Illinois Discovery Partners Institute (DPI) Seed Award. The results of these research works have previously appeared (or are appearing) as journal articles and conference presentations. The copyright permissions for reusing the published materials have been presented in Appendix.

Shahin Khobahi
July 22, 2022

CONTRIBUTION OF AUTHORS

In the work of *LoRD-Net: Unfolded Deep Detection Network With Low-Resolution Receivers* and *Model-Inspired Deep Detection with Low-Resolution Receivers*, Shahin Khobahi developed the theory, formulated the problems and carried out the experiments. The manuscript was written with the inputs from Mojtaba Soltanalian, Nir Shlezinger and Yonina C. Eldar.

In the work of *Deep Signal Recovery with One-bit Quantization*, Shahin Khobahi developed the model, and formulated the problems and carried out the experiments. The manuscript was written with the inputs from Mojtaba Soltanalian and Yonina C. Eldar. Naveed Naimipour provided some inputs on the flow of information throughout the paper.

In the work of *Model-Based Deep Learning for One-Bit Compressive Sensing*, Shahin Khobahi developed the model and theory used in the paper and carried out all the experiments. Mojtaba Soltanalian provided insights on the theory and suitable metrics for the experiments. The manuscript were written with the inputs from Mojtaba Soltanalian.

In the work of *Deep One-Bit Compressive Autoencoding*, Shahin Khobahi developed the model and theory used in the paper and carried out all the experiments with some help from Arindam Bose. The manuscript was written with the inputs from Mojtaba Soltanalian.

In the work of *Unfolded Algorithms for Deep Phase Retrieval* and *UPR: A Model-Driven Architecture for Deep Phase Retrieval*, Shahin Khobahi developed the theory and the model used in the work, formulated the problem, and carried out the experiments. The manuscript

CONTRIBUTION OF AUTHORS (Continued)

was written with the inputs from Mojtaba Soltanalian. Naveed Naimipour provided some inputs on the experiments, and helped with revising the body of the manuscript.

In the work of *Deep Radar Waveform Design for Efficient Automotive Radar Sensing*, Shahin Khobahi developed the problem formulation and system model. Shahin Khobahi and Arindam Bose carried out the experiments. Shahin Khobahi and Arindam Bose wrote the manuscript with the inputs from Mojtaba Soltanalian.

In the work of *DEEP-URL: A Model-Aware Approach to Blind Deconvolution Based on Deep Unfolded Richardson-Lucy Network*, Shahin Khobahi provided the problem formulation and developed the model used in the work. Shahin Khobahi, Arindam Bose and Chirag Agarwal carried out the experiments. Shahin Khobahi, Arindam Bose, and Chirag Agarwal collaboratively wrote the manuscript with inputs from Mojtaba Soltanalian and Dan Schonfeld.

In the work of *Guaranteed Deep Learning for Reliable Radar Signal Processing*, Shahin Khobahi formulated the problem and developed the theory and mathematical foundations in the work, and carried out all the experiments in the work. Shahin Khobahi wrote the manuscript was written with inputs from Mojtaba Soltanaliam, Mohammad Emadi, Ali Mostajeran, and Pu (Perry) Wang.

TABLE OF CONTENTS

CHAPTER PAGE

TABLE OF CONTENTS (Continued)

CHAPTER **PAGE**

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES (Continued)

FIGURE **PAGE**

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

FIGURE **PAGE**

NOTATIONS

Bold lowercase letters are used to denote the vectors and bold uppercase letters for matrices. The following mathematical notations are used throughout this thesis:

$\lvert x \rvert$	the absolute value of a scalar x
$[x]$	the integral part of a real scalar x, *i.e.*, the greatest integer $\leq x$
$\{x\}$	the fractional part of a real scalar x, *i.e.*, $\{x\} = x - [x]$
$\boldsymbol{x}_m(k)$	the k^{th} element of vector $\boldsymbol{x}_m$
$[\boldsymbol{X}]_{i,j}$	the $(i,j)^{\text{th}}$ element of matrix $\boldsymbol{X}$
$\lVert \boldsymbol{x} \rVert_p$	the l_p-norm of $\boldsymbol{x}$, defined as $(\sum_k \lvert \boldsymbol{x}(k) \rvert^p)^{\frac{1}{p}}$
$\lVert \boldsymbol{x} \rVert$	the l_2-norm of $\boldsymbol{x}$
$\boldsymbol{x} \circledast \boldsymbol{y}$	denotes convolution of $\boldsymbol{x}$ and $\boldsymbol{y}$
$\lVert \boldsymbol{X} \rVert_F$	the Frobenius norm of matrix $\boldsymbol{X}$ defined as $\sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} \lvert [\boldsymbol{X}]_{i,j} \rvert^2}$
$\boldsymbol{X}^*$	the complex conjugate of the matrix $\boldsymbol{X}$
$\boldsymbol{X}^T$	the transpose of the matrix $\boldsymbol{X}$
$\boldsymbol{X}^H$	the complex conjugate transpose of the matrix $\boldsymbol{X}$
$\boldsymbol{X}^\dagger$	the Moore-Penrose pseudoinverse of the matrix $\boldsymbol{X}$
$\text{Tr}\,(\boldsymbol{X})$	the trace of matrix $\boldsymbol{X}$
$\text{vec}\,(\boldsymbol{X})$	the vector obtained by column-wise stacking of matrix $\boldsymbol{X}$

NOTATIONS (Continued)

$\mathrm{diag}\,(\boldsymbol{X})$	denotes a vector formed by diagonal entries of the matrix $\boldsymbol{X}$
$\mathrm{Diag}\,(\boldsymbol{x})$	denotes a diagonal matrix formed by the entries of the vector $\boldsymbol{x}$
$\arg(\boldsymbol{X})$	the phase angle (in radians) of $\boldsymbol{X}$
$\mathrm{cov}\,(\boldsymbol{X})$	the covariance matrix of $\boldsymbol{X}$
$\Re(\boldsymbol{X})$	the real part of $\boldsymbol{X}$
$\Im(\boldsymbol{X})$	the imaginary part of $\boldsymbol{X}$
$\sigma_n(\boldsymbol{X})$	the n^{th} maximal eigenvalue of $\boldsymbol{X}$
$\lambda_n(\boldsymbol{X})$	the n^{th} maximal singular value of $\boldsymbol{X}$
$\boldsymbol{X} \otimes \boldsymbol{Y}$	the Kronecker product of two matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$
$\boldsymbol{X} \odot \boldsymbol{Y}$	the Hadamard product of two matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$
$\boldsymbol{X} \succ \boldsymbol{Y}$	$\boldsymbol{X} - \boldsymbol{Y}$ is positive definite
$\boldsymbol{X} \succeq \boldsymbol{Y}$	$\boldsymbol{X} - \boldsymbol{Y}$ is positive semidefinite
$\boldsymbol{I}_n$	the identity matrix of dimension n
$\mathbf{1}_n$	the all-one vector of size $n \times 1$
$\mathbf{0}_n$	the all-zero vector of size $n \times 1$
$\boldsymbol{O}$	the matrix with all elements as zero
$\boldsymbol{e}_n$	the n^{th} standard basis of $\mathbb{C}^n$ or n^{th} column of an identity matrix
$\mathbb{R}$	the set of real numbers

NOTATIONS (Continued)

$\mathbb{R}_+$	the sets of real non-negative numbers
$\mathbb{C}$	the set of complex numbers
$\mathbb{N}$	the set of natural numbers
$\mathbb{Z}$	the set of integers
$\mathbb{B}_N^M$	the set of binary vectors with size M and N non-zero elements, $N \leq M$
$\mathcal{S}^M$	the set of all real symmetric matrices of size $M \times M$
$\mathbf{F}_n$	the n dimensional discrete Fourier transform matrix
$\mathbb{E}\{\cdot\}$	the mathematical expectation of a random variable
$\mathbb{P}\mathrm{r}\{\cdot\}$	denotes the probability of a random event
$\mathrm{sign}(\cdot)$	the element-wise signum operator
$\mathrm{csign}(\cdot)$	the element-wise complex signum operator as $\mathrm{sign}(\Re\{\cdot\}) + j\mathrm{sign}(\Im\{\cdot\})$
$\mathcal{N}(\cdot,\cdot)$	the normal distribution with mean, and covariance as first and second arguments, respectively
$\ln a$	natural logarithm of a, equivalent to $\log_e a$
j	the imaginary unit *i.e.*, $j = \sqrt{-1}$

SUMMARY

The first part of this thesis presents the foundations of model-based deep learning with applications in various signal processing fields such as communication systems, compressive sensing, phase retrieval, radar signal processing, and image processing. The remaining part of the thesis is devoted to theoretical analysis of the proposed ideas, and specifically, presenting performance guarantees for the obtained model-based deep architectures.

CHAPTER 1

INTRODUCTION

Signal processing paradigm is dominated by techniques that heavily rely on the underlying statistical and mathematical model of signals and systems. Accordingly, these parametrized mathematical models play a central role in understanding and design of complex information systems that are becoming common in our age. However, they often cannot take into account the intricate interactions innate to such systems. In particular, to obtain an optimal operation regime for such model-based signal processing techniques, one usually requires the exact knowledge of the system parameters and the analytical mathematical model under which the system is operating. However, the majority of model-based algorithms usually fail in scenarios where the assumed mathematical model does not truly reflect the underlying physical model of the system. Furthermore, the parameters of the underlying model might not be fully known at the time of processing, which in turn results in a severe degradation in the performance. On the contrary, purely data-driven approaches do not need explicit mathematical models for data generation and have a wider applicability at the cost of interpretability. Although the data-driven approaches can handle large and complex datasets, they are ignorant of the underlying mathematical model of the systems generating them. Thus, it is vital to develop *hybrid* data-driven and model-based frameworks to enhance the accuracy, computational complexity, and efficiency of the data acquisition model in complex, large-scale scenarios. In this thesis, we aim to take the well-established iterative approaches, specifically developed for signal processing and inference,

and use them as a baseline to design deep architectures where each layer is specifically tailored to resemble an iteration of a well-thought traditional signal processing, inference, or optimization algorithm. Data-driven machine learning methods (particularly deep learning) will be used to boost the performance of the underlying inference optimization algorithm in terms of speed of convergence and effectiveness. Such hybrid models differ from the existing deep learning based methods in that they lead to novel model-based *deep architectures* specifically designed to resemble the iterations of a well-established optimization algorithm, and that they incorporate both parameterized and non-parameterized mathematical models for complex systems. Note that *since the emerging deep architectures are sparser due to incorporation of problem-level reasoning, the training of such architectures will be less data hungry and much faster than general "bulky" networks—thus, paving the way for real-time or scalable machine learning.*

This thesis builds upon a new paradigm based on optimization theory that allows for the incorporation of domain knowledge into the existing deep learning models.

1.0.1 Preliminaries

In this thesis, we take advantage of the following concepts:

1. Statistical and machine learning models [4–6].
2. Deep learning and automated differentiation techniques [7–10].
3. Optimization theory and variational analysis and their applications in signal processing [11–15].
4. Control theory, dynamical systems analysis via integral quadratic constraints, and reverse optimization modeling [16–18].

We assume the reader has some familiarity with the above topics.

1.0.2 Background

The field of signal processing is traditionally dominated by classical statistical inference methodologies. In particular, classical methodologies are model-based algorithms which heavily depend on having an exact knowledge of the underlying mathematical model that describes the interaction between various variables existing in the underlying physical system. Accordingly, classical model-based signal processing techniques usually depend on simplified statistical modeling of the actual physical system to make the mathematical model more tractable and interpretable. Nevertheless, inaccurate modeling and knowledge of the true underlying system usually leads to a severe degradation of the performance of these model-based techniques. In addition, as the underlying system becomes more complex, obtaining an accurate dynamical representation becomes intractable, which in turn, may result in a total failure of these models. On the other hand, due to the incorporation of domain knowledge in the classical statistical inference techniques, these models benefit from interpretability (i.e., they are trustable), having a low number of parameters, and efficiency in computation.

In parallel to the model-based signal processing methods, data-driven approaches are becoming increasingly popular among practitioners. Over the past several years, data-driven methods, and specifically deep learning techniques, have attracted unprecedented attention from research communities, across the board. The advent of low-cost specialized powerful computing resources (e.g., GPUs, and more recently TPUs) and the continually increasing amount of massive data generated by the human population and machines, in conjunction with the new optimization

and learning methods, have paved the way for DNNs and machine learning-based models to prove their effectiveness in many engineering areas. In particular, convolutional neural networks (CNNs) as a specific architecture for DNNs have proven to be a powerful tool for signal processing due to their favorable properties (e.g., being highly parallelizable algorithms). Generic DNNs are constructed in a fashion that inference is straightforward where the output of the network is obtained via consecutive matrix multiplication resulting in a fixed computational complexity inference model. The main advantage of the deep learning-based approach is that it employs several non-linear transformations to obtain an abstract representation of the underlying data. Data-driven approaches, on the other hand, lack the interpretability and trustability that comes with model-based signal processing. They are particularly prone to be questioned further, or at least not fully trusted by the users, especially in critical applications. Furthermore, the deterministic deep architectures are generic and it is unclear how to incorporate the existing knowledge on the problem in the processing stage. *The advantages associated with both model-based and data-driven methods show the need for developing frameworks that bridge the gap between the two approaches.*

In short, the significant success of deep learning models in areas such as natural language processing (NLP) [19], life sciences [20], computer vision (CV) [21], and collaborative learning [22], among many others, have surged a significant interest in employing deep learning models for radar signal processing applications. To account for difficulties in the underlying signal processing tasks, most existing deep learning approaches however resort to very large networks whose number of parameters are in the order of millions and billions [23]—making such models data

and computing power hungry. Furthermore, these bulky deep learning models further introduce non-ignorable latency during inference which hinders in-time decision making by autonomous agents. More importantly, with all their repertoire of success, the existing data-driven tools typically lack the interpretability and trustability that comes with model-based signal processing. They are particularly prone to be questioned further, or at least not fully trusted by the users, especially in critical applications such as autonomous vehicles. Last but not least, the deterministic deep architectures are generic and it is unclear how to incorporate the existing domain knowledge on the problem in the processing stage. In contrast, many signal processing algorithms (e.g., in the fields of information theory, wireless communications, and radar signal processing) are backed by decades of theoretical development and research resulting in accurate, meaningful and reliable models. Owing to their theoretical foundations, the model-based signal processing algorithms thus usually come with performance guarantees and bounds allowing for interpreting the output of the model and certifying the achievable performance required for the underlying task.

In light of the above, it has been long apparent that a mere adoption or modification of generic deep neural networks designed for applications such as NLP and CV, and ignoring years of theoretical developments mentioned earlier will result in inefficient networks. This is even more pronounced in the long-standing radar problems with a rich literature. Hence, it is to our belief that one needs to re-think the architectural design of deep neural models than re-purposing them for adoption in critical fields such as radar signal processing for autonomous vehicles.

This thesis provides a novel methodology for bridging the gap between data-driven and model-based approaches and move toward a more mature and specialized deep learning model for signal processing. In particular, we elaborate on a framework that aims to take the well-established iterative approaches, specifically developed for signal processing and inference, and use them as a baseline to design deep architectures where each layer is specifically tailored to resemble an iteration of a well-thought traditional signal processing/inference (optimization) algorithm. Data-driven machine learning methods (particularly deep learning) will be used to boost the performance of the underlying inference optimization algorithm in terms of speed of convergence and effectiveness. Such hybrid models differ from the existing deep learning-based methods in that they lead to novel model-based *deep architectures* with novel activation functions specifically designed to resemble the iterations of a well-established optimization algorithm, and that they incorporate both parameterized and non-parameterized mathematical models for complex systems. Note that *since the emerging deep architectures are sparser due to incorporation of problem-level reasoning, the training of such architectures will be less data hungry and much faster than general "bulky" networks—thus, paving the way for real-time machine learning and non-convex optimization.*

In the following chapters, we lay the groundwork for our vision in developing interpretable, trustable, and model-driven neural networks, starting from its theoretical foundations, to advance the state-of-the-art in statistical signal processing, with applications in communication systems, compressive sensing, phase retrieval, radar signal processing, optimization theory, and autonomous vehicles; among many others. In contrast to generic deep neural networks, which

cannot provide performance guarantees due to their black-box nature, our proposed network allows for performing a mathematical analysis of the worst-case performance bound of the model not only during the training of the network but also once the network is trained and is to be used for inference purposes. Last but not least, due to the incorporation of domain-knowledge in the design of the network, the total number of parameters of the network are in the order of the signal dimension and the training can be performed very quickly (allowing for on-the-fly training) with far less data samples—making the proposed approach highly favorable for real-time applications.

1.1 Model-Based versus Data-Driven Inference Methodologies

In this part, we give a high-level comparison between model-based and data-driven inference strategies upon which we introduce and motivate the concept of model-based deep learning.

An *inference* method can be interpreted as a framework to establish reasoning based on the available information (i.e., evidence). Although this is a generic definition, in the rest of this thesis we mainly focus on signal processing modules (systems) that are able to make prediction based on a set of measured variables. In particular, an inference task is concerned with mapping an input $\boldsymbol{x} \in \mathcal{X}$ to the corresponding output variable $\boldsymbol{y} \in \mathcal{Y}$, where $\mathcal{X}$ and $\mathcal{Y}$ represents the input and output space, respectively. More concretely, an inference module can be viewed as the function:

$$f : \mathcal{X} \mapsto \mathcal{Y}, \tag{1.1}$$

where the accuracy of such inference rule f is usually quantified by defining a proper energy function $\mathcal{E}(f)$.

For a given inference problem, the main difference between model-based and data-driven approaches lies in the definition of the inference rule, i.e., the mathematical characterization of $f(\cdot)$, while both approaches share the common goal of characterizing the inference rule such that the resulting model minimizes the energy function $\mathcal{E}(f)$.

Model-based algorithms are usually hand-crafted models that incorporate the *domain knowledge* into the design of the inference function f as well as the energy function. Domain knowledge usually encompasses the knowledge of the underlying analytical expression of the system, the statistical relationship between input $\boldsymbol{x}$ and output $\boldsymbol{y}$, and in general, it broadly refers to any existing knowledge on the interaction of the variables in play. Accordingly, a model-based inference technique utilizes these existing domain knowledge to characterize the inference rule f in accordance with the existing information. Most signal processing, inference, and signal design problems can be formulated and viewed as optimization problems, which then, allows for obtaining inference rules f that minimizes the underlying energy function, in an iterative manner via domain-specific mathematical manipulation (computation). As discussed above, domain-knowledge is the main ingredient in devising model-based inference algorithms and tuning such modules is usually concerned with solving an optimization problem. As a case in point, let us consider the design of an inference module in a statistical learning framework. Let $\boldsymbol{x} \in \mathcal{X}$ and $\boldsymbol{y} \in \mathcal{Y}$ represent the input and output random variable, where our goal is to learn an inference module that maps the input to the corresponding output. In such a case, suppose that the

statistical relationship between any input-output pair is known *a priori* and can be *modeled* as a log-concave density function $p(\boldsymbol{y}|\boldsymbol{x};\theta)$, where the parameter θ denotes a particular characterization of the density function. In this case, the domain knowledge is referred to the case where both the mathematical expression of the density function $p(\boldsymbol{y}|\boldsymbol{x};\theta)$ and its parameter θ is known. Furthermore, let

$$\hat{\boldsymbol{y}} = f(\boldsymbol{x};\theta), \tag{1.2}$$

represent the inference rule for a given parameter θ. Accordingly, a model-based inference strategy can be defined as:

$$f(\boldsymbol{x};\theta) = \underset{\boldsymbol{y}\in\mathcal{Y}}{\operatorname{argmin}} \ \mathcal{E}(\boldsymbol{x},\boldsymbol{y};\theta), \tag{1.3}$$

where $\mathcal{E}(\boldsymbol{x},\boldsymbol{y};\theta)$ is a *domain-knowledge driven energy function.* In particular, let

$$\mathcal{E}(\boldsymbol{x},\boldsymbol{y};\theta) \triangleq -\log p(\boldsymbol{y}|\boldsymbol{x};\theta). \tag{1.4}$$

Then, the inference rule

$$f(\boldsymbol{x};\theta) = \underset{\boldsymbol{y}\in\mathcal{Y}}{\operatorname{argmin}} \ -\log p(\boldsymbol{y}|\boldsymbol{x};\theta) \tag{1.5}$$

coincides with that of a maximum a posteriori (MAP) inference strategy.

The above example clearly demonstrates the rationale behind the term *model-based*. In particular, the following assumptions can be drawn for model-based algorithms:

1. They do not require data to learn the inference rule.
2. They are fully interpretable and are in accordance with the underlying statistical and physical model of the underlying parameters.
3. They require the exact knowledge of the statistical model, system parameters, and the analytical relationship of the variables in play.

Nonetheless, model-based inference strategies often experience a severe degradation in performance if an accurate knowledge of the system parameters (e.g., the parameter θ) and statistical model (e.g., $p(\boldsymbol{y}|\boldsymbol{x};\theta)$) is not available. In such cases, the available data is typically utilized to estimate the system parameters to improve the accuracy performance. In fact, in many practical applications one can rarely assume that an accurate knowledge of the statistical model of the variables is available, and furthermore, we are required to impose simplified assumptions on the model which do not truly capture the true statistical model relating the input-output variables. Hence, although model-based inference algorithms are extremely valuable in theory, they have limited applicability in scenarios where the model is either too complex to be mathematically characterized, or the system parameters are difficult to be estimated.

In contrast to model-based signal processing methodologies, data-driven approaches are model-agnostic and are able to learn the intricate statistical relationship between the input and output spaces by exploiting the data. In these models, the data is typically defined by a

set $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{B}$ which is comprised of B input-output pairs. Then, the main idea behind data-driven approaches is to consider a highly expressive and parameterized class of inference functions f_γ to characterize the inference rule. Note that this is contrast to the model-based inference methodology described previously, where the mathematical characterization of the inference function is usually guided by the domain knowledge.

Deep learning-based methodologies are a particular class of data-driven approaches that charachterizes the inference rule f_γ as deep neural network architecture where the set of parameters γ defines the weights of each layer of the DNN. It can be proven that DNNs are universal approximators and a proper optimization of the weights γ allows for approximating any Borel measurable mapping. A conventional strategy for learning a DNN-based inference rule $\hat{\boldsymbol{y}} = f_{\gamma^\star}(\boldsymbol{x})$ in a data-driven manner can thus be expressed as:

$$f_{\gamma^\star}(\boldsymbol{x}) = \underset{\gamma}{\operatorname{argmin}} \;\; \mathcal{E}(\boldsymbol{x}, \boldsymbol{y}; \gamma) \equiv \frac{1}{B} \sum_{i=1}^{B} \|f_\gamma(\boldsymbol{x}_i) - \boldsymbol{y}_i\|_2^2. \tag{1.6}$$

We note that a data-driven methodology is drastically different than that of a model-based technique in that data-driven techniques are highly generic and are applicable to a wide range of applications. This is in contrast to model-based approaches in which an inference rule is learning through a highly-tailored and structured manner specific to the problem at hand. Nonetheless, despite the highly expressive power of data-driven approaches, there exist several drawback associated with these models as enumerated below:

1. Due to the highly parameterization of deep learning models, one usually requires a massive amount of data to tune the inference rule.
2. Data-driven models, and specifically, deep learning based strategies are highly generic and black-box models, and their applicability is only justifiable in scenarios where an analytical model does not exist for the system
3. Due to the black-box nature of data-driven models, there exist no possible strategy to incorporate the existing domain knowledge into the model.
4. The black-box and generic nature of deep learning and data-driven models renders the overall inference rule to lack interpretability which makes it difficult to analyze the behaviour of the model and provide performance guarantees of the optimality of the model.

Evidently, there exist an untapped potential in bridging the gap between the model-based and data-driven approaches to obtain a *hybrid* model-based and data-driven framework that allows us to benefit from best of the both worlds. This thesis aims to address the drawbacks of both models by establishing the foundations of model-based deep learning for signal processing.

1.2 Foundations of Model-Based Deep Learning: A High-Level Overview

In this part, we provide the main ideas behind developing model-based deep learning methodologies for the field of signal processing. We commence by drawing the connection between deep neural networks and optimization theory, and proceed by introducing the main idea behind model-based deep learning.

We denote a deep neural network with l layers as the composition of l *differentiable* functions, given by:

$$\hat{\boldsymbol{y}} = f_\theta(\boldsymbol{x}) = g_{\theta_l} \circ \cdots \circ g_{\theta_1}(\boldsymbol{x}), \tag{1.7}$$

where $\theta = \cup_{i=1}^{l} \theta_i$ denotes the overall set of parameters of the DNN, g_{θ_i} represents a *differentiable* function whose parameters is given by the set θ_i, and the overall architecture can be viewed as a mapping belonging to the class of functions $\mathcal{F}_\theta$, mapping the input $\boldsymbol{x} \in \mathcal{X}$ to an output $\boldsymbol{y} \in \mathcal{Y}$ for a given characterization of parameters θ.

Given a set of input-output pairs $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}$, the training of a generic DNN is usually carried out in a *supervised* manner where the set of learned parameters $\theta^\star$ is given by:

$$\theta^\star = \underset{\theta}{\operatorname{argmin}} \frac{1}{|\mathcal{D}|} \sum_i \|\boldsymbol{y}_i - f_\theta(\boldsymbol{x}_i)\|_2^2. \tag{1.8}$$

Once $\theta^\star$ is obtained, the DNN can be utilized to perform an inference task, viz. $\hat{\boldsymbol{y}} = f_\theta^\star(\boldsymbol{x})$.

Observing Eq. (Equation 5.15), one can deduce that a DNN architecture is indeed performing an iterative procedure in l stages during its forward pass, where at each stage (layer), the particular mathematical manipulation of the input is dictated by the parameterized generic functions g_{θ_i}.

Recall that an inference task is concerned with finding a strategy f_θ that minimizes an energy function $\mathcal{E}(\boldsymbol{x}, \boldsymbol{y}; \Theta)$, i.e.,

$$f_\Theta(\boldsymbol{x}) = \text{argmin}_{\boldsymbol{y}}\ \mathcal{E}(\boldsymbol{x}, \boldsymbol{y}; \Theta) \tag{1.9}$$

In a model-based setting and in the presence of domain knowledge, both the inference strategy f_Θ and the energy function are *dictated* by the domain knowledge, where in a data-driven approach, the inference strategy is *learned* according to the data and an energy function which is defined based on a metric on the data rather than domain-knowledge.

In light of the above, one can view both model-based and data-driven methodologies in a unified manner by considering an energy-based inference model. For instance, in a model-based setting the energy function can be defined as the negative conditional distribution of the input-output variables. The same school of thought can be applied to deep neural network architectures. In particular, for a DNN architecture $\hat{\boldsymbol{y}} = f_\theta(\boldsymbol{x})$, define

$$\mathcal{E}(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{y} - f_\theta(\boldsymbol{x})\|_2^2. \tag{1.10}$$

Then, the inference strategy is similarly given by

$$\text{argmin}_{\boldsymbol{y}}\ \mathcal{E}(\boldsymbol{x}, \boldsymbol{y}) = \text{argmin}_{\boldsymbol{y}}\ \|\boldsymbol{y} - f_\theta(\boldsymbol{x})\|_2^2, \tag{1.11}$$

where the inferred value is exactly given by the forward pass of the learned DNN $\hat{\boldsymbol{y}} = f_\theta(\boldsymbol{x})$. Hence, an energy-based modeling of an inference task not only encompasses the model-based inference models but also consumes the feed-forward deep neural networks.

1.2.1 Model-Based Deep Learning

As it was previously discussed, most of the model-based inference problems can be viewed as an optimization problem of the form:

$$f(\boldsymbol{x};\theta) = \underset{\boldsymbol{y}\in\Omega}{\operatorname{argmin}}\ \mathcal{E}(\boldsymbol{x},\boldsymbol{y};\theta), \tag{1.12}$$

where $f(\boldsymbol{x};\theta)$ denotes the inference strategy, θ denotes the parameters of the system (required for model-based inference) drived from domain-knowledge, $\mathcal{E}$ represent an energy function dictated by domain-knowledge, and $\boldsymbol{\Omega}$ denotes the feasible set of the optimization variable. Two fundamental drawbacks of model-based signal processing techniques can be viewed as the difficulty in obtaining an accurate knowledge of the system parameters θ, as well as the computational complexity of the underlying optimization problem in (Equation 1.12). In particular, it is rarely the case that the inference rule $f(\boldsymbol{x};\theta)$ admits a closed-form solution. Hence, one usually needs to resort to advanced iterative optimization techniques to find the inference rule. Although iterative optimization techniques are powerful, they usually suffer from slow convergence properties. To address these issues, it is intriguing to consider combining data-driven approaches and model-based inference techniques to improve upon the performance of both models. To this end, let us first consider the scenario in which the system parameters θ are unknown, how-

ever, the analytical model of the energy function is available from the domain-knowledge (e.g., the conditional distribution is known). Furthermore, we have access to a dataset $\{(\boldsymbol{x}^i, \boldsymbol{y}^i)\}_{i=1}^B$ consisting of true input-output pairs. In such a scenario, employing the model-based inference rule of (Equation 1.12) is hopeless due to the fact that the energy function cannot be exactly characterized as θ is unknown, though its mathematical expression is available (i.e., we have access to the domain-knowledge). In such a scenario, one can obtain an estimation of θ in a data-driven fashion as follows:

$$\theta^\star = \text{argmin}_\theta \ \frac{1}{B}\|\boldsymbol{y}^i - f(\boldsymbol{x}^i;\theta)\|_2^2. \tag{1.13}$$

However, in many scenarios, any attempt to find $\theta^\star$ is hopeless. This is due to the fact that the solution map $f(\boldsymbol{x};\theta)$ is not *differentiable* with respect to the optimization variable θ.

In order to obtain a differentiable approximation of the solution map $f(\boldsymbol{x};\theta)$, we start with an initial point $\boldsymbol{y}_0(\theta)$ and utilize N gradient descent updating steps:

$$\boldsymbol{y}_{i+1}(\theta) = \boldsymbol{y}_i(\theta) - \boldsymbol{G}_i \nabla_{\boldsymbol{y}} \mathcal{E}(\boldsymbol{x}, \boldsymbol{y}_i; \theta), \ \text{for } i \in \{0, \cdots, N-1\} \tag{1.14}$$

where $\boldsymbol{G}_i \succeq 0$ denotes a positive-definite pre-conditioning matrix controlling the convergence rate of the underlying iterations. Note that the final iterate $\boldsymbol{y}_N(\theta)$ can be viewed as a *differentiable* approximation of the solution map $f(\boldsymbol{x};\theta)$, and with a proper choice of the pre-conditioning matrices, one can prove that as $N \to \infty$ we have $\boldsymbol{y}_N(\theta) \to f(\boldsymbol{x};\theta)$. Alternatively, the computations defined in (Equation 1.14) can be viewed as a N-layer feed-forward neural

network where the parameterized mathematical expression carried out in its i-th layer is specifically designed to carry out one iteration of the underlying optimization problem. Such a neural network can be expressed as:

$$f_\gamma(\boldsymbol{x}_0;\theta) = \boldsymbol{y}_{N-1} \circ \cdots \circ \boldsymbol{y}_1(\boldsymbol{y}_0, \boldsymbol{x};\theta), \tag{1.15}$$

where $\gamma = \{\theta, \boldsymbol{G}_0, \cdots, \boldsymbol{G}_{N-1}\}$ denotes the set of trainable parameters of such network. Then, one can easily carry out the training of such a *model-based* deep architecture in a data-driven manner using automatic differentiation techniques, as the output of the network is differentiable with respect to both θ and $\{\boldsymbol{G}_i\}$.

In the above, we have provided an introduction to the mathematical foundation of the developed model-based deep architectures. We note that more sophisticated optimization techniques and acceleration schemes can be directly used to design model-driven architectures to tackle a wide variety of task expressed in terms of constrained and unconstrained optimization problems. In the following chapters, we detail on various techniques for this purpose.

The resulting deep architecture is now fully model-based and interpretable and has far-fewer trainable parameters as compared to its black-box counterparts. Furthermore, the model-based nature of the derived architecture allows for performing mathematical analysis of the underlying model both during the training and once the training is completed (e.g., analyzing the generalization bounds).

In the following chapters, we build upon the above presented ideas and demonstrate the effectiveness of model-based deep learning for various signal processing applications. Furthermore, a literature review of the existing methodologies as well as the details of the procedural design of the proposed model-based deep neural networks will be discussed in each respective chapter.

The details of the materials used in each chapter is as follows:

Chapter 2: This chapter is based on the following published articles[1]:

[**J1**] S. Khobahi, N. Shlezinger, M. Soltanalian and Y. C. Eldar, "LoRD-Net: Unfolded Deep Detection Network With Low-Resolution Receivers," in IEEE Transactions on Signal Processing, vol. 69, pp. 5651-5664, 2021, doi: 10.1109/TSP.2021.3117503.

[**C1**] S. Khobahi, N. Shlezinger, M. Soltanalian and Y. C. Eldar, "Model-Inspired Deep Detection with Low-Resolution Receivers," 2021 IEEE International Symposium on Information Theory (ISIT), 2021, pp. 3349-3354, doi: 10.1109/ISIT45174.2021.9517812.

Chapter 3: This chapter is based on the following published articles:

[**C2**] S. Khobahi, N. Naimipour, M. Soltanalian and Y. C. Eldar, "Deep Signal Recovery with One-bit Quantization," ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2019, pp. 2987-2991, doi: 10.1109/ICASSP.2019.8683876.

Chapter 4: This chapter is based on the following published articles:

---

[1][C]: Peer-Reviewed Conference Paper, [J]: Journal Papers, [SJ]: Submitted Journal Paper.

[**J2**] S. Khobahi and M. Soltanalian, "Model-Based Deep Learning for One-Bit Compressive Sensing," in IEEE Transactions on Signal Processing, vol. 68, pp. 5292-5307, 2020, doi: 10.1109/TSP.2020.3022319.

[**C3**] S. Khobahi, A. Bose and M. Soltanalian, "Deep One-Bit Compressive Autoencoding," 2021 IEEE Statistical Signal Processing Workshop (SSP), 2021, pp. 371-375, doi: 10.1109/SSP49050.2021.95

Chapter 5: This chapter is based on the following published articles:

[**SJ1**] N. Naimipour, S. Khobahi, and M. Soltanalian. "Unfolded algorithms for deep phase retrieval.", Submitted to IEEE Transactions on Signal Processing, arXiv preprint arXiv:2012.11102

[**C4**] N. Naimipour, S. Khobahi and M. Soltanalian, "UPR: A Model-Driven Architecture for Deep Phase Retrieval," 2020 54th Asilomar Conference on Signals, Systems, and Computers, 2020, pp. 205-209, doi: 10.1109/IEEECONF51394.2020.9443438.

Chapter 6: This chapter is based on the following published articles:

[**C5**] S. Khobahi, A. Bose and M. Soltanalian, "Deep Radar Waveform Design for Efficient Automotive Radar Sensing," 2020 IEEE 11th Sensor Array and Multichannel Signal Processing Workshop (SAM), 2020, pp. 1-5, doi: 10.1109/SAM48682.2020.9104367.

Chapter 7: This chapter is based on the following published articles:

[**C6**] C. Agarwal, S. Khobahi, A. Bose, M. Soltanalian and D. Schonfeld, "DEEP-URL: A Model-Aware Approach to Blind Deconvolution Based on Deep Unfolded Richardson-Lucy Network," 2020 IEEE International Conference on Image Processing (ICIP), 2020, pp. 3299-3303, doi: 10.1109/ICIP40778.2020.9190825.

Chapter 8: This chapter is based on the following published articles:

[**C7**] S. Khobahi, A, Mostajeran, M. Emadi, P. Wang, and M. Soltanalian, "Guaranteed Deep Learning for Reliable Radar Signal Processing," IEEE Sensor Array and Multichannel Signal Processing Workshop (SAM) 2022.

Other Contributions by Author: In addition to the above publications, the author has published the following articles during his Ph.D. studies:

[**J3**] Y. Zeng, S. Khobahi, and M. Soltanalian. "One-Bit Compressive Sensing: Can We Go Deep and Blind?." In press, IEEE Signal Processing Letter, 2022.

[**J4**] A. Bose, S. Khobahi, and M. Soltanalian, "Efficient waveform covariance matrix design and antenna selection for MIMO radar," Signal Processing 183 (2021): 107985.

[**J5**] S. Khobahi, M. Soltanalian, F. Jiang and A. L. Swindlehurst, "Optimized Transmission for Parameter Estimation in Wireless Sensor Networks," in IEEE Transactions on Signal and Information Processing over Networks, vol. 6, pp. 35-47, 2020, doi: 10.1109/TSIPN.2019.2945631.

[**C8**] C. Agarwal, S. Khobahi, D. Schonfeld and M. Soltanalian . CoroNet: a deep network architecture for enhanced identification of COVID-19 from chest x-ray images. In Medical Imaging 2021: Computer-Aided Diagnosis (Vol. 11597, pp. 484-490), SPIE, (2021, February).

[**C9**] A. Bose, S. Khobahi and M. Soltanalian, "Joint Optimization of Waveform Covariance Matrix and Antenna Selection for MIMO Radar," 2019 53rd Asilomar Conference on Signals, Systems, and Computers, 2019, pp. 1534-1538, doi: 10.1109/IEEECONF44664.2019.9048709.

[**C10**] S. Khobahi and M. Soltanalian, "Signal Recovery From 1-Bit Quantized Noisy Samples via Adaptive Thresholding," 2018 52nd Asilomar Conference on Signals, Systems, and Computers, 2018, pp. 1757-1761, doi: 10.1109/ACSSC.2018.8645383.

[**C11**] S. Khobahi and M. Soltanalian, "Optimized Transmission for Consensus in Wireless Sensor Networks," 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2018, pp. 3419-3423, doi: 10.1109/ICASSP.2018.8461401.

Part I

Areas

I-A

Model-Based Deep Learning for Communication Systems

CHAPTER 2

LORD-NET: UNFOLDED DEEP DETECTION NETWORK WITH LOW-RESOLUTION RECEIVERS

Overview: The need to recover high-dimensional signals from their noisy low-resolution quantized measurements is widely encountered in communications and sensing. In this paper, we focus on the extreme case of one-bit quantizers, and propose a deep detector entitled LoRD-Net for recovering information symbols from one-bit measurements. Our method is a model-aware data-driven architecture based on deep unfolding of first-order optimization iterations. LoRD-Net has a task-based architecture dedicated to recovering the underlying signal of interest from the one-bit noisy measurements without requiring prior knowledge of the channel matrix through which the one-bit measurements are obtained. The proposed deep detector has much fewer parameters compared to black-box deep networks due to the incorporation of domain-knowledge in the design of its architecture, allowing it to operate in a data-driven fashion while benefiting from the flexibility, versatility, and reliability of model-based optimization methods. LoRD-Net operates in a blind fashion, which requires addressing both the non-linear nature of the data-acquisition system as well as identifying a proper optimization objective for signal recovery. Accordingly, we propose a two-stage training method for LoRD-Net, in which the first stage is dedicated to identifying the proper form of the optimization process to unfold, while the latter trains the resulting

---

model in an end-to-end manner. We numerically evaluate the proposed receiver architecture for one-bit signal recovery in wireless communications and demonstrate that the proposed hybrid methodology outperforms both data-driven and model-based state-of-the-art methods, while utilizing small datasets, on the order of merely ~ 500 samples, for training.

2.1 Introduction

Analog-to-digital conversion plays an important role in digital signal processing systems. While physical signals take values in continuous-time over continuous sets, they must be represented using a finite number of bits in order to be processed in digital hardware [26]. This operation is carried out using analog-to-digital converters (ADCs), which typically perform uniform sampling followed by a uniform quantization of the discrete-time samples. When using high-resolution ADCs, this conversion induces a minimal distortion, allowing to effectively process the signal using methods derived assuming access to the continuous-amplitude samples. However, the cost, power consumption and memory requirements of ADCs grow with the sampling rate and the number of bits assigned to each sample [27]. Consequently, recent years have witnessed an increasing interest in digital signal processing systems operating with low-resolution ADCs. Particularly, in multiple-input multiple-output (MIMO) communication receivers, which are required to simultaneously capture multiple analog signals with high bandwidth, there is a growing need to operate reliably with low-resolution ADCs [28]. The most coarse form of quantization is reduction of the signal to a single bit per sample, which may be accomplished via comparing the sample to some reference level, and recording whether the signal is above or below the reference. One-bit acquisition allows using high sampling rates at a low cost and

low energy consumption. Due to such favorable properties of one-bit ADCs, they have been employed in a wide array of applications, including in wireless communications [29–31], radar signal processing [32–34], and sparse signal recovery [35, 36].

The non-linear nature of low-resolution quantization makes symbol detection a challenging task. This situation is significantly exacerbated in practical one-bit communication and sensing where the channel is to be estimated in conjunction with symbol detection. A *coherent* symbol detection task is concerned with recovering the underlying signal of interest from the one-bit measurements assuming the channel state information (CSI) is known at the receiver. On the other hand, the more difficult task of *blind* symbol detection, which is the focus here, carries out recovery of the underlying transmitted symbols when CSI is not available.

Two main strategies have been proposed in the literature to facilitate operation with low-resolution ADCs: The first designs the overall acquisition system in light of the task for which the signals are acquired. For instance, MIMO communication receivers acquire their channel output in order to extract some underlying information, e.g., symbol detection. As the analog signals are not required to be recovered from their digital representation, one can design the acquisition system to reliably infer the desired information while operating with low resolution ADCs [37–41]. Such task-based quantization systems rely on pre-quantization processing, which requires dedicated hardware in the form of hybrid receiver architectures [42,43] or unique antenna structures [44, 45], which are configured along with the quantization rule.

An alternative approach to task-based quantization, which does not require additional configurable analog hardware and is the focus of the current work, is to recover the desired infor-

mation from the distorted coarsely discretized representation of the signal in the digital domain. The main benefit of schemes carried out only in the digital domain is their simplicity of implementation, as they do not require to introduce modifications to the quantization system and circumvent the need for adding pre-quantization analog processing hardware. In the context of MIMO systems, various methods have been proposed in the literature for channel estimation and signal decoding from quantized outputs, including model-based signal processing methods as surveyed in [46], as well as model-agnostic systems based on machine learning and data-driven techniques [47–54].

Most existing model-based detection algorithms require coherent operation, i.e., they rely on prior knowledge of the CSI and other system parameters. Among these works are the near-Maximum Likelihood (nML) detector proposed for one-bit MIMO receivers in [55], the linear receivers studied in [56, 57], and the message passing based detectors considered in [58, 59]. The fact that such approaches require accurate CSI led to several works specifically dedicated to CSI estimation in the presence of low-resolution ADCs. These include [55, 60], which studied maximum-likelihood estimation for recovering the CSI in the presence of one-bit data, the works in [61, 62], which developed linear estimators for CSI estimation purposes in one-bit MIMO systems, and [63] which focuses on sparse channels and utilizes one-bit sparse recovery methods for CSI estimation. However, all these strategies inevitably induce non-negligible CSI estimation error, which may notably degrade the accuracy in signal detection based on the estimated CSI.

Over the past several years, data-driven methods, and specifically deep neural networks (DNNs), have attracted unprecedented attention from research communities across the board.

The advent of low-cost specialized powerful computing resources and the continually increasing amount of massive data generated by the human population and machines, along with new optimization and learning methods, have paved the way for DNNs and machine learning-based models to prove their effectiveness in many engineering areas, such as computer vision and natural language processing [7]. DNNs learn their mapping from data in a model-agnostic manner, and can thus facilitate non-coherent (blind) detection.

Previously proposed DNN-aided symbol detection techniques for communication receivers can be divided based on their receiver architectures; namely, those that utilize conventional machine learning architectures for detection, including [64–66], and schemes combining DNNs with model-based detection methods, such as the blind DNN-aided receivers proposed in [67–70] and the coherent detectors of [71, 72], see also surveys in [73, 74]. In the context of one-bit DNN-aided receivers, previous works to date focus mainly on the first approach, i.e., applying conventional DNNs for the overall detection task. Among these works are [47, 50] and [48], which applied generic DNNs for channel estimation in one-bit MIMO receivers. The application of conventional architectures for symbol detection was studied in [49, 52] and [53], while [51] showed that autoencoders can facilitate the design of error correction codes for communications with one-bit receivers. Recently, the authors in [54] considered the problem of symbol detection for a one-bit massive MIMO system and proposed a linear estimator module based on the Bussgang decomposition technique combined with a model-driven neural network.

The vast majority of the aforementioned works on learning-aided one-bit receivers rely on conventional DNN architectures. Such DNNs require a massive amount of training samples and

must be trained on data from the same (or a similar) statistical model as the one under which they are required to operate, imposing a major challenge in dynamic wireless communications. In fact, the use of generic black-box DNNs is mostly justified in applications where a satisfactory description of the underlying governing dynamics of the system is not achievable, as is the case in computer vision and natural language processing fields. As surveyed above, this is not the case in the field of one-bit MIMO systems. This gives rise to the need that is bridging the gap between data-driven and model-based approaches in this context, and moving towards specialized deep learning models for signal processing techniques in one-bit MIMO systems—which is the aim of this work.

In this paper, we develop a hybrid model-based and data-driven system which learns to carry out blind symbol detection from one-bit measurements. The proposed architecture, referred to as LoRD-Net (Low Resolution Detection Network), combines the well-established model-based maximum-likelihood estimator (MLE) with machine learning tools through the deep unfolding method [75–80] for designing DNNs based on model-based optimization algorithms. To derive LoRD-Net, we first formulate the MLE for the task of symbol detection from one-bit samples. Next, we resort to first-order gradient-based methods for the MLE computation, and unfold the iterations onto layers of a DNN. The resulting LoRD-Net learns to carry out MLE-approaching symbol detection without requiring CSI.

Applying conventional gradient-based optimization methods requires knowledge of the underlying system parameters, i.e., full CSI. Hence, a typical approach to unfold such a symbol detection algorithm would be to estimate the unknown parameters from training, and substi-

tute it into the unfolded network [70]. We show that instead of estimating the unknown system parameters, it is preferable to learn an *alternative channel* which allows the receiver to detect the symbols reliably. Surprisingly, we demonstrate that the alternative channel learned by LoRD-Net is in general not the true channel. Based on this observation, we propose a two-stage training procedure, comprised of learning the proper optimization process to unfold, followed by an end-to-end training of the unfolded DNN.

The proposed LoRD-Net has thus the following properties:

i) Compared to the vanilla MLE symbol detector, our model does not need to estimate the channel separately.

ii) Owing to its hybrid nature, it has low computational cost in operation and is highly scalable, facilitating much faster inference as compared to its black-box data-driven and model-based counterparts.

iii) The proposed deep architecture is interpretable and has far fewer parameters compared to existing black-box deep learning solutions. This follows from the incorporation of domain-knowledge in the design of the network architecture (i.e., being model-based), allowing LoRD-Net to train with much fewer labeled samples as compared to existing data-driven one-bit receivers.

We verify the above characteristics of LoRD-Net in an experimental study, where we show that training of the proposed LoRD-Net architecture can be performed with far fewer samples as compared to its data-driven counterparts, and demonstrate substantially superior performance

compared to existing model-based and data-driven algorithms for symbol detection in massive MIMO channels with one-bit ADCs.

The rest of the paper is organized as follows. In Section 2.2, we present the considered system model and the corresponding MLE formulation. In Section 2.3, we derive LoRD-Net by unfolding the first-order gradient iterations associated with the MLE computation, and present its two-stage training procedure. Section 2.4 provides a detailed numerical analysis of LoRD-Net applied to MIMO communications. Finally, Section 2.5 concludes the paper.

Throughout the paper, we use the following notation. Bold lowercase and bold uppercase letters denote vectors and matrices, respectively. We use $(\cdot)^T$, Diag$(\cdot)$, and sign$(\cdot)$, and log$\{\cdot\}$ to denote the transpose operator, the diagonal matrix formed by the entries of the vector argument, the sign operator, and the natural logarithm, respectively. The symbol $\odot$ represents the Hadamard product, while $\mathbf{1}$ and $\mathbf{0}$ are the all-one and all-zero vectors/matrices. The i-th entry of the vector $\boldsymbol{x}$ is x_i, and $\|\boldsymbol{x}\|_p$ is the ℓ_p-norm of $\boldsymbol{x}$; $\mathcal{M}^n$ is the n-ary Cartesian product of a set $\mathcal{M}$, and $\mathbb{S}_+$ denotes the cone of symmetric positive definite matrices.

2.2 System Model and Preliminaries

In this section, we discuss the considered system model. We focus on one-bit data acquisition and blind signal recovery. We then formulate the MLE for this problem, which is used in designing the LoRD-Net architecture in Section 2.3.

2.2.1 Problem Formulation

We consider a low-resolution data-acquisition system which utilizes m one-bit ADCs. By letting $\boldsymbol{y} \in \mathbb{R}^m$ denote the received signal, the discrete output of the ADCs can be written as $\boldsymbol{r} = \text{sign}\,(\boldsymbol{y} - \boldsymbol{b})$, where $\boldsymbol{b} \in \mathbb{R}^m$ denotes the vector of quantization thresholds, and sign$(\cdot)$ is the

sign function, i.e., $\text{sign}(x) = +1$ if $x \geq 0$ and $\text{sign}(x) = -1$ otherwise. The received vector $\boldsymbol{y}$ is statistically related to the unknown vector of interest $\boldsymbol{x} \in \mathcal{M}^n \subseteq \mathbb{R}^n$ according to the following linear relationship:

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{n}, \tag{2.1}$$

where $\boldsymbol{n} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{C})$ denotes additive Gaussian noise with a covariance matrix of the form $\boldsymbol{C} = Diag(\sigma_0^2, \sigma_1^2, \ldots, \sigma_{m-1}^2)$ with diagonal entries $\{\sigma_i^2\}_{i=0}^{m-1}$ representing the noise variance at each respective dimension, and $\boldsymbol{H} \in \mathbb{R}^{m \times n}$ is the channel matrix. We assume that the elements of the unknown vector $\boldsymbol{x}$ are chosen independently from a finite alphabet $\mathcal{M} = \{s_1, s_2, \cdots, s_{|\mathcal{M}|}\}$. This setup represents low-resolution receivers in uplink multi-user MIMO systems, where $\boldsymbol{x}$ is the symbols transmitted by the users, and $\boldsymbol{y}$ is the corresponding channel output, as illustrated in Figure 1.

The overall dynamics of the system are thus compactly expressed as:

$$\boldsymbol{r} = \text{sign}(\boldsymbol{H}\boldsymbol{x} + \boldsymbol{n} - \boldsymbol{b}). \tag{2.2}$$

In the sequel, we refer to $\Theta = \{\boldsymbol{H}, \boldsymbol{C}\}$ as the system parameters. Note that the above system model can be modified using conventional transformations to accommodate a complex-valued system model.

Our main goal is to perform the task of symbol detection, i.e., recover $\boldsymbol{x}$, from the collected one-bit measurements $\boldsymbol{r}$. We focus on blind (non-coherent) recovery, namely, the system

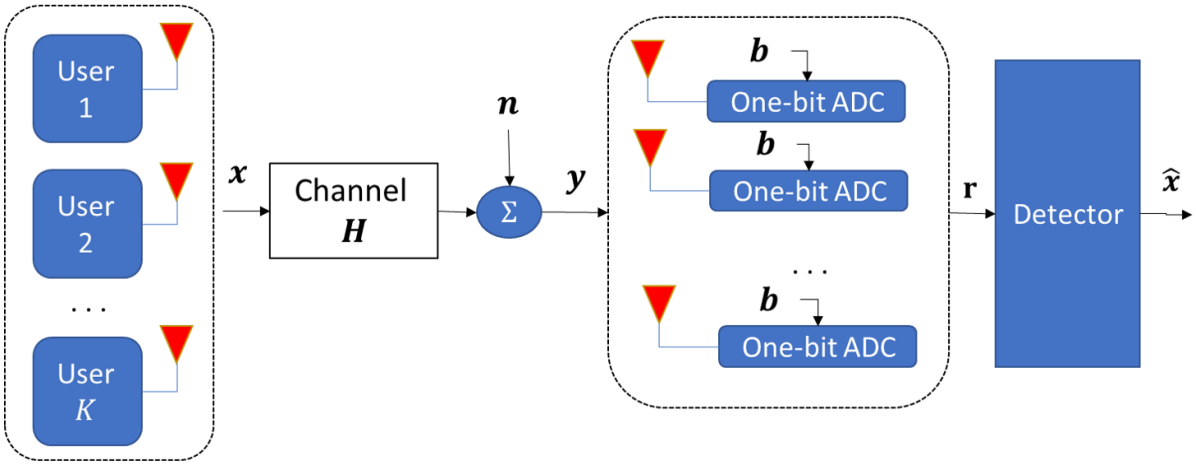


Figure 1. System model illustration.

parameters $\Theta = \{\boldsymbol{H}, \boldsymbol{C}\}$, i.e., the channel matrix and the covariance of the noise, are not available to the receiver. Nonetheless, the receiver has access to a limited set of B labeled samples $\{\boldsymbol{x}_p^b, \boldsymbol{r}_p^b\}_{b=0}^{B-1}$, representing, e.g., pilot transmissions. The quantization thresholds of the ADCs, i.e., the vector $\boldsymbol{b}$, are assumed to be fixed and known. While we do not consider the selection of $\boldsymbol{b}$ in the following, we discuss in the sequel how its optimization can be incorporated into the detection method.

2.2.2 Maximum Likelihood Recovery

To understand the challenges associated with blind low-resolution detection, we next discuss the MLE for recovering $\boldsymbol{x}$ from $\boldsymbol{r}$. In particular, the intuitive model-based approach is to utilize the labeled data to estimate the system parameters Θ, and then to use this estimation to

compute the coherent (non-blind) MLE. Therefore, to highlight the limitations of this strategy, we assume here that the system parameters $\Theta = \{\boldsymbol{H}, \boldsymbol{C}\}$ are fully known at the receiver. Let

$$\begin{aligned}\mathcal{F}_\Theta(\boldsymbol{x};\boldsymbol{r}) &\triangleq \log \Pr(\boldsymbol{r}|\boldsymbol{x},\Theta) \\ &\overset{(a)}{=} -\sum_{i=0}^{m-1} \log\left\{ Q\left(\frac{r_i}{\sigma_i}\left(b_i - \boldsymbol{h}_i^T \boldsymbol{x}\right)\right)\right\}, \end{aligned} \tag{2.3}$$

represent the log-likelihood objective for a given vector of one-bit observations $\boldsymbol{r}$, where (a) is proven in [31]. The coherent MLE is then given by

$$\hat{\boldsymbol{x}}_{\mathrm{ML}}(\boldsymbol{r}) = \underset{\boldsymbol{x}\in\mathcal{M}^n}{\operatorname{argmax}} \ \mathcal{F}_\Theta(\boldsymbol{x};\boldsymbol{r}). \tag{2.4}$$

Although the MLE in (Equation 2.4) has full accurate knowledge of the parameters Θ, its computation is still challenging. The main difficulty emanates from solving the underlying optimization problem in the discrete domain, implying that the MLE requires an exhaustive search over the discrete domain $\mathcal{M}^n$, whose computational complexity grows exponentially with n. A common strategy to tackle the discrete optimization problem in (Equation 2.4) is to relax the search space to be continuous. This results in the following relaxed unconstrained MLE rule:

$$\bar{\boldsymbol{x}}_\Theta(\boldsymbol{r}) = \underset{\boldsymbol{x}\in\mathbb{R}^n}{\operatorname{argmax}} \ \mathcal{F}_\Theta(\boldsymbol{x};\boldsymbol{r}). \tag{2.5}$$

The optimization problem in (Equation 2.5) is convex due to the log-concavity of $Q(\cdot)$, and thus can be solved using first-order gradient optimization. In particular, the negative log-likelihood function with respect to the unknown vector $\boldsymbol{x}$ can be compactly expressed as [31]:

$$\nabla_{\boldsymbol{x}} \mathcal{F}_{\Theta}(\boldsymbol{x}; \boldsymbol{r}) = \boldsymbol{H}^T \tilde{\boldsymbol{R}}\, \boldsymbol{\eta} \left(\tilde{\boldsymbol{R}} \left(\boldsymbol{b} - \boldsymbol{H}\boldsymbol{x} \right) \right), \tag{2.6}$$

where $\boldsymbol{\eta}$ is a non-linear function defined as $\boldsymbol{\eta}(\boldsymbol{x}) \triangleq Q^{'}(\boldsymbol{x}) \oslash Q(\boldsymbol{x})$, in which the operator $\oslash$ denotes the element-wise division operation, $Q^{'}(x)$ is the derivative of $Q(x)$, that is given by the negative probability density function of a standard Normal distribution, and $\tilde{\boldsymbol{R}} = \boldsymbol{R}\boldsymbol{C}^{-\frac{1}{2}}$ is the semi-whitened version of the *one-bit matrix* $\boldsymbol{R} = Diag\left(r_0, \ldots, r_{m-1}\right)$.

As $\bar{\boldsymbol{x}}_{\Theta}(\boldsymbol{r})$ obtained via (Equation 2.5) is not guaranteed to take values in $\mathcal{M}^n$, the final estimate of the symbols is obtained by applying a projection operator $\mathcal{P}_{\mathcal{M}^n} : \mathbb{R}^n \mapsto \mathcal{M}^n$ to $\bar{\boldsymbol{x}}(\boldsymbol{r})$. This operator maps the continuous input vector onto its closest lattice point on the discrete set $\mathcal{M}^n$, i.e.,

$$\mathcal{P}_{\mathcal{M}^n}(\boldsymbol{x}) = \underset{\boldsymbol{z} \in \mathcal{M}^n}{\operatorname{argmin}} \|\boldsymbol{z} - \boldsymbol{x}\|_2^2. \tag{2.7}$$

Tackling a discrete program via continuous relaxation, as done in (Equation 2.5), is subject to an inherent drawback. As a case in point, one can only expect $\bar{\boldsymbol{x}}_{\Theta}(\boldsymbol{r})$ to provide an accurate approximation of the true MLE if the real-valued vector $\bar{\boldsymbol{x}}_{\Theta}(\boldsymbol{r})$ is very close to the discrete valued MLE $\hat{\boldsymbol{x}}_{ML}(\boldsymbol{r})$. In such a case, the MLE is obtained by projecting into the lattice points in $\mathcal{M}^n$. However, this is not the case in many scenarios, and specifically, when the noise variance in each

respective dimension is high. In other words, it is not necessarily the case that the minimizer of the objective function on the continuous domain (Equation 2.5) is close to the MLE, which takes values in the discrete set $\mathcal{M}^n$. Note that utilizing the true system parameters will only lead to optimal estimates when considering the original discrete problem (Equation 2.4). In fact, one can no longer necessarily argue that the true system parameters are optimal choices for Θ in the relaxed MLE. This insight, which is obtained from the computation of the coherent MLE, is used in our derivation of the blind unfolded detector in the following section.

2.3 Proposed Methodology

In this section, we present the proposed **Lo**w **R**esolution **D**etection **Net**work, abbreviated as **LoRD-Net**. We begin with a high-level description of LoRD-Net in Subsection 2.3.1. Then, we present the unfolded architecture in Subsection 2.3.2 and discuss the training procedure in Subsection 2.3.3. Finally, we provide a discussion in Subsection 2.3.4.

2.3.1 High-Level Description

As noted in the previous section, the intuitive approach to blind symbol detection is to utilize the labeled data $\{\boldsymbol{x}_p^b, \boldsymbol{r}_p^b\}_{b=0}^{B-1}$ to estimate the true system model Θ, and then to recover the symbol vector $\boldsymbol{x}$ from $\boldsymbol{r}$ using the MLE. Nonetheless, the coherent MLE (Equation 2.4) is computationally prohibitive, while its relaxed version in (Equation 2.5) may be inaccurate. Alternatively, one can seek a purely data-driven strategy, using the data to train a black-box highly-parameterized DNN for detection, requiring a massive amount of labeled samples. Consequently, to facilitate accurate detection at affordable complexity and with limited data, we design LoRD-Net via model-based deep learning [81], by combining the learning of a *competitive objective*, combined with *deep unfolding* of the relaxed MLE.

Learning a competitive objective refers to the setting of the unknown system parameters Θ. However, the goal here is not to estimate the *true system parameters*, but rather the ones for which the solution to the relaxed MLE coincides with the true value of $\boldsymbol{x}$. This system identification problem can be written as

$$\mathcal{F}_{\Theta^\star}(\boldsymbol{r};\boldsymbol{x}) = \min_{\Theta} \frac{1}{B} \sum_{b=0}^{B-1} \left\| \bar{\boldsymbol{x}}_{\Theta}(\boldsymbol{r}_p^b) - \boldsymbol{x}_p^b \right\|_2^2, \tag{2.8}$$

where $\bar{\boldsymbol{x}}_\Theta$ is the relaxed MLE (Equation 2.5). The optimization problem (Equation 2.8) yields a surrogate objective function $\mathcal{F}_{\Theta^\star}$, or equivalently, a set of system parameters $\Theta^\star$, referred to as a *competitive objective* to the true $\mathcal{F}_\Theta$. An illustration of such a competitive objective obtained for the case of $n = 1$ is depicted in Figure 2.

The main difficulty in solving (Equation 2.8) stems from the fact that $\bar{\boldsymbol{x}}_\Theta(\boldsymbol{r}) = \underset{\boldsymbol{x}\in\mathbb{R}^n}{\operatorname{argmax}}\ \mathcal{F}_\Theta(\boldsymbol{x};\boldsymbol{r})$ is not differentiable with respect to the system parameters Θ. We overcome this obstacle by applying a differentiable approximation of $\bar{\boldsymbol{x}}(\boldsymbol{r})$, or equivalently, an algorithm that approximates the argmax operator specific to our problem. Since $\bar{\boldsymbol{x}}_\Theta(\boldsymbol{r})$ can be computed by first-order gradient methods, we design a deep unfolded network [76] to compute the relaxed MLE in manner which is differentiable with respect to Θ. The usage of deep unfolding allows not only to learn a competitive objective via (Equation 2.8), but also results in accurate inference with a reduced number of iterations compared to model-based first-order gradient optimization. Furthermore, the unfolded network utilizes a relatively small amount of trainable parameters, thus enabling learning from small amounts of labeled samples.

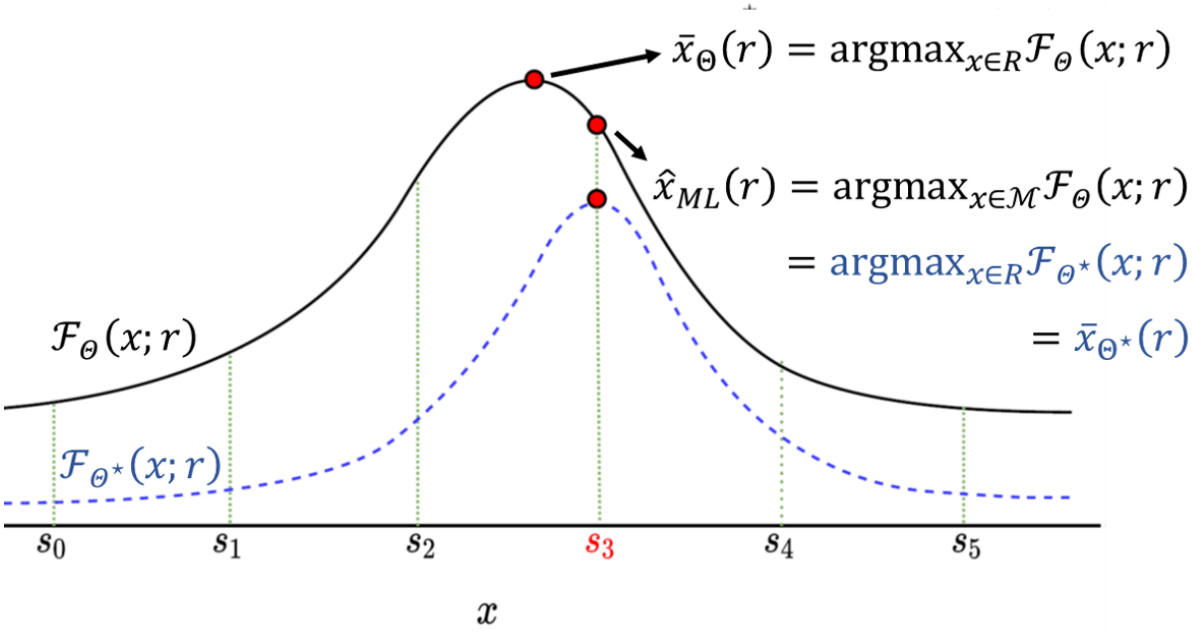


Figure 2. An illustration of the relation between the optimal point of a competitive objective function (dashed blue line) and the true MLE $\hat{\boldsymbol{x}}_{ML}$ obtained by an exact maximization of the log-likelihood objective function (solid black line) over the discrete set $\mathcal{M}$ as well as an approximation of the MLE $\bar{\boldsymbol{x}}_\Theta$ obtained by a maximization of the log-likelihood objective function over the continuous space $\mathbb{R}$, when the true transmitted symbol is $s_3 \in \mathcal{M}$.

2.3.2 LoRD-Net Architecture

We now present the architecture of LoRD-Net, which maps the low resolution $\boldsymbol{r}$ into an estimated $\hat{\boldsymbol{x}}$. For given system parameters Θ whose learning is detailed in Subsection 2.3.3 based on the competitive objective rationale described above, LoRD-Net is obtained by unfolding the iterations of a first-order optimization of the relaxed MLE (Equation 2.5). Our derivation thus begins by formulating the first-order methods to iteratively solve (Equation 2.5) for a given Θ.

Let $g_{\phi_i} : \mathbb{R}^n \mapsto \mathbb{R}$ be a parametrized operator defined as $g_{\phi_i}(\boldsymbol{x}; \Theta, \boldsymbol{r}) = \boldsymbol{x} - \boldsymbol{G}_i \nabla_{\boldsymbol{x}} \mathcal{F}_\Theta(\boldsymbol{x}; \boldsymbol{r})$, where $\boldsymbol{G}_i \in \mathbb{R}^{n \times n}$ is a positive-definite weight matrix and $\phi_i = \{\boldsymbol{G}_i\}$ denotes the set of parameters of the operator g_{ϕ_i}. Such a linear operator can be used to model a first-order optimization solver by considering a composition of t mappings of the form:

$$\begin{aligned} \boldsymbol{x}_{t+1} = \mathcal{G}^t_{\boldsymbol{\phi}}(\boldsymbol{x}_0; \Theta, \boldsymbol{r}) &= \boldsymbol{x}_t - \boldsymbol{G}_t \nabla_{\boldsymbol{x}} \mathcal{F}_\Theta(\boldsymbol{x}_t; \Theta, \boldsymbol{r}), \\ &\triangleq g_{\phi_t} \circ \cdots \circ g_{\phi_1} \circ g_{\phi_0}(\boldsymbol{x}_0; \Theta, \boldsymbol{r}) \end{aligned} \tag{2.9}$$

where $\boldsymbol{x}_0$ is an initial point, $\boldsymbol{\phi} = \{\phi_0, \cdots, \phi_{t-1}\}$ is the set of parameters of the overall mapping $\mathcal{G}^t_{\boldsymbol{\phi}}$. The mapping (Equation 2.9) is differentiable with respect to the system parameters Θ, and its local weights $\boldsymbol{\phi}$. For a fixed number of iterations L, the resulting function $\mathcal{G}^L_{\boldsymbol{\phi}}(\boldsymbol{x}_0; \Theta, \boldsymbol{r})$ is thus differentiable with respect to the set of parameters $\{\boldsymbol{\phi}, \Theta\}$ and its input (unlike the original *argmax* operator). Therefore, it can now be used as a differentiable approximation of $\bar{\boldsymbol{x}}_\Theta(\boldsymbol{r})$, which allows for a training (optimization) over the set of its parameters based on the gradient-based training algorithms and the back-propagation technique.

Following the deep unfolding framework [76], the function $\mathcal{G}_{\boldsymbol{\phi}}^{L}(\boldsymbol{x}_0;\Theta,\boldsymbol{r})$ can be implemented as a L-layer feed-forward neural network, where the initial point $\boldsymbol{x}_0$ and the one-bit samples $\boldsymbol{r}$ constitute the input to the network, and with trainable parameters that are given by $\{\Theta,\boldsymbol{\phi}\}$. By (Equation 2.6), the i-th layer computes:

$$g_{\phi_i}(\boldsymbol{x}_i;\Theta,\boldsymbol{r}) = \boldsymbol{x}_i - \boldsymbol{G}_i\boldsymbol{z}_i, \text{ with} \tag{2.10}$$

$$\boldsymbol{z}_i = \boldsymbol{H}^T\tilde{\boldsymbol{R}}\,\boldsymbol{\eta}\left(\tilde{\boldsymbol{R}}\left(\boldsymbol{b}-\boldsymbol{H}\boldsymbol{x}_i\right)\right), \tag{2.11}$$

where the overall dynamics of the LoRD-Net is given by:

$$\mathcal{G}_{\boldsymbol{\phi}}^{L}(\boldsymbol{x}_0;\Theta,\boldsymbol{r}) = g_{\phi_{L-1}}\circ g_{\phi_{L-2}}\circ\cdots\circ g_{\phi_0}(\boldsymbol{x}_0;\Theta,\boldsymbol{r}). \tag{2.12}$$

Each vector $\boldsymbol{x}_i$ in (Equation 2.10) represents the input to the i-th layer (or equivalently, the output of the previous iteration), with $\boldsymbol{x}_0$ being the input of the entire network (which represents the initial point for the optimization task). Upon the arrival of any new one-bit measurement $\boldsymbol{r}$, the recovered symbols $\hat{\boldsymbol{x}}$ are obtained by feed-forwarding $\boldsymbol{r}$ through the L layers of LoRD-Net. In order to obtain discrete samples, the output of LoRD-Net is projected into the feasible discrete set $\mathcal{M}^n$, viz.

$$\hat{\boldsymbol{x}} = \mathcal{P}_{\mathcal{M}^n}\left(\mathcal{G}_{\boldsymbol{\phi}}^{L}(\boldsymbol{x}_0;\Theta,\boldsymbol{r})\right). \tag{2.13}$$

An illustration of LoRD-Net is depicted in Figure 3.

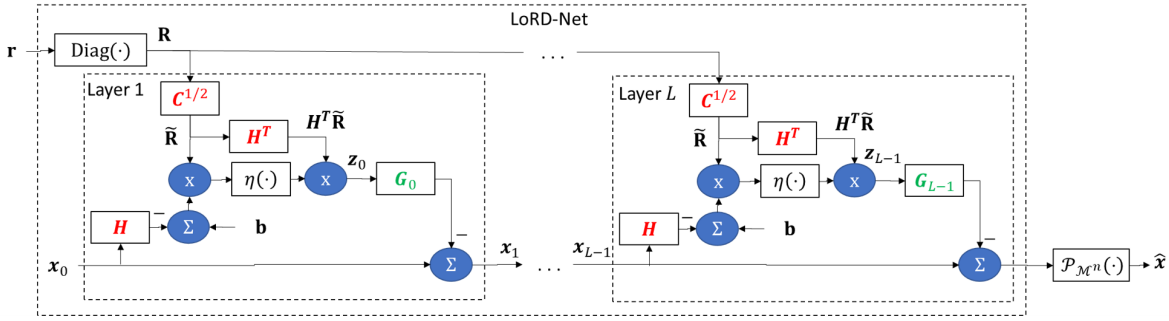


Figure 3. An illustration of LoRD-Net, where trainable system parameters and unfolded weights are highlighted in red and green colors, respectively.

We note that one can also propose an alternative architecture, derived by applying the projection operator $\mathcal{P}_{\mathcal{M}^n}$ at the output of each layer, i.e., by defining $g_{\phi_i}(\boldsymbol{x}_i;\theta,\boldsymbol{r}) = \mathcal{P}_{\mathcal{M}^n}(\boldsymbol{x}_i - \boldsymbol{G}_i\boldsymbol{z}_i)$. Such a setting corresponds to the unfolding of a projected gradient descent method. However, our numerical investigations have consistently shown that such an architecture suffers from the vanishing gradient problem during training and a significant degradation in performance. As a result, we implement LoRD-Net while applying the projection operator once on the output of the network, and only during inference, as discussed above.

In principle, one can fix $\boldsymbol{G}_i = \delta\boldsymbol{I}$ for some $\delta > 0$, for which (Equation 2.12) represents L steps of gradient descent with step size δ. In the unfolded implementation, the weights $\{\boldsymbol{G}_i\}$ are tuned from data, allowing to detect with less iterations, i.e., layers. As a result, once LoRD-Net is trained, i.e., its weight matrices $\boldsymbol{\phi} = \{\boldsymbol{G}_i\}$ and the unknown system parameters Θ are learned from data, it is capable of carrying out fast inference, owing to its hybrid model-based/data-

driven structure. Furthermore, the number of iterations L is optimized to boost fast inference in the training procedure, as detailed in the following.

2.3.3 Training Procedure

Herein, we present the training procedure for LoRD-Net. In particular, our main goal is to perform inference of the unknown system parameters Θ based on the rationale detailed in Subsection 2.3.1, i.e., to obtain a competitive objective. The learning competitive objective is used to tune the weights of the unfolded network $\boldsymbol{\phi}$. Accordingly, we present a two-stage training procedure for LoRD-Net (Equation 2.12). Once the training of the LoRD-Net is completed, it carries out symbol detection from one-bit information without requiring the knowledge of system parameters Θ.

2.3.3.1 Training Stage 1 - Learning a Competitive Objective

The first stage corresponds to learning the unknown system parameter Θ. However, as formulated in (Equation 2.8), we do not seek to estimate the true values of the channel matrix $\boldsymbol{H}$ and noise covariance $\boldsymbol{C}$, but rather learn the surrogate values which will facilitate accurate detection using the relaxed MLE formulation. We do this by taking advantage of two propertities of LoRD-Net: The first is the differentiability of the unfolded architecture with respect to $\boldsymbol{\Theta}$, which facilitates gradient-based optimization optimization; The second is the fact that for $\boldsymbol{G}_i = \delta \boldsymbol{I}$, LoRD-Net essentially implements L steps of gradient descent with step size δ over the convex objective (Equation 2.5), and is thus expected to reach its maxima.

Based on the above properties, we fix a relatively large number of layers/iterations L for this training stage, and fix the weights $\boldsymbol{\phi}$ to $\boldsymbol{G}_i = \delta \boldsymbol{I}$. Under this setting, the output of LoRD-

Net $\mathcal{G}^{L}_{\boldsymbol{\phi}=\{\delta \boldsymbol{I}\}}(\boldsymbol{x};\Theta,\boldsymbol{r})$ represents an approximation of the relaxed MLE for a given parameter Θ, denoted $\bar{\boldsymbol{x}}_{\Theta}(\boldsymbol{r})$, i.e., we have that

$$\bar{\boldsymbol{x}}_{\Theta}(\boldsymbol{r}) \approx \mathcal{G}^{L}_{\boldsymbol{\phi}=\{\delta \boldsymbol{I}\}}(\boldsymbol{x}_0;\Theta,\boldsymbol{r}). \tag{2.14}$$

We refer to the setting $\boldsymbol{\phi}=\{\delta \boldsymbol{I}\}$ using in this stage as the *basic* optimization policy. Note that as the number of layers grows large, the above approximation becomes more accurate. Hence, by substituting (Equation 2.14) into (Equation 2.8) and replacing $\bar{\boldsymbol{x}}_{\Theta}(\boldsymbol{r}^i_p)$ with the corresponding outputs of LoRD-Net, we formulate the loss measure of the first training stage of LoRD-Net as:

$$\Theta^{\star} = \underset{\Theta}{\operatorname{argmin}} \ \frac{1}{B}\sum_{i=0}^{B-1} \left\| \mathcal{G}^{L}_{\boldsymbol{\phi}=\{\delta \boldsymbol{I}\}}(\boldsymbol{x}_0;\Theta,\boldsymbol{r}^i_p) - \boldsymbol{x}^i_p \right\|^2_2. \tag{2.15}$$

Owing to the differentiable nature of $\mathcal{G}^{L}_{\boldsymbol{\phi}}(\boldsymbol{x}_0;\Theta,\boldsymbol{r})$ with respect to Θ, we recover $\Theta^{\star}$ based on (Equation 2.15) using conventional gradient-based training, e.g., stochastic gradient descent with backpropagation, as detailed in our numerical evaluations description in Section 2.4

2.3.3.2 Training Stage 2 - Learning the Unfolded Weights

Having learned the unknown system parameters Θ in Stage 1, we turn to tuning the parameters of LoRD-Net, i.e., the set $\boldsymbol{\phi}=\{\boldsymbol{G}_i\}$. We note that in Stage 1, the rationale was to use the basic optimization policy $\boldsymbol{\phi}=\{\boldsymbol{G}_i=\delta \boldsymbol{I}\}_{i=0}^{L-1}$ with a large number of layers L, exploiting the insight that under this setting, LoRD-Net effectively implements conventional gradient descent. However, once Stage 1 is concluded and $\Theta^{\star}$ is learned, it is preferable to reduce the number of layers L compared to that used in Stage 1, thus exploiting the ability of the unfolded network

to carry out faster inference compared to their model-based iterative counterparts by learning the weights applied in each iteration [76, 82]. Consequently, the first step in this stage is to set a number of layers to a value which can potentially be smaller than that used in the first training stage, and then optimize the weights according to the following criterion:

$$\boldsymbol{\phi}^\star = \underset{\boldsymbol{\phi}}{\operatorname{argmin}} \frac{1}{B} \sum_{i=0}^{B-1} \left\| \mathcal{G}^L_{\boldsymbol{\phi}=\{\boldsymbol{G}_l\}_{l=1}^L}(\boldsymbol{x}_0; \Theta^\star, \boldsymbol{r}_p^i) - \boldsymbol{x}_p^i \right\|_2^2. \tag{2.16}$$

Generally speaking, in order for a first-order optimizer (LoRD-Net in this case) to provide a descent direction at each iteration (layer), the pre-conditioning matrices must be positive-semidefinite so that each iteration does not reverse the gradient direction. To incorporate this requirement into LoRD-Net training, we re-parameterize the pre-conditioning matrices by writing $\{\boldsymbol{G}_i = \boldsymbol{W}_i \boldsymbol{W}_i^T\}$ and performing the traning over the matrices $\{\boldsymbol{W}_i\}$. The resulting two-stage training algorithm is summarized as Algorithm 2.1.

Algorithm 2.1 Training LoRD-Net

Input: Labeled data $\{\boldsymbol{x}_p^b, \boldsymbol{r}_p^b\}_{b=0}^B$
Stage 1 Init: Fix (large) L, step-size $\delta \in (0, 1)$, and weights $\boldsymbol{G}_l = \delta \boldsymbol{I}$. Initialize $\boldsymbol{x}_0$;
-Optimize $\Theta^\star$ via (Equation 2.15)
Stage 2 Init: Fix (small) L. Initialize $\boldsymbol{x}_0$;
-Set the trainable parameters to $\{\boldsymbol{G}_i = \boldsymbol{W}_i \boldsymbol{W}_i^T\}$;
-Optimize $\boldsymbol{\phi}^\star$ according to (Equation 2.16)
Output: LoRD-Net parameters $\{\Theta^\star, \boldsymbol{\phi}^\star\}$

When the network is properly trained, LoRD-Net is expected to carry out learned and accelerated first-order optimization, tuned to operate even in channel conditions for which such an approach does not yield the MLE for the true channel.

2.3.4 Discussion

LoRD-Net is a data-driven acquisition system based on unfolding first-order gradient optimization methods, designed for low-resolution MIMO receivers operating without analog processing. Its model-awareness enables the receiver to learn to accurately infer from smaller training sets compared to conventional DNN architectures applied to such setups, as suggested, e.g., in [49], giving rise to the possibility of tracking block-fading channel conditions via online training, as in [67]. Furthermore, LoRD-Net differs from previously proposed deep unfolded MIMO receivers as surveyed in [73] in two key aspects: First, LoRD-Net is particularly designed for one-bit observations, being derived from the iterative optimization formulation which arises from such setups. Second, previous unfolded MIMO receivers either assumed prior knowledge of the channel parameters, as in [71], or alternatively, utilize external modules to directly estimate the CSI as in [70]. LoRD-Net exploits the fact that, for its unfolded relaxed convex optimization algorithm to yield the desired MLE, an alternative channel parameters, which differ from the true Θ, should be estimated. Consequently, the training procedure of LoRD-Net does not aim to recover the true CSI, but the one which yields a competitive objective which facilitates symbol detection, thus accounting for the overall system task.

The proposed training procedure detailed in Algorithm 2.1 carries out each training stage once in a sequential manner. This strategy can be extended to optimizing the hyperparameters

and the weights in an alternating fashion, i.e. repeating the stages multiple times, while using the learned $\boldsymbol{\phi}$ in Stage 2 in the Stage 1 that follows. Alternatively, the hyperparameters and the weights can be learned jointly in an end-to-end manner, by optimizing (Equation 2.16) with respect to both Θ and $\boldsymbol{\phi}$ simultaneously. The main requirement for carrying out these training strategies compared to that detailed in Subsection 2.3.3 is that the same number of layers L should be used when learning both Θ and $\boldsymbol{\phi}$, while when these stages are carried out once sequentially, it is preferable to use large L at Stage 1 and a smaller value, which dictates the number of learned weights, in Stage 2. Furthermore, our numerical evaluations show that training once in a two-stage fashion via Algorithm 2.1 yields similar and sometimes even improved performance compared to learning both Θ and $\boldsymbol{\phi}$ simultaneously in a one-stage manner, as well as when alternating between these two stages, as demonstrated in Section 2.4.

A possible extension of the training procedure is to account for ADCs with more than one bit, as well as allow LoRD-Net to optimize the quantization thresholds $\boldsymbol{b}$ in light of the overall symbol recovery task. While accounting for multi-level ADCs is a rather simple extension achieved by reformulating the objective function (Equation 2.3), optimizing the quantization thresholds requires modifying the overall training strategy. The challenge here is that modifying $\boldsymbol{b}$ results in different one-bit measurements $\boldsymbol{r}$. In a communication setup, in which periodic pilots are transmitted, one can envision gradual optimization of $\boldsymbol{b}$ between consecutive pilot sequences, using their corresponding one-bit observations to further optimize LoRD-Net. The study of LoRD-Net with multi-level ADCs and optimized thresholds is left for future work.

2.4 Numerical Study

In this section, we numerically evaluate LoRD-Net[1], and compare its performance with state-of-the-art model-based and data-driven methodologies. As a motivating application for the proposed LoRD-Net, we focus on the evaluation of LoRD-Net for blind symbol detection task in one-bit MIMO wireless communications. In the following, we first detail the considered one-bit MIMO simulation settings in Subsection 2.4.1, after which we evaluate the receiver performance, compare LoRD-Net to alternative unfolded architectures, and numerically investigate its training procedure in Subsections 2.4.2, 2.4.3, and 2.4.4, respectively. .

2.4.1 Simulation Setting

We consider an up-link one-bit multi-user MIMO scenario as in (Equation 5.2). We focus on a single cell in which a base station (BS) equipped with m antenna elements serves n single-antenna users. Specifically, we consider two cases of $(m, n) = (128, 16)$ and $(m, n) = (64, 10)$, i.e., a 128×16 and a 64×10 MIMO channel setup. The transmitted symbols of the users, represented by the unknown vector $\boldsymbol{x}$, are randomized in an independent and identically distributed (i.i.d.) fashion from a BPSK constellation set $\mathcal{M} = \{-1, +1\}$. The projection mapping is thus $\mathcal{P}_{\mathcal{M}^n}(\boldsymbol{x}) = \text{sign}(\boldsymbol{x})$, where the sign function is applied element-wise on the vector argument. In the sequel, we assume that while the channel matrix $\boldsymbol{H}$, representing the CSI, is not available at the BS, the noise statistics $\boldsymbol{C}$ are known and are fixed to $\boldsymbol{C} = \boldsymbol{I}$. Accordingly, our goal is to utilize LoRD-Net to recover the transmitted symbols from the one-bit measurements. Note that the

[1]The source code is available at: `https://github.com/skhobahi/LoRD-Net`.

proposed methodology can carry out the task of symbol detection even for the case in which the noise statistics $\boldsymbol{C}$ is unknown.

Channel Models: We evaluate LoRD-Net under two channel models: *(i)* i.i.d. Rayleigh fading channels, where $\boldsymbol{H} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$; and *(ii)* the COST-2100 massive MIMO channel [83]. The COST-2100 channel model is a realistic geometry-based stochastic model which accounts for prominent characteristics of massive MIMO channels, and is considered to be an established benchmark for evaluating MIMO communication systems. We generate the channel matrices for the COST-2100 model for a narrow-band indoor scenario with closely-spaced users at 2.6 GHz band, where the BS is equipped with a uniform linear array (ULA) that has m omni-directional receive antenna elements. The one-bit ADC operation uses zero thresholds, i.e. $\boldsymbol{b} = \boldsymbol{0}$. We define the signal-to-noise ratio (SNR) as:

$$\text{SNR} = \mathbb{E}\left\{\|\boldsymbol{H}\boldsymbol{x}\|_2^2\right\} / \mathbb{E}\left\{\|\boldsymbol{n}\|_2^2\right\}. \tag{2.17}$$

Benchmark Algorithms: As LoRD-Net combines both model-based and data-driven inference, we compare its performance with state-of-the-art model-based and data-driven methodologies in a one-bit MIMO receiver scenario. In particular, we use the following benchmarking detection algorithms:

- The model-based nML proposed in [55]. The nML algorithm is based on a convex relaxation of the conventional ML estimator, and requires the exact knowledge of the channel parameters $\Theta = \{\boldsymbol{H}, \boldsymbol{C}\}$. We set the number of iterations of the nML algorithm to 700,

and the step-size is chosen using a grid search method to further improve the performance of the nML, while the remaining parameters are those reported in [55].

- The data-driven Deep Soft Interference Cancellation (DeepSIC) methodology proposed in [68], with five learned interference cancellation iterations. DeepSIC is channel-model-agnostic and can be utilized for symbol detection in non-linear settings such as low-resolution quantization setups. Unlike LoRD-Net, which is designed particularly for observations of the form (Equation 5.2) where $\Theta = \{\boldsymbol{H}, \boldsymbol{C}\}$ is unknown, DeepSIC has no prior knowledge of neither the channel model nor its parameters.

LoRD-Net Setting: The LoRD-Net receiver is implemented with $L = 30$ layers. Recall that the first training stage of the LoRD-Net is concerned with finding a competitive objective by carrying out the training of the network over the unknown set of channel parameters $\Theta = \{\boldsymbol{H}, \boldsymbol{C}\}$. Unless otherwise specified, we focus on the case where only $\boldsymbol{H}$ is unknown, and the correlation matrix of the noise $\boldsymbol{C}$ is available.

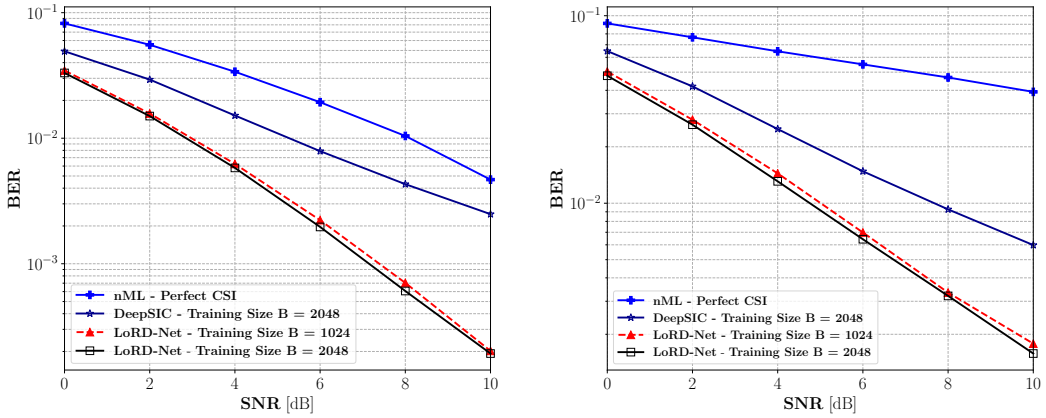
During the first training stage, we set $\delta = 0.01$, and recover $\Theta^\star$ based on the objective (Equation 2.15) using the Adam stochastic optimizer [84] with a constant learning rate of 10^{-3}. Next, we carry out the training of the LoRD-Net during the second stage according to the objective function defined in (Equation 2.16) and over the set of trainable parameters $\boldsymbol{\phi}$, using the Adam optimizer with a learning rate of 10^{-4}, and a mini-batch of size 512. We consider the learning of diagonal pre-conditioning matrices (unfolded weights) during the second training stage. The network is trained for 400 epochs during the first training stage, and 400 epochs during the second training stage, with the same value of $L = 30$ used in both stages.

2.4.2 Receiver Performance

Here, we evaluate the performance of the proposed LoRD-Net, comparing it to the aforementioned benchmarks as well as examining its dependence on the number of training samples B. In particular, we numerically evaluate the bit-error-rate (BER) performance versus SNR using different training sizes $B \in \{1024, 2048\}$, for both 128×16 and 64×10 channel configurations. For DeepSIC, we use only $B = 2048$, while the nML recever of [55] operates with perfect CSI, i.e., with full accurate knowledge of Θ. All data-driven receivers are trained for each SNR separately, using a dataset corresponding to that specific SNR value.

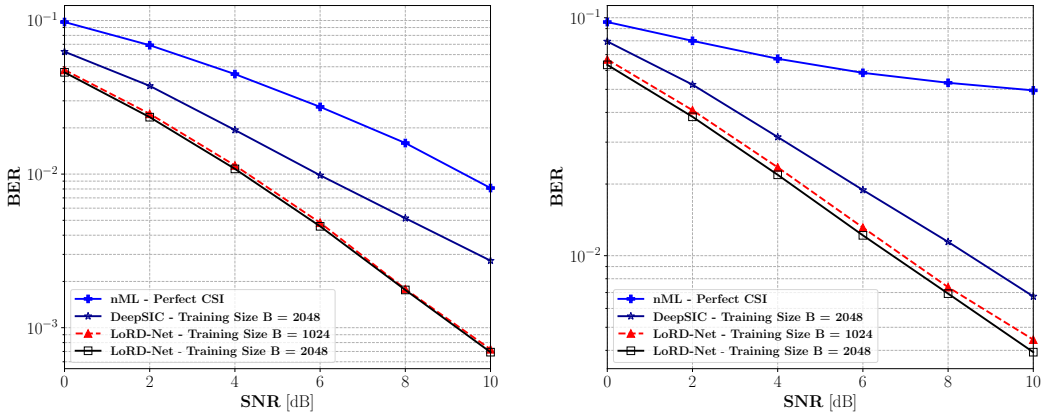
The results are depicted in 4(a) and 4(b) for a 128×16 channel configuration under the Rayleigh fading and COST-2100 channel models, respectively. Furthermore, the BER performance for a 64×10 configuration under both channel models are illustrated in 5(a) for the Rayleigh fading channel, and in 5(b), for the COST-2100 channel model. Based on the results presented in Figure 4 and Figure 5, one can observe that LoRD-Net significantly outperforms the competing model-based and data-driven algorithms and achieves improved detection performance under both simulated channels, as well as both MIMO configurations.

In particular, the nML algorithm, which is designed to iteratively approach the MLE using ideal CSI (prior knowledge of the channel matrix), is notably outperformed by LoRD-Net. Such gains by LoRD-Net, which learns to compute the MLE from data without requiring CSI, compared to the model-based nML algorithm, demonstrate the benefits of learning a competitive objective function combined with a relaxed deep unfolded optimization process. Specifically, the results depicted in Figure 4-Figure 5 illustrate that one can significantly improve the receiver



(a) Rayleigh fading (b) COST-2100 massive MIMO

Figure 4. BER performance versus SNR over a 128×16 channel configuration.



(a) Rayleigh fading (b) COST-2100 massive MIMO

Figure 5. BER performance versus SNR over a 64×10 channel configuration.

performance by learning a new channel matrix $\boldsymbol{H}$ upon which the learned competitive objective function admits optimal points near the true symbols. The learning of the competitive objective function is possible due to the hybrid model-based/data-driven nature of LoRD-Net, and the

fact that it is derived based on the unfolding of first-order optimization techniques. From a computational complexity point-of-view, the depicted performance of the nML algorithm in Figure 4-Figure 5 is achieved by employing 700 iterations of a first-order optimization algorithm, while LoRD-Net uses only $L = 30$ layers/iterations—exhibiting a significant reduction in the computational cost during inference as compared to the nML algorithm.

Comparing LoRD-Net to DeepSIC illustrates that LoRD-Net benefits considerably from its model-aware architecture. The fact that LoRD-Net is particularly tailored to the one-bit system model of (Equation 5.2) allows it to achieve improved accuracy, even in the case of training with small amounts of data. For instance, for the 128×16 MIMO Rayleigh fading channel (see 4(a)), LoRD-Net trained with $B = 2048$ samples, achieves BER of 10^{-2} at SNR of 3dB, while DeepSIC trained with the same dataset requires SNR as high as 5dB to achieve such an error rate. Considering 4(b), a similar behavior is observed in the COST-2100 channel, for a BER of 3×10^{-2}. A similar performance gain for LoRD-Net can be observed in a 64×10 configuration; see Figure 5. Furthermore, it can be observed that the LoRD-Net still outperforms the DeepSIC methodology, even when trained on 2 times less training samples. In particular, for the (128×16) channel setup considered in this part, the total number of trainable parameters of LoRD-Net is merely $|\Theta = \{\boldsymbol{H}\}| + |\boldsymbol{\phi}| = n\,(L + m) = 2528$. For comparison, DeepSIC, which uses and trains a multi-layer fully-connected network for each user at each interference cancellation iterations, consists here of over 8×10^5 trainable parameters. Such a reduction in the number of parameters allows for achieving substantially improved performance with much smaller training points, as observed in Figure 4-Figure 5. Finally, we note that the small number of trainable parameters

of LoRD-Net shows its potential for online or real-time training, as proposed in [67]. This can be achieved by using periodic pilots with minimal overhead on the communication, while inducing a relatively low computational burden in its periodic retraining.

So far, we have investigated the performance of the proposed LoRD-Net for scenarios with known noise statistics, and unknown $\boldsymbol{H}$ (i.e., $\Theta = \{\boldsymbol{H}\}$). Next, we investigate the detection performance of LoRD-Net when both the channel and noise covariance matrices are not available, i.e., we set $\Theta = \{\boldsymbol{H}, \boldsymbol{C}\}$ and carry out the training according to the proposed two stage methodology. Specifically, we consider the learning of a diagonally structured $\boldsymbol{C}$ in addition to the channel matrix $\boldsymbol{H}$ for this scenario. Figure 6 demonstrates the BER versus SNR performance of LoRD-Net under both channel models, when trained using a dataset of size $B = 1024$. The performance of LoRD-Net for the case of $\Theta = \{\boldsymbol{H}\}$ is further provided for comparison purposes. Observing Figure 6, one can readily conclude that the proposed network can successfully perform the task of symbol detection also when $\boldsymbol{C}$ in unknown. Furthermore, it can be observed that a small gain in performance is achieved for both channel models when $\Theta = \{\boldsymbol{H}, \boldsymbol{C}\}$ as compared to the case of $\Theta = \{\boldsymbol{H}\}$, which is presumably due to the careful addition of more degrees of freedom in learning a competitive surrogate model.

2.4.3 Performance of Competing Deep Unfolded Architectures

In this part, we compare the performance of the proposed LoRD-Net with alternative deep unfolding-based architectures tailored for the problem at hand. Recall that the architecture of LoRD-Net uses trainable parameters which are shared among the different layers, as illustrated in Figure 3. Thus, LoRD-Net is comprised of a relatively small number of trainable param-

eters, and uses a two-stage learning method to train the shared parameters, representing the competitive model, and the iteration-specific weights, encapsulating the first-order optimization coefficient. Nonetheless, the conventional approach for unfolding first-order optimization techniques is to over-parameterize the iterations, and then, train in an end-to-end manner using a one-stage training procedure discussed earlier. Therefore, to numerically evaluate the proposed unfolding mechanism of LoRD-Net, we next compare it to two conventional unfolding based benchmarks derived from the relaxed MLE:

- *Benchmark 1*: An over-parameterized deep unfolded architecture obtained by setting the computational dynamics for the ith layer as:

$$\bar{g}_{\phi_i}(\boldsymbol{x}_i; \boldsymbol{r}) = \boldsymbol{x}_i - \boldsymbol{A}_i \boldsymbol{R}\, \boldsymbol{\eta}\left(\boldsymbol{R}\left(\boldsymbol{b} - \boldsymbol{B}_i \boldsymbol{x}_i\right)\right). \tag{2.18}$$

  Here, $\phi_i = \{\boldsymbol{A}_i \in \mathbb{R}^{n\times m}, \boldsymbol{B}_i \in \mathbb{R}^{m\times n}\}$ are the trainable parameters of the ith layer, and $\boldsymbol{R} = \mathrm{Diag}(\boldsymbol{r})$.

- *Benchmark 2*: Here, we again use the unfolded architecture given in (Equation 2.18), while limiting the number of trainable parameters by constraining the rank of the learned matrices. In particular, we set $\boldsymbol{A}_i = \boldsymbol{P}_i \boldsymbol{Q}_i$ and $\boldsymbol{B}_i = \boldsymbol{R}_i \boldsymbol{S}_i$, where $\phi_i = \{\boldsymbol{P}_i \in \mathbb{R}^{n\times r}, \boldsymbol{Q}_i \in \mathbb{R}^{r\times m}, \boldsymbol{R}_i \in \mathbb{R}^{m\times r}, \boldsymbol{S}_i \in \mathbb{R}^{r\times n}\}$ denotes the set of trainable parameters of the ith layer of the unfolded network. The dimension $r < \min(m, n)$ controls the rank of the resulting weight matrices $\{\boldsymbol{A}_i, \boldsymbol{B}_i\}$, and thus the number of trainable parameters.

Comparing (Equation 2.18) with the corresponding dynamics of LoRD-Net in (Equation 2.10), we note that the channel matrix $\boldsymbol{H}$, the pre-conditioning matrices $\boldsymbol{G}_i$, and the noise covariance matrix $\boldsymbol{C}$ are now absorbed into the *per-layer* trainable matrices $\boldsymbol{A}_i$ and $\boldsymbol{B}_i$. Accordingly, these unfolded benchmarks, which follow the conventional approach for unfolding optimization algorithms, are less faithful to the underlying model. These benchmarks also differ from LoRD-Net in their number of trainable parameters. In particular, Benchmark 1 with L layers has $2Lnm$ trainable parameters, while Benchmark 2 has $2Lr(m+n)$ weights, which can be controlled by the setting of the hyperparameter r. For comparison, LoRD-Net has $n(L+m)$ trainable parameters for the case of $\Theta = \{\boldsymbol{H}\}$ and diagonally structured pre-conditioning matrices, while for the case of $\Theta = \{\boldsymbol{H}, \boldsymbol{C}\}$ with a diagonally structured pre-conditioning matrix and noise covariance matrix it has $n(L+m)+m$ trainable parameters.

We evaluate the performance of the proposed LoRD-Net compared to the unfolded benchmarks in the following simulation setup. We consider train all the considered network using a dataset of size $B = 1024$, while the highly-parameterized Benchmark 1 is also trained using $B = 2048$ samples. For Benchmark 2, we set $r = 1$. All architectures have $L = 30$ layers and their performance are evaluated on the same testing dataset of size $B = 2048$. The unfolded benchmarks are trained in the conventional end-to-end fashion. The channel model is a (128×16) Rayleigh fading channel. Foror the considered scenario above, the LoRD-Net admits a total of 2528 trainable parameters, while Benchmark 1 ha a total of 122880 (approximately 50 times more parameters than LoRD-Net), while Benchmark 2 has 8640 trainable parameters.

Fig Figure 7 depicts the BER versus SNR of LoRD-Net compared to the unfolded benchmarks. We observe in Figure 7 that the proposed LoRD-Net significantly outperforms the conventional unfolding based benchmarks,indicating the gains of the increased level of domain knowledge Incorporated in to the architecture of LoRD-Net and its two stage training procedure. It is also observed that the performance of Benchmark 1 increases with more training samples. Interestingly, for a small training set of $B = 1024$ samples, Benchmark 2, which is obtained by imposing a rank constraint on the trainable parameters of Benchmark 1, achieves improved performance over Benchmark 1, due to its notable reduction in the number of trainable parameters.

2.4.4 Training Analysis

In this part, we analyze the performance of the proposed two-stage training procedure described in Subsection 2.3.3. The training aspects of LoRD-Net are numerically evaluated for the 128×16 Rayleigh channel model detailed before.

Following our insight on the ability of LoRD-Net to accurately train with small datasets, we begin by evaluating the performance of the LoRD-Net versus the training data size B. For this study, we generate training datasets of size $B \in \{32, 64, 128, 256, 512, 1024, 2048\}$ and evaluate the performance of LoRD-Net using 2048 test samples. Figure 8 depicts the BER achieved for each training size B, for SNR $\in \{0, 2, 4, 6, 8, 10\}$ dB. We can observe from Figure 8 that the performance of the LoRD-Net improves across all SNR values, where the improvements are most notable for $B \leq 256$. Interestingly, it may be concluded from Figure 8 that LoRD-Net is capable of accurately and reliably performing the task of symbol detection without CSI with

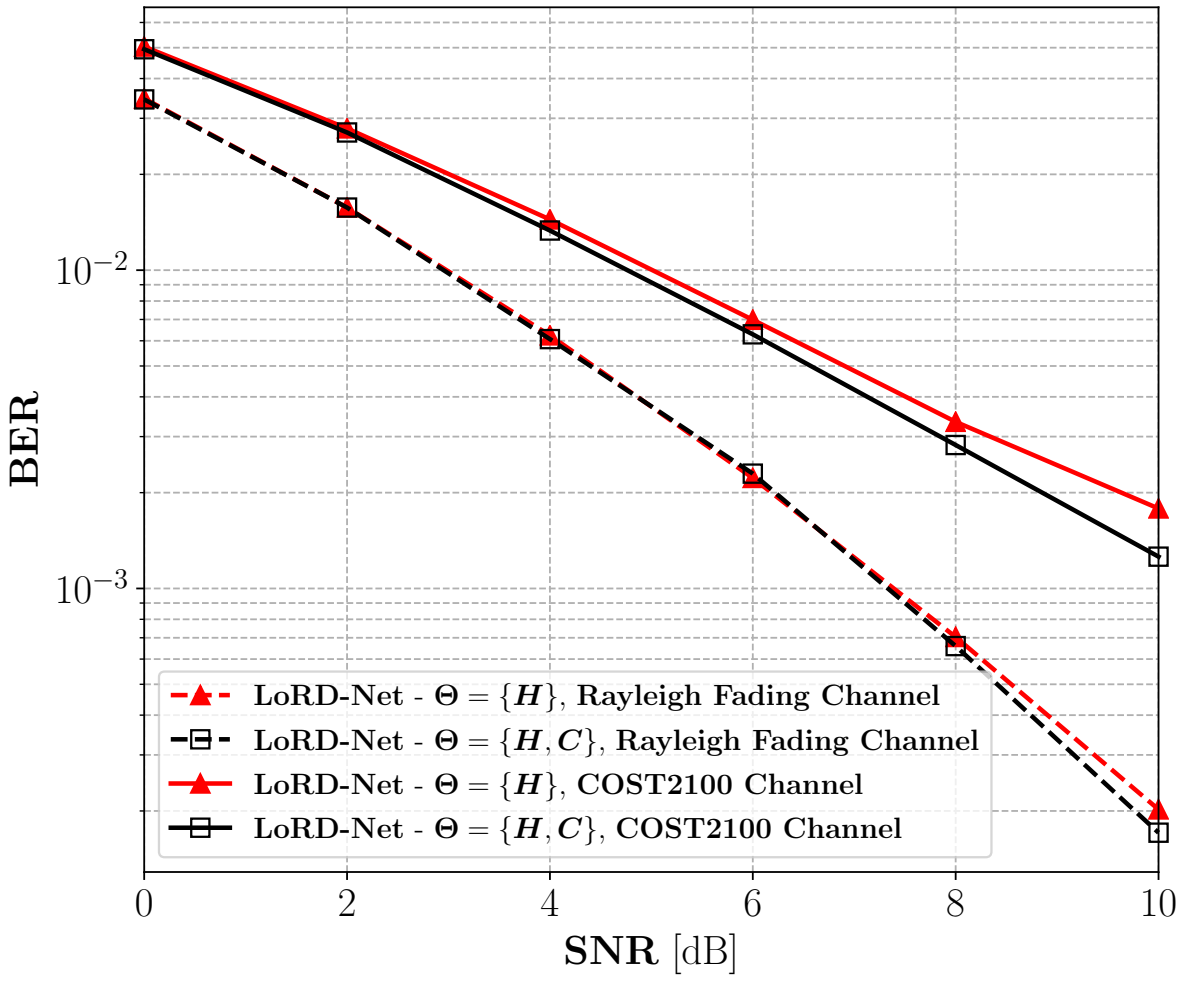


Figure 6. BER versus SNR for both channel models and a training size of $B = 1024$. The performance of the proposed LoRD-Net is provided for both scenarios of training over $\Theta = \{\boldsymbol{H}\}$ (i.e., known noise statistics $\boldsymbol{C}$), and over $\Theta = \{\boldsymbol{H}, \boldsymbol{C}\}$ corresponding to unknown channel matrix and noise statistics.

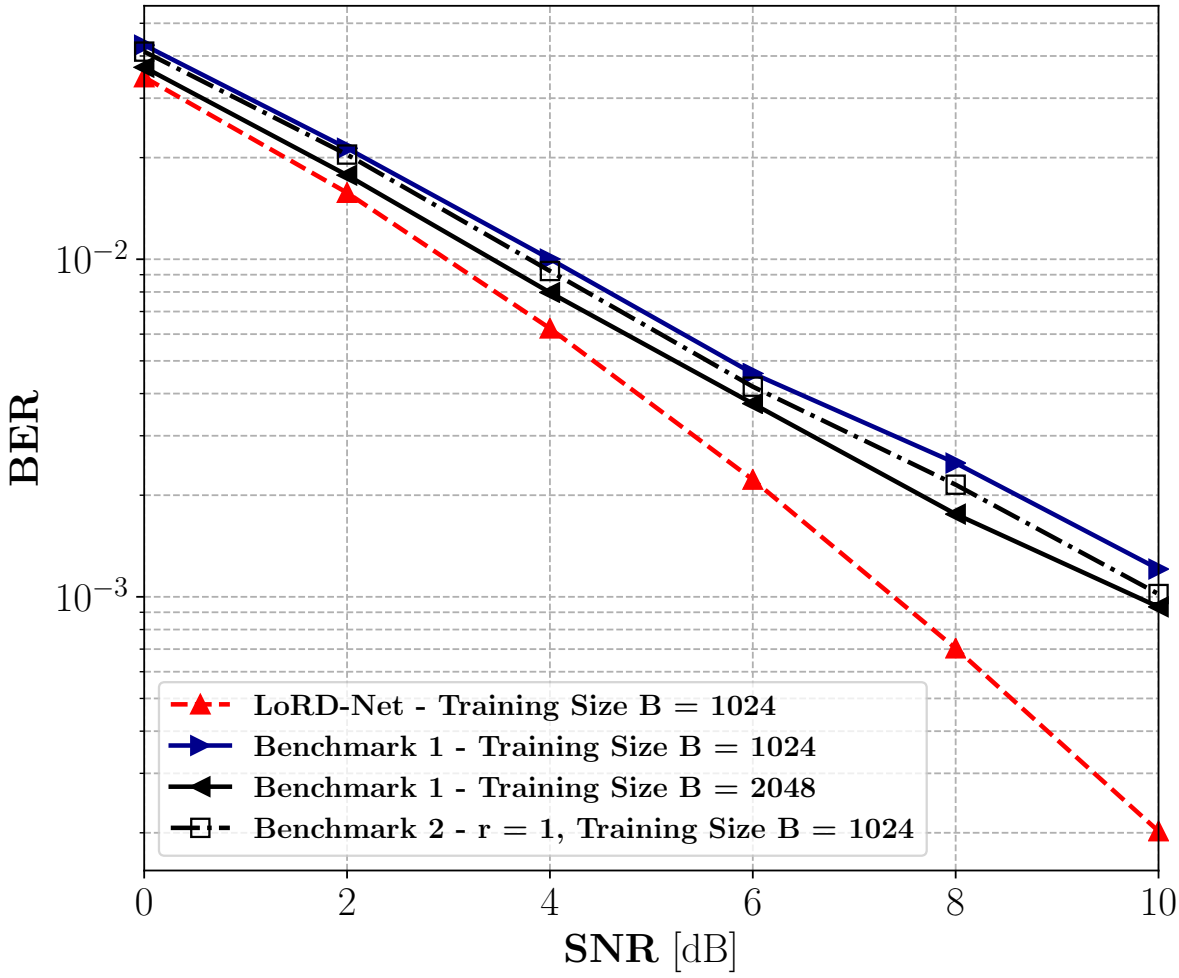


Figure 7. BER versus SNR of LoRD-Net compared to the unfolded benchmark for a (128×16) Rayleigh fading channel model.

as few as $B = 512$ samples. The ability of LoRD-Net to train with very few training samples (compared to the black-box DNN models for one-bit MIMO receivers [47, 49], as well as the DeepSIC architecture), stems from its incorporation of the domain-knowledge in designing the LoRD-Net architecture. This in turn leads to far fewer trainable parameters requiring much less training samples for optimizing the network.

Next, we analyze the performance and the effect of the two stage training methodology detailed in Algorithm 2.1 on the detection performance of the LoRD-Net architecture. Recall that the first training stage is concerned with finding a competitive objective function through an optimization of LoRD-Net over the unknown system parameters Θ, while the second training stage tunes the positive definite preconditioning matrices $\boldsymbol{\phi} = \{\boldsymbol{G}_i\}$ to accelerate the convergence of the LoRD-Net to the optimal points of the obtained competitive objective. To numerically evaluate the performance of the training methodology, we set SNR = 8 dB, and generate a training dataset of size $B = 512$ and a testing dataset of size 2048. Then, we compare performance of Algorithm 2.1 with two other competing training procedures:

• *One-Stage Training*: Here, the weights $\boldsymbol{\phi}$ and the unknown system parameters Θ are jointly learned in a single stage. The objective of this one stage training procedure for LoRD-Net is

$$\min_{\boldsymbol{\phi}=\{\boldsymbol{G}_l\}_l, \Theta \in \boldsymbol{\Theta}} \ \frac{1}{B} \sum_{i=0}^{B-1} \left\| \mathcal{G}_{\boldsymbol{\phi}}^{L}(\boldsymbol{x}_0; \Theta, \boldsymbol{r}_p^i) - \boldsymbol{x}_p^i \right\|_2^2 . \tag{2.19}$$

• *Alternating Training*: This procedure is concerned with training the network by alternating between the two optimization problems (Equation 2.15) and (Equation 2.16) consecutively with

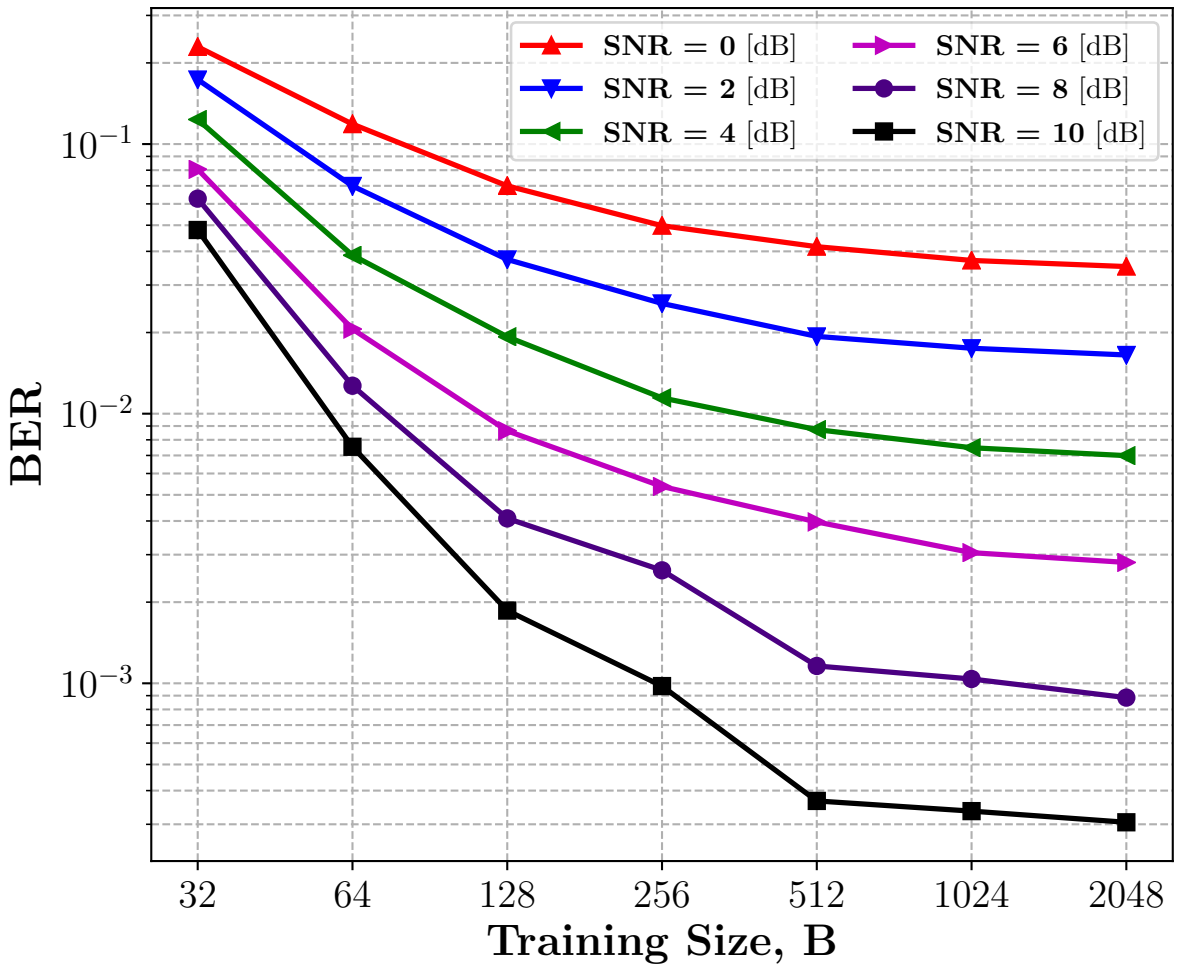


Figure 8. BER versus training size B for the Rayleigh fading channel.

respect to each training epoch. Here, the network is trained over 400 alternations, corresponding to a total of 800 training epochs. Namely, at each epoch index i, we update the variables Θ for odd i and update $\boldsymbol{\phi}$ for even i.

Figure 9 depicts the BER versus the training epoch for both the training and testing dataset. We first note that the proposed two-stage training method (Algorithm 2.1) outperforms the competing procedures, yielding lower testing error values. Interestingly, we observe that the proposed methodology successfully closes the generalization gap as the testing and training error are very close to each other. On the other hand, the other two training procedures admit relatively large generalization gaps, indicating the fact that their utilization has resulted in an over-fitting of the network to the data. Furthermore, it can be observed from Figure 9 that the major improvement of the detection accuracy of LoRD-Net is taking place during the first training stage when finding a competitive objective function, where a slight improvement in the BER is achieved during the second training stage, starting at epoch index $80(\times 5)$.

Before we proceed with the evaluation results, we provide some useful connections to notions widely used in the deep learning literature. Generally speaking, the performance of a statistical learning framework and its training procedure is evaluated using its generalization gap and testing error. The *generalization gap* of a model can be defined as the difference between the training and testing errors. Specifically, a model with smaller generalization gap and smaller testing error is highly favourable. Furthermore, a higher generalization gap may indicate that the network has over-fitted to the data, and hence, it does not generalize well. For two models with the same generalization gap, the one with lower testing error is favourable. Figure 9 depicts

the BER versus the training epoch for both the training and testing dataset. We first note that the proposed two stage training method outperforms all other competing procedures and it assumes a significantly lower testing error as compared to other algorithms. Interestingly, one can observe that the proposed methodology has successfully closed the generalization gap as the testing and training error are very close to each other. On the other hand, the other two training procedures admits very large generalization gap indicating the fact that their utilization has resulted in an over-fitting of the network to the data. Furthermore, it can be observed from Figure 9 that the major improvement of the detection accuracy of the LoRD-Net is taking place during the first training stage when finding a competitive objective function, i.e., epochs $i < 80(\times 5)$, where a slight improvement in the BER is achieved during the second stage, i.e., $i \geq 80(\times 5)$.

The success of the proposed two stage training procedure in closing the generalization gap compared to the one stage training procedure is presumably due to the fact that the two-stage training approach leads to an *implicit regularization* on the model capacity limiting the total number of parameters used during the entire training procedure. On the contrary, the one stage training procedure allows the neural network to use its full capacity leading to an over-fitting and a larger generalization gap, as observed in Figure 9.

As discussed in Subsection 2.3.3, the second training stage allows LoRD-Net to achieve fast inference, i.e., accelerated convergence to the optimal points of the competitive objective function. To illustrate this behavior, we perform a per-layer BER evaluation of LoRD-Net, exploiting the interpretable model-based nature of the LoRD-Net, in which each layer represents

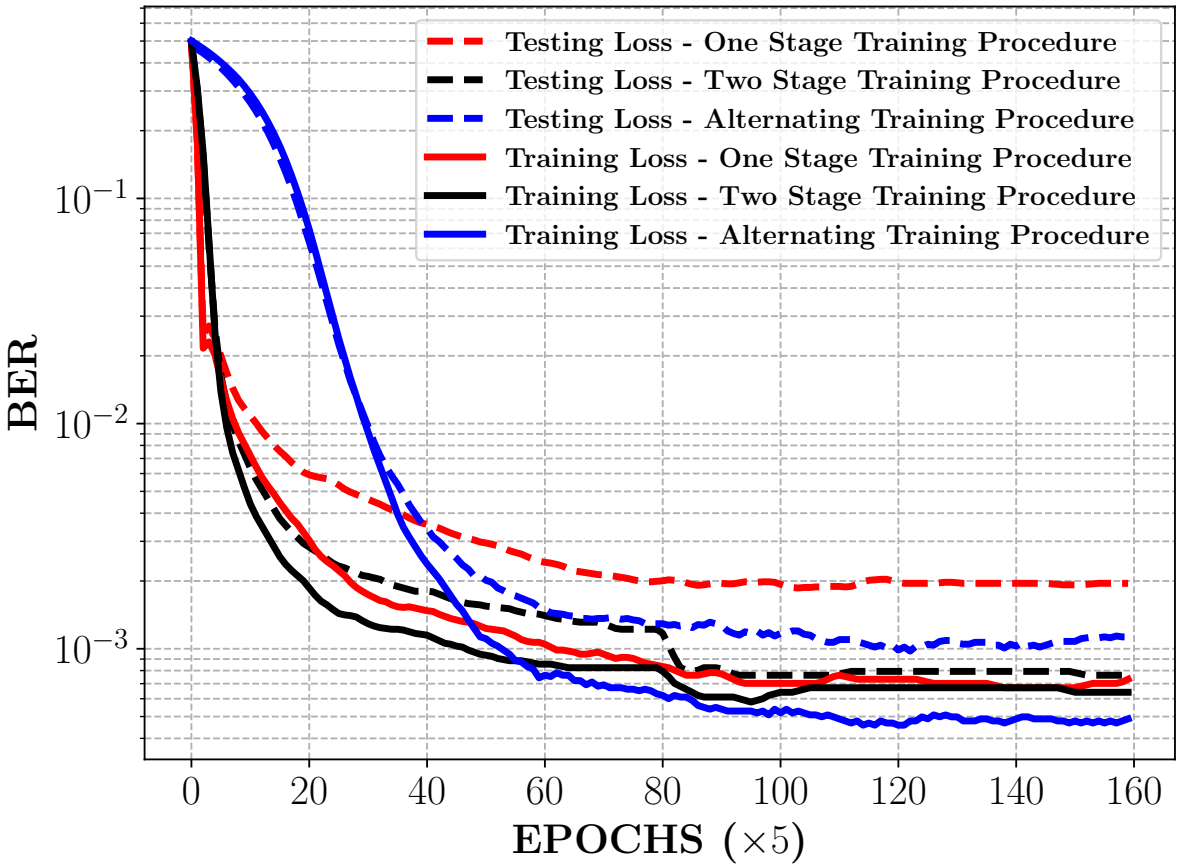
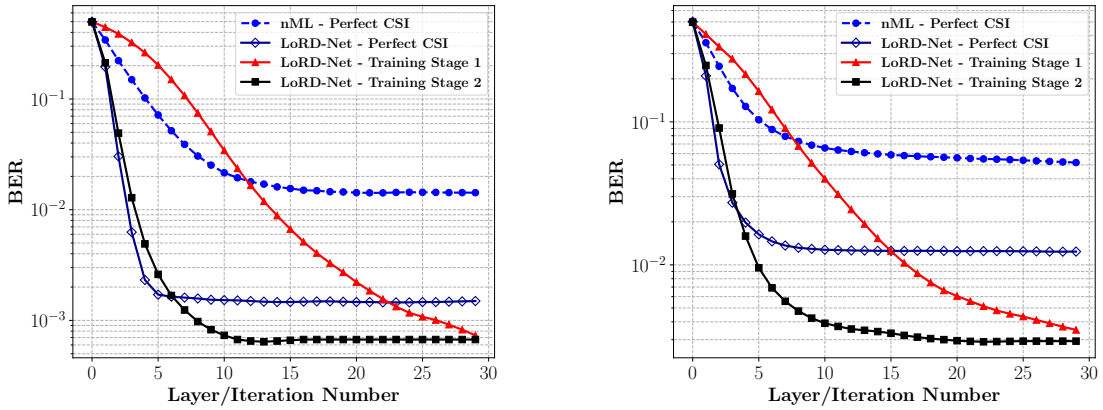


Figure 9. BER versus the training epoch number of LoRD-Net, Rayleigh fading channel, $SNR = 8$ dB.



(a) Rayleigh fading (b) COST-2100 Massive MIMO

Figure 10. BER performance of LoRD-Net after completing training stages 1 and 2 versus the layer/iteration number for (a) the Rayleigh fading channel, and (b) the COST-2100 massive MIMO channel, with $SNR = 8$ dB.

an unfolded first-order optimization iteration, and thus its output can be used as an estimate of the transmitted symbols. 10(a) and 10(b) depict the BER versus the layer/iteration number of LoRD-Net at the completion of training stages 1 and 2, for the Rayleigh fading channel and the COST-2100 channel model, respectively. We observe in Figure 10 that the convergence of LoRD-Net after the completion of the first training stage is slow and requires at least $L = 30$ layers/iterations to converge. Interestingly, we note from Figure 10 that the second training stage indeed results in an acceleration of the convergence of LoRD-Net via learning the best set of pre-conditioning matrices for the problem at hand in an end-to-end manner. In particular, after the completion of the second training stage, LoRD-Net can accurately and reliably recover the symbols with as few as 10 layers. This observation hints that one can consider further

truncation of the LoRD-Net after the training to reduce the computational complexity while maintaining its superior performance.

In order to quantify the quality of the learned competitive objective in closing the gap between the discrete optimization problem and its continuous version, we further provide the per-iteration performance of the nML algorithm and the LoRD-Net algorithm which operate with perfect CSI. For this scenario, LoRD-Net utilizes the true Θ, and is thus optimizer only over the weights $\boldsymbol{\phi}$ while employing the exact channel model $\boldsymbol{H}$. It is observed from 10(a)-10(b) that learning a new surrogate model for the continuous optimization problem at hand is indeed highly beneficial and admits a far superior performance in recovering the transmitted symbols. The analysis provided in Figure 10 further supports the rationale behind the proposed two-stage training methodology, and the fact that the second training stage results in an acceleration of the underlying first-order optimization solver (i.e., achieving a much faster descent per step) upon which the layers of the LoRD-Net are based.

2.5 Conclusion

In this work, we introduced LoRD-Net, which is a hybrid data-driven and model-based deep architecture for blind symbol detection from one-bit observations. The proposed methodology is based the unfolding of first-order optimization iterations for the recovery of the MLE. We proposed a two-stage training procedure incorporating the learning of a competitive objective function, for which the unfolded network yields an accurate recovery of the transmitted symbols from one-bit noisy measurements. In particular, owing to its model-based nature, LoRD-Net has far fewer trainable parameters compared to its data-driven counterparts, and can be trained

with very few training samples. Our numerical results demonstrate that the proposed LoRD-Net architecture outperforms the state-of-the-art model-based and data-driven symbol detectors in multi-user one-bit MIMO systems. We also numerically illustrate the benefits of the proposed two-stage training procedure, which allows to train with small training sets and infer quickly, due to its interpretable model-aware nature.

CHAPTER 3

DEEP SIGNAL RECOVERY WITH ONE-BIT QUANTIZATION

Overview: Machine learning, and more specifically deep learning, have shown remarkable performance in sensing, communications, and inference. In this paper, we consider the application of the deep unfolding technique in the problem of signal reconstruction from its one-bit noisy measurements. Namely, we propose a model-based machine learning method and unfold the iterations of an inference optimization algorithm into the layers of a deep neural network for one-bit signal recovery. The resulting network, which we refer to as *DeepRec*, can efficiently handle the recovery of high-dimensional signals from acquired one-bit noisy measurements. The proposed method results in an improvement in accuracy and computational efficiency with respect to the original framework as shown through numerical analysis.

Keywords: Deep learning, deep unfolding, MIMO communications, big data, machine learning, neural network, maximum likelihood, one-bit quantization

3.1 Introduction

Quantization of signals of interest is an integral part of all modern digital signal processing applications such as sensing, communication, and inference. In an ideal hardware implementation of a quantization system, a high-resolution analog-to-digital converter (ADC) with b-bit resolution and sampling frequency of f_s samples the original analog signal and maps the obtained

---

samples into a discrete state space of size $2^b f_s$. Generally, a large number of bits is required to obtain an accurate digital representation of the analog signal. In such a case, the quantization process has negligible impact on the performance of algorithms which were typically developed on the assumptions of infinite precision samples, and thus, the high-resolution (in terms of amplitude) quantization process can be directly modeled as an additive noise source. However, a crucial obstacle with modern ADCs is that their power consumption, manufacturing cost, and chip area grows exponentially with their resolution b [37, 85, 86].

The required high sampling data rate of ADCs used in next generataion communications systems is another obstacle that must be tackled in such applications. For instance, the promising millimeter wave (mmWave) multiple-input multiple output (MIMO) communication technology requires a very large bandwidth, and the corresponding sampling rate of the ADCs must increase accordingly. However, manufacturing ADCs with high-resolution (e.g., more than 8 bits) and high sampling rate are extremely costly and may not be available. Moreover, in other applications such as spectral sensing and cognitive radio, which require extremely high sampling rates, the cumulative cost and power consumption of using high-resolution and extremely fast ADCs are typically prohibitive and impractical. Hence, when signals across a wide frequency band are of interest, a fundamental trade-off between sampling rate, amplitude quantization precision, cost, and power consumption is encountered. An immediate solution to such challenges is to use low-resolution, and specifically *one-bit*, ADCs. The use of one-bit signed measurements obtained via one-bit ADCS enables a very high sampling rate while keeping the cost and power consumption very low. From a sampling viewpoint, the most extreme case of quantization is to

use only one bit per sample. More precisely, one-bit sampling can be seen as a process through which we repeatedly compare the amplitude of a signal (at each sample) to some reference threshold level and use only one bit to convey whether the signal amplitude resides above or below that threshold. Due to its appealing sampling properties, the problem of recovering a signal from its one-bit measurements has attracted a great deal of interest over the past few years [61, 87–90]. Therefore, it is vital to develop algorithms that can deal with low-resolution samples for different applications.

The fields of machine learning (ML), and more particularly deep learning, are impacting various fields of engineering and have recently attracted a great deal of attention in tackling long-standing signal processing problems. The advent of low-cost specialized powerful computing resources (e.g., GPUs, and more recently TPUs) and the continually increasing amount of massive data generated by the human population and machines, in conjunction with the new optimization and learning methods, have paved the way for deep neural networks (DNNs) and machine learning-based models to prove their effectiveness in many engineering areas (see, e.g., [7, 91, 92] and the references therein).

The main advantage of the deep learning-based model herein is that it employs several non-linear transformations to obtain an abstract representation of the underlying data. Model-based machine learning frameworks (e.g., probabilistic graphical models) incorporate prior knowledge of the system parameters into the inference process. A recent promising approach in bridging the gap between deep learning-based and model-based methods is the paradigm of *deep unfolding* [75]. Particularly, iterations of a conventional recursive algorithm, such as fast iterative soft

thresholding algorithm (FISTA), projected gradient descent, and approximate message passing (AMP), can be used as a baseline to design the architecture of a deep network with trainable parameters specifically customized to the problem of interest. Such a methodology results in an improvement in accuracy, and computational efficiency of the original framework. The deep unfolding method has already shown remarkable performance improvement in a wide range of applications such as MIMO communications [93, 94], multi-channel source separation [95], and sparse inverse problems [82, 96].

In this paper, we consider the general problem of high-dimensional signal recovery from random one-bit measurements. Specifically, we propose an efficient signal recovery framework based on the deep unfolding technique that has the advantage of low-complexity and near-optimal performance compared to traditional methods. Our proposed inference framework has a wide range of applications in the areas of wireless communications, detection and estimation, and sensing.

3.2 Problem Formulation

We begin by considering a general linear signal acquisition and one-bit quantization model with time-varying thresholds, described as follows:

$$\text{Signal Model:} \qquad \boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{n}, \tag{3.1}$$

$$\text{Quantization Model:} \qquad \boldsymbol{r} \triangleq \operatorname{sign}(\boldsymbol{y} - \boldsymbol{\tau}), \tag{3.2}$$

where $\boldsymbol{\tau} = [\tau_1, \ldots, \tau_M]^T$ denotes the vector of one-bit quantization thresholds, $\boldsymbol{y} \in \mathbb{R}^M$ denotes the received signal prior to quantization, $\boldsymbol{H} \in \mathbb{R}^{M \times N}$ denotes the sensing matrix, $\boldsymbol{x} \in \mathbb{R}^N$ denotes the multidimensional unknown vector to be recovered, and $\boldsymbol{n} \sim \mathcal{N}(0, \boldsymbol{C})$ denotes the zero-mean Gaussian noise with a known covariance matrix $\boldsymbol{C} = \mathbf{Diag}(\sigma_1^2, \ldots, \sigma_M^2)$. Furthermore, sign$(\cdot)$ denotes the signum function applied element-wise on the vector argument.

The above model covers a wide range of applications. For instance, the described model (Equation 6.1)-(Equation 6.2) can be used in MIMO communication systems in which $\boldsymbol{H}$ is the channel matrix, $\boldsymbol{x}$ is the signal sent by the transmitter, $\boldsymbol{n}$ is the additive Gaussian noise in the system, and the base station is equipped with one-bit ADCs, where the goal is to recover the transmitted symbols from $\boldsymbol{r}$.

3.2.1 Maximum Likelihood Estimator Derivation

Given the knowledge of the sensing matrix $\boldsymbol{H}$, noise covariance $\boldsymbol{C}$, and the corresponding quantization thresholds $\boldsymbol{\tau}$, our goal is to recover the original (likely high-dimensional) signal $\boldsymbol{x}$ from the one-bit random measurements $\boldsymbol{r}$. In such a scenario, each binary observation $\{r_i\}_{i=1}^N$ follows a Bernoulli distribution with parameter p_i, given by:

$$p_i = \text{Prob}\{\boldsymbol{h}_i^T \boldsymbol{x} + n_i - \tau_i > 0\} = Q\left(\frac{\tau_i - \boldsymbol{h}_i^T \boldsymbol{x}}{\sigma_i}\right), \tag{3.3}$$

where $Q(x) = 1 - \Phi(x)$ with $\Phi(x)$ representing the cumulative distribution function (cdf) of a standard Gaussian distribution and $\boldsymbol{h}_i^T$ denotes the i-th row of the matrix $\boldsymbol{H}$. In particular, the probability mass function (pmf) of each binary observation can be compactly expressed as:

$$p(r_i) = Q\left(\frac{r_i}{\sigma_i}\left(\tau_i - \boldsymbol{h}_i^T\boldsymbol{x}\right)\right), \tag{3.4}$$

where $r_i \in \{-1, +1\}$. Therefore, the log-likelihood of the quantized observations $\boldsymbol{r}$ given the unknown vector $\boldsymbol{x}$ can be expressed as:

$$\mathcal{L}(\boldsymbol{x}) = p(\boldsymbol{r}|\boldsymbol{x}) = \log\left\{\prod\nolimits_{i=1}^{N} Q\left(\frac{r_i}{\sigma_i}\left(\tau_i - \boldsymbol{h}_i^T\boldsymbol{x}\right)\right)\right\} \tag{3.5}$$

$$= \sum_{i=1}^{N} \log\left\{ Q\left(\frac{r_i}{\sigma_i}\left(\tau_i - \boldsymbol{h}_i^T\boldsymbol{x}\right)\right)\right\}, \tag{3.6}$$

where $\log\{\cdot\}$ denotes the natural logarithm. As a result, the maximum likelihood (ML) estimation of $\boldsymbol{x}$ can be obtained as

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}}{\operatorname{argmax}}\ \mathcal{L}(\boldsymbol{x}). \tag{3.7}$$

Observe that the maximum likelihood estimator $\hat{\boldsymbol{x}}$ has to satisfy the following condition:

$$\nabla_{\boldsymbol{x}}\mathcal{L}(\boldsymbol{x}) = \boldsymbol{0}, \tag{3.8}$$

where the gradient of the log-likelihood function with respect to the unknown vector $\boldsymbol{x}$ can be derived as follows:

$$\nabla_{\boldsymbol{x}}\mathcal{L}(\boldsymbol{x}) = \sum_{i=1}^{N}\left[-\frac{r_i}{\sigma_i}\left(\frac{Q'\left(\frac{r_i}{\sigma_i}\left(\tau_i - \boldsymbol{h}_i^T\boldsymbol{x}\right)\right)}{Q\left(\frac{r_i}{\sigma_i}\left(\tau_i - \boldsymbol{h}_i^T\boldsymbol{x}\right)\right)}\right)\right]\boldsymbol{h}_i, \tag{3.9}$$

where $Q'(x) = -\frac{1}{\sqrt{2\pi}}\exp\left(-x^2/2\right)$. It can be observed from (Equation 4.4) that the gradient of the log-likelihood function is a linear combination of the rows of the sensing matrix $\boldsymbol{H}$. Let $\boldsymbol{\eta}\colon \mathbb{R}^M \mapsto \mathbb{R}^M$ be a non-linear function defined as follows:

$$\boldsymbol{\eta}(\boldsymbol{x}) \triangleq \frac{Q'(\boldsymbol{x})}{Q(\boldsymbol{x})}, \tag{3.10}$$

where the functions $Q(\cdot)$, $Q'(\cdot)$, and the division, are applied element-wise on the vector argument $\boldsymbol{x}$. In addition, let $\boldsymbol{\Omega} = \mathbf{Diag}(r_1, \ldots, r_M)$ be a diagonal matrix containing the one-bit observations and $\tilde{\boldsymbol{\Omega}} = \boldsymbol{\Omega}\boldsymbol{C}^{-\frac{1}{2}}$ be the semi-whitened version of the *one-bit matrix* $\boldsymbol{\Omega}$. Then, the gradient of the likelihood function in (Equation 4.4) can be compactly written as follows:

$$\nabla_{\boldsymbol{x}}\mathcal{L}(\boldsymbol{x}) = -\boldsymbol{H}^T\tilde{\boldsymbol{\Omega}}\,\boldsymbol{\eta}\left(\tilde{\boldsymbol{\Omega}}(\boldsymbol{\tau} - \boldsymbol{H}\boldsymbol{x})\right). \tag{3.11}$$

Recall that the ML estimator $\hat{\boldsymbol{x}}$ must satisfy the condition given in (Equation 4.4), i.e.,

$$\nabla_{\boldsymbol{x}}\mathcal{L}(\boldsymbol{x}) = -\boldsymbol{H}^T\tilde{\boldsymbol{\Omega}}\,\boldsymbol{\eta}\left(\tilde{\boldsymbol{\Omega}}(\boldsymbol{\tau} - \boldsymbol{H}\boldsymbol{x})\right) = 0. \tag{3.12}$$

Other than certain low-dimensional cases, finding a closed-form expression for $\hat{\boldsymbol{x}}$ that satisfies (Equation 3.12) is a difficult task [58, 59, 97]. Therefore, we resort to iterative methods in order to find the ML estimate, i.e., to solve (Equation 4.3).

In this paper, the well-known gradient ascent method is employed to iteratively solve (Equation 4.3). Namely, given an initial point $\boldsymbol{x}^{(0)}$, the update equation at each iteration is given by:

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \delta^{(k)} \nabla_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{x}) \tag{3.13}$$

$$= \boldsymbol{x}^{(k)} - \delta^{(k)} \boldsymbol{H}^T \tilde{\boldsymbol{\Omega}}\, \boldsymbol{\eta} \left(\tilde{\boldsymbol{\Omega}} (\boldsymbol{\tau} - \boldsymbol{H}\boldsymbol{x}^{(k)}) \right), \tag{3.14}$$

where $\delta^{(k)}$ is the step size at the k-th iteration. The obtained maximum likelihood estimator derived from the signal model, and the corresponding optimization steps, can be unfolded into a multi-layer deep neural network, which improves the accuracy and computational effciency of the original framework.

In the next section, we *unfold* the above iterations into the layers of a deep neural network where each layer denotes one iteration of the above optimization method. Interestingly, we fix the complexity budget of the inference framework (via fixing the number of layers), and apply the gradient descent method to yield the most accurate estimation of the parameter in at most K iterations.

3.3 Signal Recovery via Deep Unfolding

Conventionally, first-order optimization methods, such as gradient descent algorithms, have slow convergence rate, and thus take a large number of iterations to converge to a solution.

Herein, we are interested in finding a good solution under the condition that the complexity of the inference algorithm is fixed. This is important since, via unfolding the optimization algorithm, we fix the computational complexity of the inference model (a DNN with K layers in such a case) and optimize the parameters of the network to find the best possible estimator with a fixed-complexity constraint. Below, we introduce ***DeepRec***, our deep learning based signal recovery framework which is designed based on the iterations of the form (Equation 3.14), to find the maximum likelihood estimation of the unknown parameter.

—**The *DeepRec* Architecture.** The construction of DeepRec involves the unfolding of $k = 1, \ldots, K$, iterations each of which are of the form (Equation 3.14), as the layers of a deep neural network. Particularly, each step of the gradient descent method depends on the previous signal estimate $\boldsymbol{x}^{(k)}$, the step size $\delta^{(k)}$, the scaled one-bit matrix $\tilde{\boldsymbol{\Omega}}$, the sensing matrix $\boldsymbol{H}$, and the threshold vector $\boldsymbol{\tau}$. In addition, the form of the gradient vector (Equation 4.6) makes it convenient and insightful to unfold the iterations onto the layers of a DNN in that each iteration of the gradient descent method is a linear combination of the system paramteres followed by a

non-linearity. The k-th layer of DeepRec can be characterized via the following operations and variables:

$$\boldsymbol{z}^{(k)} = \boldsymbol{W}_{1k}\tilde{\boldsymbol{\Omega}}\boldsymbol{\tau} - \boldsymbol{W}_{2k}\boldsymbol{H}\boldsymbol{x}^{(k)} + \boldsymbol{b}_{1k}, \tag{3.15}$$

$$\boldsymbol{p}^{(k)} = \boldsymbol{\eta}\left(\boldsymbol{z}^{(k)}\right), \tag{3.16}$$

$$\boldsymbol{t}^{(k)} = \boldsymbol{H}^T\tilde{\boldsymbol{\Omega}}\,\boldsymbol{p}^{(k)}, \tag{3.17}$$

$$\boldsymbol{x}^{(k+1)} = f\left(\boldsymbol{W}_{3k}\begin{bmatrix}\boldsymbol{x}^{(k)}\\ \boldsymbol{t}^{(k)}\end{bmatrix} + \boldsymbol{b}_{2k}\right), \tag{3.18}$$

where $\boldsymbol{x}^{(1)} = 0$, $f(\cdot)$ denotes a non-linear activation function where in this work we consider $f(x) \triangleq \text{ReLU}(x) = \max\{0, x\}$, and the goal is to optimize the DNN parameters, described as follows:

$$\boldsymbol{\Xi} = \{\boldsymbol{W}_{1k}, \boldsymbol{W}_{2k}, \boldsymbol{W}_{3k}, \boldsymbol{b}_{1k}, \boldsymbol{b}_{2k}\}_{k=1}^{K}. \tag{3.19}$$

The proposed DeepRec architecture with L layers can be interpreted as a class of estimator functions $\boldsymbol{\Psi}_{\boldsymbol{\Xi}}(\boldsymbol{r}, \boldsymbol{H}, \boldsymbol{\tau})$ parametrized by $\boldsymbol{\Xi}$ to estimate the unknown parameter $\boldsymbol{x}$ given the system parameters. In order to find the best estimator function $\boldsymbol{\Psi}_{\boldsymbol{\Xi}}(\boldsymbol{r}, \boldsymbol{H}, \boldsymbol{\tau})$ associated with our problem, we conduct a learning process via minimizing a loss function $\mathcal{R}(\boldsymbol{x}; \boldsymbol{\Psi}_{\boldsymbol{\Xi}}(\boldsymbol{r}, \boldsymbol{H}, \boldsymbol{\tau}))$, i.e.,

$$\min_{\boldsymbol{\Xi}} \quad \mathcal{R}(\boldsymbol{x}; \boldsymbol{\Psi}_{\boldsymbol{\Xi}}(\boldsymbol{r}, \boldsymbol{H}, \boldsymbol{\tau})). \tag{3.20}$$

In this work, we employ the following least squares (LS) loss function:

$$\mathcal{R}\left(\boldsymbol{x}; \boldsymbol{\Psi}_{\boldsymbol{\Xi}}(\boldsymbol{r}, \boldsymbol{H}, \boldsymbol{\tau})\right) = ||\boldsymbol{x} - \boldsymbol{\Psi}_{\boldsymbol{\Xi}}(\boldsymbol{r}, \boldsymbol{H}, \boldsymbol{\tau})||_2^2, \tag{3.21}$$

where during the training phase, we synthetically generate the system parameters $\boldsymbol{\Theta} = \{\boldsymbol{x}, \boldsymbol{r}, \boldsymbol{H}, \boldsymbol{\tau}\}$ according to their statistical model.

3.4 Numerical Results

We now demonstrate the performance of the proposed DeepRec framework for the problem of one-bit signal recovery. The proposed framework was implemented using the TensorFlow library [98], with the ADAM stochastic optimizer [84] and an exponential decaying step size. In the learning process of the network, we employed the batch training method with a batch size of 500 at each epoch and we performed the training for 2000 epochs. In all of the simulations, we assumed $N = 3$, i.e., $\boldsymbol{x} \in \mathbb{R}^3$, and we used the normalized mean square error (NMSE) defined as $NMSE = ||\boldsymbol{x} - \hat{\boldsymbol{x}}||_2^2 / ||\boldsymbol{x}||_2^2$, for the performance metric.

The training was performed based on the data generated via the following model. Each element of the vector $\boldsymbol{x}$ is assumed to be i.i.d and uniformly distributed, i.e., $\boldsymbol{x} \sim \mathcal{U}(\delta_l^{\boldsymbol{x}, \delta_u^{\boldsymbol{x}}})$. The sensing matrix is assumed to be fixed and follow a Normal distribution, where we consider $\boldsymbol{H} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. The quantization thresholds were also generated from a uniform distribution, $\boldsymbol{\tau} \sim \mathcal{U}(\delta_l^{\boldsymbol{\tau}, \delta_u^{\boldsymbol{\tau}}})$, where the lower and upper bound of the distribution is chosen in a fashion that at least covers the domain of $\boldsymbol{x}$. The noise is assumed to be independent from one sample to another and follows a Normal distribution, where the variance of each corresponding noise element is

different, e.g., the noise covariance $\boldsymbol{C} = \mathbf{Diag}(\sigma_1^2, \ldots, \sigma_M^2)$, with $\sigma_i^2 \sim \mathcal{U}(\delta_1^{\boldsymbol{n}}, \delta_M^{\boldsymbol{n}})$. Note that we trained the network over a wide range of noise powers in order to make the DeepRec network more robust to noise.

Fig. 11(a) demonstrates the performance of the DeepRec network for different numbers of layers K. It can be observed that the averaged NMSE decreases dramatically as the number of layers increases. Such a result is also expected as each layer corresponds to one iteration of originial optimization algorithm. Thus, as the number of layers increases, the output of the network will converge to a better estimation as well.

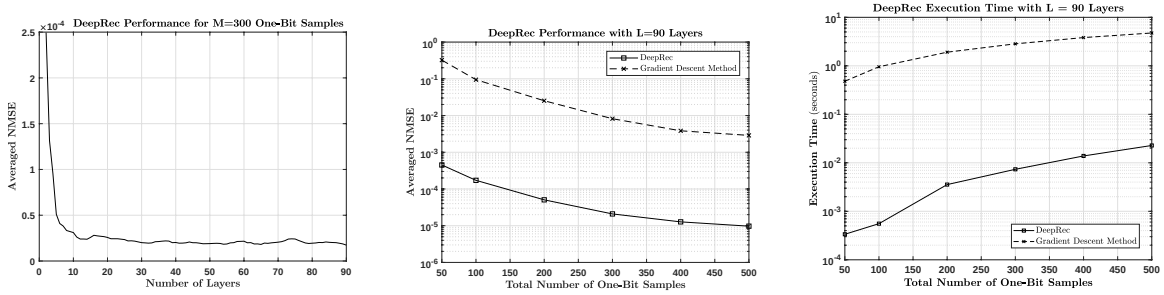
Fig. 11(b) demonstrates the performance of the proposed DeepRec architecture and the original Gradient Descent method of (Equation 3.14) in terms of averaged NMSE for different numbers of one-bit samples M. In this simulation, we implemented the DeepRec network with $K = 90$ layers. It can be clearly seen from Fig. 11(b) that the proposed deep recovery architecture (DeepRec) significantly outperforms the original optimization method in terms of accuracy and provides a considerably better estimation than that of the gradient descent method for the same number of iterations/layers. As a fair comparison, we also assumed a fixed-step size of $\delta = 0.01$ for the gradient descent method.

Fig. 11(c) shows a comparison of the computational cost (machine runtime) between the gradient descent method and the proposed DeepRec network for different numbers of one-bit samples M. It can be seen that our proposed method (DeepRec) has a significantly lower computational cost than that of the original optimization algorithm for our problem. Hence,

making the DeepRec a good candidate for real-time signal processing or big data applications (the results were obtained on a standard PC with a quad-core 2.30GHz CPU and 4 GB memory).

3.5 Conclusion

We have considered the application of model-based machine learning, and specifically the deep unfolding technique, in the problem of recovering a high-dimensional signal from its one-bit quantized noisy measurements via random thresholding. We proposed a novel deep architecture, which we refer to as *DeepRec*, that was able to accurately perform the task of one-bit signal recovery. Our numerical results show that the proposed DeepRec network significantly improves the performance of traditional optimization methods both in terms of accuracy and efficiency.



(a) **(b)** **(c)**

Figure 11. The performance of DeepRec: **(a)** demonstrates the NMSE performance of the DeepRec network for different numbers of layers K. **(b)** shows the performance of the proposed DeepRec architecture and the original gradient descent method of (Equation 3.14) in terms of averaged NMSE for different numbers of one-bit samples M. **(c)** shows a comparison of the computational cost between the gradient descent method and the proposed DeepRec network for different numbers of one-bit samples M.

I-B

Model-Based Deep Learning for Compressive Sensing

CHAPTER 4

MODEL-BASED DEEP LEARNING FOR ONE-BIT COMPRESSIVE SENSING

Overview: In this work, we consider the problem of one-bit deep compressive sensing from both a system design and a signal recovery perspective. In particular, we develop hybrid model-based deep learning architectures based on the deep unfolding methodology. We further interpret the overall data-acquisition and signal recovery modules as an auto-encoder structure allowing for learning task-specific sensing matrix, quantization thresholds, as well as the latent-parameters of iterative first-order optimization algorithms specifically designed for the problem of one-bit sparse signal recovery. The proposed model-based deep architectures have the ability to adaptively learn the proper quantization thresholds, paving the way for amplitude recovery in one-bit compressive sensing. We further show that the proposed methodology implicitly learns task-specific sensing matrices with very low coherence, which is highly desirable in a compressive sensing setting. Due to the model-based nature of the proposed deep architecture, it enjoys from the interpretability and versatility of model-based techniques as well as benefiting from the expressive power of data-driven methods. Specifically, owing to its model-based nature, it has far fewer parameters and requires far less samples for training as compared to black-box

---

machine learning models. Our results demonstrate a significant improvement compared to state-of-the-art algorithms.

Keywords: Compressive sensing, low-resolution signal processing, one-bit quantization, deep unfolding, deep neural networks, model-based deep learning, autoencoders

4.1 Introduction

In the past two decades, compressive sensing (CS) has shown significant potential in enhancing sensing and recovery performance in signal processing, occasionally with simpler hardware, and thus, has attracted noteworthy attention among researchers. CS is a method of signal acquisition which ensures the exact or almost exact reconstruction of certain classes of signals using far less number of samples than what is needed in the Nyquist sampling regime [101]—where the signals are typically reconstructed by finding the sparsest solution of an under-determined system of equations using various available means.

In a practical setting, each measurement is to be digitized into finite-precision values for further processing and storage purposes, which inevitably introduces a quantization error. This error is usually modeled as an additive Gaussian noise, independent of the input source signal; an approach that does not perform well in extreme cases of quantization. One-bit CS is an extreme case of qunatization where only the information about the sign of each measured sample is retained $r \in \{\pm 1\}$ [102–106]. One-bit quantizers are favourable due to their cost effectiveness and low power consumption properties. Moreover, their sampling speed is far higher than traditional scalar quantizers [107], accompanied by great reduction in the complexity of hardware implementation. Several algorithms have been introduced in the literature for efficient recon-

struction of sparse signals in one-bit CS scenarios (e.g., see [102–106, 108] and the references therein). A detailed discussion of such algorithms is provided in Sec II.

Notation: We use bold lowercase letters for vectors and bold uppercase letters for matrices. $(\cdot)^T$, and $(\cdot)^H$ denote the vector/matrix transpose, and the Hermitian transpose, respectively. $\mathbf{1}$ and $\mathbf{0}$ are the all-one and all-zero vectors. $\|\boldsymbol{x}\|_n$ denotes the ℓ_n-norm of the vector $\boldsymbol{x}$ defined as $(\Sigma_k|\boldsymbol{x}(k)|^n)^{\frac{1}{n}}$. $\boldsymbol{x}(i)$ denotes the i-th element of the vector $\boldsymbol{x}$ and $\boldsymbol{A}(i,j)$ denotes the ij-th element of the matrix $\boldsymbol{A}$. $\text{Diag}(\boldsymbol{x})$ denotes the diagonal matrix formed by the entries of the vector argument $\boldsymbol{x}$. The operator $\succeq$ denotes the element-wise vector inequality operator.

4.1.1 Background and Relevant Prior Art

One-bit compressive sensing is mainly concerned with the following data-acquisition model:

$$\boldsymbol{r} = \text{sign}(\boldsymbol{\Phi}\boldsymbol{x} - \boldsymbol{b}), \tag{4.1}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ denotes a K-sparse source signal, $\boldsymbol{\Phi} \in \mathbb{R}^{m\times n}$ is the sensing matrix, and $\boldsymbol{b} \in \mathbb{R}^n$ denotes the quantization thresholds vector. In addition to the mentioned advantages of using one-bit ADCs for data-acquisition purposes, the use of one-bit information offers increased robustness to undesirable non-linearities in the data-acquition process. Furthermore, there exists strong empirical evidence that recovering a sparse source signal from only one-bit measurement can outperform its multi-bit CS counterpart [104, 109].

In this works, we seek to take a deeper look at the one-bit CS arena from both a *system design* perspective related to the design of task-specific high-quality sensing matrices $\boldsymbol{\Phi}$ and

one-bit quantization systems (i.e., designing task-specific qunatization thresholds $\boldsymbol{b}$), and the development of accurate and efficient *task-specific signal reconstruction techniques* for one-bit signal recovery in a CS setting. In the following, we go into the details of each aspect and give an overview of the existing methodologies.

Sensing Matrix Design. From a system design point of view, the most relevant factors to be taken into account are the sensing matrix and the one-bit quantization thresholds used for data-acquisition. In particular, two key requirements of a CS-based signal reconstruction algorithm are the *sparsity* of the underlying signal of interest and the *incoherence* which is mainly related to the underlying sensing matrix $\boldsymbol{\Phi}$ employed at time of the acquisition. It can be shown that one can recover the underlying sparse signal from the linear compressive measurements with overwhelming high probability if the sensing matrix have a low coherence property [110]. Accordingly, such performance guarantees is based upon the Restricted Isometric Property (RIP) which is at the heart of CS theory. Specifically, for a sensing matrix $\boldsymbol{\Phi}$, and two sparse vectors $\boldsymbol{x}$ and $\boldsymbol{y}$, the RIP can be stated as follows:

$$(1-\gamma)\|\boldsymbol{x}-\boldsymbol{y}\|_2^2 \leq \|\boldsymbol{\Phi}(\boldsymbol{x}-\boldsymbol{y})\|_2^2 \leq (1+\gamma)\|\boldsymbol{x}-\boldsymbol{y}\|_2^2,$$

where $\gamma \in (0,1)$. In short, for a matrix $\boldsymbol{\Phi}$ satisfying the RIP condition, we have that the distance of any two vectors (signals) is maintained up to the bounding factors $\{1-\delta, 1+\delta\}$ after applying the transformation $\boldsymbol{\Phi}$. Accordingly, one can perform an almost perfect reconstruction of an sparse signal with high probability when the RIP condition is met by the measurement

matrix [101]. Nonetheless, certifying the RIP condition for a given matrix is a difficult task in general and it has been shown to be an NP-hard problem [111]. Consequently, there exist two main strategies in employing sensing matrices for CS in the literature: The first one considers the deployment of random matrices at the time of acquisition, while the other approach makes use of task-specific deterministic sensing matrices. Having said that, in a general CS setting, the works of [112, 113] have shown that random sensing matrices (e.g., Gaussian matrices) satisfy the RIP condition with a very high probability, hence providing mathematical ground-work for robust sparse signal recovery. Taking this into consideration, many signal reconstruction techniques, including but not limited to basis pursuit techniques, can be shown to provably recover a sparse signal when random matrices are employed [114]. Nonetheless, using random matrices is not applicable in many applications due to the imposed randomness in the measurement system [114]. More importantly, in many applications the sensing matrix must be designed in a fashion to account for the intricate physical model of the system and the measurement model. In such applications, one can resort to a deterministic design (in contrast to a purely random linear measurement) of the sensing matrix to accommodate for the measurement medium of interest. However, as previously mentioned, it is a very difficult task to verify the RIP condition for a matrix, and it cannot be easily used as an objective for a deterministic design of sensing matrices. Note that there exist several alternative measures for quantifying the quality of a sensing matrix. The most notable one and widely used metric is the *mutual coherence* [115], which is a mathematically tractable alternative metric for measuring the incoherence required by the compressed sensing theory and the success of many basis pursuit algorithms. Specifically,

let $\bar{\boldsymbol{\Phi}}$ denote the column-normalized version of the sensing matrix $\boldsymbol{\Phi}$ and define the Gram matrix $\boldsymbol{M} \triangleq \bar{\boldsymbol{\Phi}}^H \bar{\boldsymbol{\Phi}}$. Then, the mutual coherence of a sensing matrix $\boldsymbol{\Phi}$ is given by

$$\mu(\boldsymbol{\Phi}) = \max_{i \neq j} \ |\boldsymbol{M}(i,j)|. \tag{4.2}$$

Furthermore, the off-diagonal entries $\{|\boldsymbol{M}(i,j)|\}_{i \neq j}$ represent the coherence coefficients of the sensing matrix $\boldsymbol{\Phi}$. Briefly speaking, the mutual coherence factor $\mu(\boldsymbol{\Phi})$ provides a measure of the worst-case similarity between the columns of $\boldsymbol{\Phi}$, and furthermore, a high mutual coherence results in a significant degradation in the performance of basis pursuit signal recovery techniques [116]. Hence, it is highly desirable to have a sensing matrix with low mutual coherence, corresponding to a Gram matrix $\boldsymbol{M}$ close to identity $\boldsymbol{I}$.

Designing task-specific and deterministic sensing matrices with low-coherence is an active research area in various fields such as coding and communication [117], quantum signal processing [118], machine learning [119], radar signal processing [120], among many others. Perhaps, one of the most interesting of these applications is the one-bit compressive sensing area. To the best of our knowledge, there exist no existing work in the literature that studies the design of task-specific deterministic sensing matrices in a one-bit CS setting and its advantages over using random matrices. Hence, *one of the main motivations of this work is to address this issue and to propose a unified framework that allows for designing task-specific deterministic sensing matrices that can handle the severe non-linearity imposed by the one-bit quantization at the time of acquisition, which further allows for a significant improvement of the signal reconstruction*

accuracy at the time of inference. The proposed methodology does not require an explicit optimization over the mutual coherence which may be difficult to handle from an optimization point of view. Indeed, we empirically show that the proposed methodology *implicitly* learns task-specific sensing matrices with *very low mutual coherence* leading significantly enhancing the signal reconstruction accuracy.

We conclude this part by emphasizing that although it is common to consider the problem of one-bit CS (or CS in general) and the design of task-specific sensing matrices for such systems from a purely mathematical point-of-view, the development of CS and one-bit CS-based hardware data acquisition systems is still a great challenge in practice. The successful implementation of such systems might require further integration of theory and practice, considering various limitations of physical hardware.

Low-Resolution Quantization and Signal Recovery. The other factor that affects the performance of the signal recovery in a one-bit CS setting is the choice of the quantization thresholds. There exist two strategies to be undertaken regarding the qunatization thresholds: The first one is concerned with setting the quantization thresholds to zero while the other way is to consider non-zero thresholds. In both settings, the current one-bit CS recovery algorithms typically exploit the *consistency principle*, which represents the fact that the element-wise product of the sparse signal and the corresponding measurement is always positive [102], i.e. $\boldsymbol{r} \odot (\boldsymbol{\Phi x} - \boldsymbol{b}) \succeq 0$. However, most of the existing literature on one-bit CS considers zero-level one-bit quantization thresholds (i.e., $\boldsymbol{b} = \boldsymbol{0}$) leading to a total loss of amplitude information during the data-acquisition process. Hence, by comparing the signal level with zero, one can only

recover the direction of the source signal, i.e. $\boldsymbol{x}/\|\boldsymbol{x}\|_2$, and not the amplitude information $\boldsymbol{x}$. In its most general form, any solution $\boldsymbol{x}^*$ to the one-bit CS problem should: (i) satisfy the sparsity condition, i.e. $\|\boldsymbol{x}^*\|_0 \leq K$ with $K = \|\boldsymbol{x}\|_0$, and (ii) achieve consistency, i.e. $\boldsymbol{r} \odot (\boldsymbol{\Phi}\boldsymbol{x}^* - \boldsymbol{b}) \succeq 0$. As mentioned above, most of the existing literature on the problem of one-bit CS recovery problem considers the case of $\boldsymbol{b} = \boldsymbol{0}$. In such a case, the solution to the one-bit CS problem can be expressed as:

$$\boldsymbol{x}^* = \underset{\boldsymbol{x}}{\operatorname{argmin}} \ \|\boldsymbol{x}\|_0 \ \text{ s.t. } \ \boldsymbol{r} = \operatorname{sign}(\boldsymbol{\Phi}\boldsymbol{x}).$$

The above program is NP-hard and mathematically intractable [104]. However, there exist several powerful iterative algorithms to find $\boldsymbol{x}^*$ (for the case of $\boldsymbol{b} = \boldsymbol{0}$) that rely on a relaxation of the ℓ_0-norm to its convex hull (i.e., using ℓ_1-norm in lieu of ℓ_0-norm) to obtain an estimate of the support of the true source signal by restricting the feasible solutions to the unit-sphere, i.e. $\|\boldsymbol{x}\|_2 = 1$.

The most notable works which considers a zero quantization thresholding scheme are as follows. In [102], the authors assume a zero-level quantization threshold and propose an iterative algorithm called *renormalized fixed point iteration* (RFPI) where a convex barrier function is used to enforce the consistency principle (as a regularization term in the objective function). A detailed analysis of the RFPI algorithm is provided in Sec. II. It is worth mentioning that in a traditional CS setting, one consider the under-sampled measurements (i.e., $m < n$), however, the *over-sampling* regime is beneficial and of paramount interest in a one-bit CS setting in that the use of one-bit ADCs provide a cheap and fast way to acquire measurements and to potentially go beyond the limitations of the traditional CS methods. Another such reconstruction algorithm

can be found in [103], referred to as *restricted step shrinkage* (RSS), for which a nonlinear barrier function is used as the regularizer to enforce the consistency principle. Compared to RFPI algorithm, RSS has three important advantages: provable convergence, improved consistency, and feasible performance [121]. Ref. [104] introduces a penalty-based robust recovery algorithm, called *binary iterative hard thresholding* (BIHT), in order to enforce the consistency principle. Contrary to RFPI algorithm, BIHT exploits the knowledge of the sparsity level of the signal as input, and was shown to be more robust to outliers and have a superior performance than that of the RFPI method in some cases (at the cost of knowing the sparsity level of the source signal a priori). Both RFPI and BIHT, however, only consider a zero-level quantization threshold, as a result, the amplitude information is lost due to comparing the acquired signal with zero. In [105] and [106], authors proposed modified versions of RFPI and BIHT, referred to as *noise-adaptive renormalized fixed point iteration* (NARFPI) and *adaptive outlier pursuit with sign flips* (AOP-f), that are robust against bit flips in the measurement signal. In [122], the authors lay the ground work for a theoretical analysis of noisy one-bit CS problem based on a convex programming approach for the problem of one-bit sparse signal recovery in a noisy setting.

There exist limited work on employing non-zero quantization thresholds in a one-bit CS setting. Recently, the authors in [109] considered the problem of one-bit CS signal reconstruction in a non-zero quantization thresholds setting that enables the recovery of the norm of the source signal, i.e. recovering $\|\boldsymbol{x}\|_2$. However, the proposed method in [109] still fails to accurately recover the amplitude information of the source signal, and does not offer a straight-forward method to design the quantization thresholds. Although the one-bit CS has a deterministic

system model, the authors in [123] consider a non-zero quantization scheme and provide a Bayesian formulation of the problem upon which a generalized approximate message passing (GAMP) algorithm is used for signal recovery purposes. Furthermore, non-zero quantization thresholds for one-bit compressive systems has been adopted in other fields such as one-bit compressive radar systems [120].

In light of the above, it is of paramount importance to develop *computationally efficient* one-bit CS models that can incorporate non-zero quantization thresholds to allow for recovering the amplitude information. Additionally, the vast literature on the one-bit CS recovery problem, does not yet tap into the potential of the available data at hand (to improve the performance recovery). One can significantly benefit from a methodology that can facilitate not only incorporation of the domain knowledge on the problem (i.e., being model-driven), but also the available data at hand to go beyond the performance of the traditional sparsity aware signal processing techniques in a one-bit CS scenario.

There has recently been a high demand for developing effective real-time signal processing algorithms that use the data to achieve improved performance [66, 67, 124–127]. In particular, the data-driven approaches relying on deep neural architectures such as convolutional neural networks [124], deep fully connected networks [125], stacked denoising autoencoders [126], and generative adversarial networks [128] have been studied for sparse signal recovery in generic quantized CS settings. we note that, parameterized mathematical models discussed above play a central role in understanding and design of large-scale information systems and signal processing methods. However, they usually fail to incorporate the complex interactions in such systems.

In contrast to the mentioned models, black-box data-driven methodologies, and specifically deep learning techniques, do not need explicit mathematical models for data generation and have a wider applicability at the cost of interpretability. The main advantage of the deep learning-based approach is that it employs several non-linear transformations to obtain an abstract representation of the underlying data. Data-driven approaches, on the other hand, lack the interpretability and trustability that comes with model-based signal processing. They are particularly prone to be questioned further, or at least not fully trusted by the users, especially in critical applications. Furthermore, the deterministic deep architectures are generic and it is unclear how to incorporate the existing knowledge on the problem in the processing stage. *The advantages associated with both model-based and data-driven methods show the need for developing frameworks that bridge the gap between the two approaches.*

The recent advent of the *deep unfolding framework* [75, 79, 95, 129–131] and the corresponding deep unfolding networks (DUNs) has paved the way for a game-changing fusion of models and well-established signal processing approaches with data-driven architectures. In this way, we not only exploit the vast amounts of available data, but also integrate the prior knowledge of the system model in the processing stage. Deep unfolding relies on the establishment of an optimization or inference iterative algorithm, whose iterations are then *unfolded* into the layers of a deep network, where each layer is designed to resemble one iteration of the optimization/inference algorithm. The resulting hybrid method benefits from low computational cost (in execution stage) of deep neural networks, and at the same time, from the versatility and reliability of model-based methods; thus, appears to be an excellent tool in real-time signal

processing applications due to the smaller degrees of freedom required for training and execution (afforded by integration of the problem-level reasoning, or the *model*). A detailed analysis of the deep unfolding methodology for the problem of one-bit CS is provided in Sec. 7.3.

4.1.2 Contributions of the Paper

In this paper, we propose a novel hybrid model-based and data-driven methodology (based on DUNs) that addresses the drawbacks of both purely model-based (such as the discussed RFPI and BIHT algorithm) and purely data-driven approaches. The resulting methodology is far less data-hungry and assumes a slight over-parametrization of the system model as opposed to traditional deep learning techniques with extremely large number of variables to be learned. In particular, the proposed method seeks to bridge the gap between the data-driven and model-based approaches in the one-bit CS paradigm, resulting in a specialized architecture for the purpose of sparse signal recovery from one-bit measurements. In particular, the proposed methodology allows for learning task-specific sensing matrices with very low mutual coherence factor, as well as learning data-specific quantization thresholds. Furthermore, we propose a novel model-based interpretable deep learning model for sparse signal recovery in a one-bit CS setting and show that the proposed methodology outperforms the existing state-of-the-art methodologies in the area of one-bit CS. The proposed framework can be seen as a unification of system design and signal recovery techniques, allowing for a joint optimization of all system parameters. The contributions of this paper can be summarized as follows:

• We propose a novel hybrid model-based and data-driven one-bit compressive autoencoding (AE) methodology that can deal with the optimization of the sensing matrix $\mathbf{\Phi}$ (learning task-

specific deterministic sensing matrices), the one-bit quantization thresholds $\boldsymbol{b}$, and the latent-variables of the decoder module according to the underlying distribution of the source signal. Hence, such a methodology allows for quick adaptation to new data distributions and environments.

• To the best of our knowledge, this is the first attempt in the one-bit CS paradigm that allows for joint optimization of the quantization thresholds and sensing matrix, also facilitating the *recovery of the amplitude information of the source signal.* We show that by using the proposed AEs, one can significantly improve upon existing iterative algorithms and gain much higher accuracy both in terms of recovering the magnitude and the support of the underlying source signal.

• The proposed methodology exhibits performance that goes beyond the traditional one-bit CS state-of-the-art and allows for designing sensing matrices that are distribution-specific. In conjunction to learning task-specific $\boldsymbol{\Phi}$, the quantization thresholds can also be learned in a joint manner such that the learned parameters improve the signal reconstruction accuracy and speed.

• We propose two generalized optimization algorithms that can be used as standalone algorithms for recovering the amplitude information of the source signal by utilizing non-zero quantization thresholds.

Organization of the Paper: The remainder of this paper is organized as follows. In Sec. II, we discuss the general problem formulation and system model of the one-bit compressive sensing problem and propose two general algorithms that pave the way for incorporating non-zero quantization thresholds. The proposed one-bit compressive autoencoding methodology is

presented in Sec. III. The loss function characterization and training method for the proposed model-based deep architectures are discussed at the end of Sec. III. In Sec. IV, we investigate the performance of the proposed methods through various numerical simulations and for various scenarios. Finally, Sec. V concludes the paper.

4.2 System Model and Problem Formulation

In this paper, we are interested in a one-bit CS measurement model (i.e., the encoder module) with dynamics that can be described as follows:

$$\text{Encoder Module:} \qquad \boldsymbol{r} = \text{sign}(\boldsymbol{\Phi x} - \boldsymbol{b}), \tag{4.3}$$

where $\boldsymbol{\Phi}^{m\times n}$ denotes the sensing matrix, $\boldsymbol{b} \in \mathbb{R}^m$ is the quantization thresholds, and $\boldsymbol{x} \in \mathbb{R}^n$ is assumed to be a K-sparse signal. Having the one-bit measurements of the form (Equation 4.3), one can pose the problem of sparse signal recovery from one-bit measurements $\boldsymbol{r}$ by solving the following non-convex program:

$$\mathcal{P}_0 : \quad \min_{\boldsymbol{x}\in\mathbb{R}^n} \ \|\boldsymbol{x}\|_0, \quad \text{s.t.} \quad \boldsymbol{r} = \text{sign}(\boldsymbol{\Phi x} - \boldsymbol{b}), \tag{4.4}$$

where the constraint in (Equation 4.4) is imposed to ensure a consistent reconstruction with the available one-bit information. Further note that the one-bit measurement consistency principle in (Equation 4.4) can be equivalently expressed as

$$\boldsymbol{R}\,(\boldsymbol{\Phi x} - \boldsymbol{b}) \succeq \boldsymbol{0}, \tag{4.5}$$

where $\boldsymbol{R} = \text{Diag}(\boldsymbol{r})$.

Let us first consider the scenario in which the quantization thresholds $\boldsymbol{b}$ are all set to zero. In this case, the non-convex optimization problem $\mathcal{P}_0$ can be further relaxed and expressed as a well-known non-convex ℓ_1-minimization program on the unit sphere [102]:

$$\mathcal{P}_1 : \min_{\boldsymbol{x}} \quad \|\boldsymbol{x}\|_1, \quad \text{s.t.} \quad \boldsymbol{R}\boldsymbol{\Phi}\boldsymbol{x} \succeq \boldsymbol{0}, \;\; \|\boldsymbol{x}\|_2 = 1, \tag{4.6}$$

where the ℓ_1-norm acts as a sparsity inducing function. The intuition behind finding the sparsest signal on the ℓ_2 unit-sphere (i.e., fixing the energy of the recovered signal) is two-fold. First, it reduces the feasible set of the optimization problem as the amplitude information is lost, and second, it avoids the the trivial solution of $\hat{\boldsymbol{x}} = \boldsymbol{0}$. By comparing the acquired data $\boldsymbol{y} = \boldsymbol{\Phi}\boldsymbol{x}$ with non-zero quantization thresholds, the constraint defined in (Equation 4.5) not only reduces the feasible set of the problem by defining a set of hyper-planes where the signal can reside on, but also, implicitly exclude the trivial solution. There exists an extensive body of research on approximately solving the non-convex optimization problem $\mathcal{P}_1$ (e.g., see [102,103,105,132,133], and the references therein). The most notable methods utilize a regularization term $\mathcal{R}(\boldsymbol{s})$ to enforce the consistency principle via a penalty term added to the ℓ_1-objective function, viz.

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x} \in \mathbb{R}^n}{\arg\min} \; \|\boldsymbol{x}\|_1 + \alpha \mathcal{R}(\boldsymbol{R}\boldsymbol{\Phi}\boldsymbol{x}), \;\; \text{s.t.} \;\; \|\boldsymbol{x}\|_2 = 1, \tag{4.7}$$

where $\alpha > 0$ is the penalty factor.

Among the numerous iterative algorithms available for tackling the optimization problem in (Equation 4.7), we plan to utilize and improve upon the state-of-the-art renormalized fixed-point iterations (RFPI) [102], and the Binary Iterative Hard Thresholding (BIHT) [104] algorithms as the starting point for our proposed hybrid model-aware deep architecture for the problem of one-bit compressive sensing. To this end, we interpret a one-bit CS setting as a single auto-encoder (AE) module allowing for an optimization over all system parametes (i.e., the sensing matrix, quantization thresholds, and the laten variables of first-order optimization techniques). Namely, in the subsequent sections, we use the mentioned algorithms as a base-line to design the decoder module of our one-bit CS AE. In particular, we unfold the iterations of the two specialized algorithms onto the layers of a deep neural network in a fashion that each layer of the proposed deep architecture mimics the behavior of one iteration of the base-line algorithm. Next, we perform an end-to-end learning approach by utilizing the back-propagation method to tune the parameters of both the decoder and the encoder functions of the proposed one-bit compressive AE.

4.2.1 Renormalized Fixed-Point Iteration (RFPI)

The RFPI algorithm considers a one-bit CS data acquisition model where the quantization thresholds are all set to zero. With $\boldsymbol{c} = \boldsymbol{R\Phi x}$ and $\boldsymbol{b} = \boldsymbol{0}$, the RFPI algorithm utilizes the following regularization term to enforce the consistency constraint in (Equation 4.6): $\mathcal{R}(\boldsymbol{c}) = \frac{1}{2}\left\|\rho(\boldsymbol{c})\right\|_2^2$, where $\rho(\boldsymbol{c}) \triangleq \max\{-\boldsymbol{c}, \boldsymbol{0}\}$, and the function max is applied element-wise on the vector arguments. Note that the function $\rho(\cdot)$ can be expressed in terms of the well-known Rectifier Linear Unit () function extensively used by the deep learning research community,

i.e. $\rho(\boldsymbol{c}) = (-\boldsymbol{c})$. Briefly speaking, the RFPI algorithm is a first-order optimization method (gradient-based) that operates as follows: given an initial point $\boldsymbol{x}_0$ on the unit-sphere (i.e., $\|\boldsymbol{x}_0\|_2 = 1$), the gradient step-size δ and a shrinkage thresholds α (or equivalently the penalty term), at each iteration i, the estimated signal $\boldsymbol{x}_i$ is obtained using the following update steps:

$$\boldsymbol{d}_i = \nabla_{\boldsymbol{x}} \mathcal{R}(\boldsymbol{z})\big|_{\boldsymbol{x}=\boldsymbol{x}_{i-1}} = -\left(\boldsymbol{R}\boldsymbol{\Phi}\right)^T \rho\left(\boldsymbol{R}\boldsymbol{\Phi}\boldsymbol{x}_{i-1}\right), \tag{4.8a}$$

$$\boldsymbol{t}_i = \left(1 + \delta \boldsymbol{d}_i^T \boldsymbol{x}_{i-1}\right) \boldsymbol{x}_{i-1} - \delta \boldsymbol{d}_i, \tag{4.8b}$$

$$\boldsymbol{v}_i = \operatorname{sign}\left(\boldsymbol{t}_i\right) \odot \left(|\boldsymbol{t}_i| - (\delta/\alpha)\mathbf{1}\right), \tag{4.8c}$$

$$\boldsymbol{x}_i = \frac{\boldsymbol{v}_i}{\|\boldsymbol{v}_i\|_2}. \tag{4.8d}$$

After the descent in (Equation 4.8a)-(Equation 6.6), the update step in (Equation 6.7) corresponds to a shrinkage step. More precisely, any element of the vector $\boldsymbol{t}_i$ that is below the threshold δ/α will be pulled down to zero (leading to enhanced sparsity). Finally, the algorithm projects the obtained vector $\boldsymbol{v}_i$ on the unit sphere to produce the latest estimation of the signal. Note that the latter step is necessary due to the fact that a zero-threshold vector (i.e., $\boldsymbol{b} = \mathbf{0}$) is employed at the time of the data acquisition, and hence, the amplitude information is lost.

While effective in signal reconstruction, there exist several drawbacks in using the RFPI method. For instance, it is required to use the algorithm on several problem instances, while increasing the value of the penalty factor α at each outer iteration of the algorithm, and to use the previously obtained solution as the initial point for tackling the recovery problem for any new problem instance. Moreover, it is not straight-forward how to choose the fixed step-size

and the shrinkage threshold, that may depend on the latent-parameters of the system. In fact, it is evident that by carefully tuning the step-sizes and the shrinkage threshold $\tau = \delta/\alpha$, one can significantly boost the performance of the algorithm, and further alleviate the mentioned drawbacks of this method. In what follows, we extend the above iterations in a fashion that it allows for incorporating the non-zero quantization thresholds, and hence, enabling us to effectively recover the amplitude information of the source signal.

A.1. Extending the RFPI framework to non-zero quantization thresholds:

Recall that our focus is on the following encoding (measurement) model with an arbitrary threshold vector $\boldsymbol{b}$:

$$\boldsymbol{r} = \text{sign}(\boldsymbol{\Phi x} - \boldsymbol{b}). \tag{4.9}$$

Therefore, the problem of one-bit CS signal recovery with a non-zero quantization threshold vector can be cast as:

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \ \|\boldsymbol{x}\|_1, \quad \text{s.t.} \quad \boldsymbol{R}(\boldsymbol{\Phi x} - \boldsymbol{b}) \succeq \boldsymbol{0}. \tag{4.10}$$

Inspired by the regularization-based relaxation employed in [102], we relax the above program and cast it as follows:

$$\mathcal{P}_2 : \ \min_{\boldsymbol{x}\in\mathbb{R}^n} \|\boldsymbol{x}\|_1 + \frac{1}{2}\|\rho\left(\boldsymbol{R}\left(\boldsymbol{\Phi x} - \boldsymbol{b}\right)\right)\|_2^2. \tag{4.11}$$

Note that the second term in the objective function above applies a quadratic penalty to the negative entries of the vector $\boldsymbol{R}(\boldsymbol{\Phi x} - \boldsymbol{b})$, i.e., the ones which are not consistent with the acquired one-bit measurements.

In this work, we consider an iterative first-order optimization solver to tackle the above optimization problem. Specifically, we make use of the proximal algorithm [15] to derive the updating steps required to solve $\mathcal{P}_2$. Let $f(\boldsymbol{x}) := g(\boldsymbol{x}) + h(\boldsymbol{x})$ be a composite convex objective function and consider the following optimization problem:

$$\min_{\boldsymbol{x} \in \mathbb{R}^n} f(\boldsymbol{x}) \equiv g(\boldsymbol{x}) + h(\boldsymbol{x}). \tag{4.12}$$

Furthermore, define the proximal operator $\text{prox}_{\alpha h}(\boldsymbol{x})$ for a given convex differentiable function $h(\boldsymbol{x})$ as follows:

$$\text{prox}_{\alpha h}(\boldsymbol{x}) = \underset{\boldsymbol{z} \in \mathbb{R}^n}{\text{argmin}} \; h(\boldsymbol{z}) + \frac{1}{2\alpha}\|\boldsymbol{z} - \boldsymbol{x}\|_2^2, \tag{4.13}$$

where $\alpha > 0$. Then, starting from an initial point $\boldsymbol{x}_0$ and a step-size $\delta \in (0, 1)$, the overall updating equations of the proximal gradient method for solving (Equation 4.12) can be expressed as:

$$\boldsymbol{x}_{i+1} := \text{prox}_{\alpha h}\left(\boldsymbol{x}_i - \delta \nabla_{\boldsymbol{x}} g(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x}_i}\right), \tag{4.14}$$

where it can be shown that for a convex function h, the proximal operator is monotone and the above mapping (Equation 4.14) has a fixed point, coinciding with the global solution of (Equation 4.12) [15]. Evidently, the proximal gradient algorithm is well suited for tackling our optimization problem $\mathcal{P}_2$. In order to find the updating steps specific to our problem, we consider the splitting of the objective function in (Equation 4.11) into $g(\boldsymbol{x}) = (1/2)\|\rho(\boldsymbol{R}(\boldsymbol{\Phi}\boldsymbol{x} - \boldsymbol{b}))\|_2^2$, and $h(\boldsymbol{x}) = \|\boldsymbol{x}\|_1$. In the following, we first derive the gradient calculations of the function $g(\boldsymbol{x})$, and then, proceed with presenting the final update equations akin to our optimization problem at hand based on (Equation 4.14).

Gradient calculation of $g(\boldsymbol{x})$: Recall that the function $\rho(\boldsymbol{x}) = \max\{-\boldsymbol{x}, \boldsymbol{0}\}$ is applied element-wise on the vector argument. Let $[\boldsymbol{x}]_i$ represent the i-th element of the vector $\boldsymbol{x}$, $\boldsymbol{\phi}_i$ be a column-vector denoting the i-th row of the sensing matrix $\boldsymbol{\Phi}$, and r_i and b_i represent the i-th one-bit measurement and quantization threshold, respectively. Define

$$g_i(\boldsymbol{x}) \triangleq \frac{1}{2}\left([\rho(\boldsymbol{R}(\boldsymbol{\Phi}\boldsymbol{x} - \boldsymbol{b}))]_i\right)^2 \tag{4.15}$$

$$= \frac{1}{2}\left(\rho(r_i(\boldsymbol{\phi}_i^T\boldsymbol{x} - b_i))\right)^2$$

$$= \begin{cases} \frac{1}{2}(r_i)^2\left(\boldsymbol{\phi}_i^T\boldsymbol{x} - b_i\right)^2 & \text{if } r_i\left(\boldsymbol{\phi}_i^T\boldsymbol{x} - b_i\right) < 0, \\ 0 & \text{else.} \end{cases} \tag{4.16}$$

Note that $r_i \in \{\pm 1\}$ and the term $(r_i)^2 = 1$ is provided to obtain a concise representation later. Using the above definition, we have that $g(\boldsymbol{x}) = \sum_i g_i(\boldsymbol{x})$, and the convexity of each

sub-function $g_i(\boldsymbol{x})$ renders the overall function $g(\boldsymbol{x})$ convex. Now, assuming $(\boldsymbol{\phi}_i^T\boldsymbol{x} - b_i) \neq 0$, the gradient of the sub-function $g_i(\boldsymbol{x})$ can be expressed as:

$$\begin{aligned}\nabla_{\boldsymbol{x}} g_i(\boldsymbol{x}) &= \rho(r_i(\boldsymbol{\phi}_i^T\boldsymbol{x} - b_i))(-r_i\boldsymbol{\phi}) \\ &= \begin{cases} r_i\left(\boldsymbol{\phi}_i^T\boldsymbol{x} - b_i\right)(-r_i\boldsymbol{\phi}_i) & \text{if } r_i\left(\boldsymbol{\phi}_i^T\boldsymbol{x} - b_i\right) < 0, \\ 0 & \text{else.} \end{cases}\end{aligned} \tag{4.17}$$

Furthermore, note that $g_i(\boldsymbol{x})$ is convex, and hence, in the case of $(\boldsymbol{\phi}^T\boldsymbol{x} - b_i) = 0$, the set of subgradients of the function are given by the convex hull $\{\lambda\left(\boldsymbol{\phi}_i^T\boldsymbol{x} - b_i\right)(-\boldsymbol{\phi}_i) : \ \lambda \in [0,1]\} \ni \rho(r_i(\boldsymbol{\phi}_i^T\boldsymbol{x} - b_i))(-r_i\boldsymbol{\phi})$ (i.e., the term (Equation 4.17) is a subgradient as well). Therefore, the gradient of the overall objective function $g(\boldsymbol{x})$ can be compactly expressed as:

$$\nabla_{\boldsymbol{x}} g(\boldsymbol{x}) = \sum_i \nabla_{\boldsymbol{x}} g_i(\boldsymbol{x}) = -(\boldsymbol{R}\boldsymbol{\Phi})^T \rho\left(\boldsymbol{R}\left(\boldsymbol{\Phi}\boldsymbol{x} - \boldsymbol{b}\right)\right). \tag{4.18}$$

The final step is to derive the proximal operator for the function $h(\boldsymbol{x})$. The proximal operator can be analytically derived for many convex functions including the one considered in this work. In particular, the proximal mapping for the function $h(\boldsymbol{x}) = \|\boldsymbol{x}\|_1$ is given by the

well-known (element-wise) *soft-thresholding* function (defined below), which recasts the overall update equation given in (Equation 4.14) for solving our problem $\mathcal{P}_2$ as follows:

$$\boldsymbol{x}_{i+1} = \text{prox}_{\alpha h}\left(\tilde{\boldsymbol{t}}_i\right) \tag{4.19}$$

$$= \text{sign}\left(\tilde{\boldsymbol{t}}_i\right) \odot \max\left\{|\tilde{\boldsymbol{t}}_i| - (\delta/\alpha)\mathbf{1}, 0\right\}, \text{ and} \tag{4.20}$$

$$\tilde{\boldsymbol{t}}_i = \boldsymbol{x}_i - \delta \nabla_{\boldsymbol{x}} g(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x}_i}, \tag{4.21}$$

where all the functions above are applied element-wise on the vector argument. Generally speaking, the performance and the convergence of the above iterations depends heavily on the choice of step-size δ and the thresholding factors α. In the following section, we show how the above iterations can be unfolded onto the layers of a deep neural network allowing for obtaining an enhanced first-order optimizer.

The proposed algorithm for solving the optimization problem $\mathcal{P}_2$ associated with the incorporation of non-zero quantization thresholds is summarized below.

The Proposed Generalized RFP Iterations:

$$\tilde{\boldsymbol{d}}_i = -\left(\boldsymbol{R}\boldsymbol{\Phi}\right)^T \rho\left(\boldsymbol{R}\left(\boldsymbol{\Phi}\boldsymbol{x}_{i-1} - \boldsymbol{b}\right)\right), \tag{4.22a}$$

$$\tilde{\boldsymbol{t}}_i = \boldsymbol{x}_{i-1} - \delta\tilde{\boldsymbol{d}}_i, \tag{4.22b}$$

$$\boldsymbol{x}_i = \text{sign}\left(\tilde{\boldsymbol{t}}_i\right) \odot \left(|\tilde{\boldsymbol{t}}_i| - (\delta/\alpha)\mathbf{1}\right), \tag{4.22c}$$

where (Equation 4.22a) corresponds to computing the gradient at the current point, while the step (Equation 4.22b) can be viewed as taking a descent step on the one-sided ℓ_2-norm using the obtained gradient, and (Equation 4.22c) corresponds to applying the proximal mapping operator. In the rest of this paper, we refer to the iterations presented in (Equation 4.22) as *Generalized RFPI* (G-RFPI).

4.2.2 Binary Iterative Hard Thresholding Algorithm (BIHT)

The BIHT algorithm is a simple, yet powerful, first-order iterative reconstruction algorithm for the problem of one-bit CS where the sparsity level K is assumed to be known a priori. BIHT iterations can be seen as a simple modification of the iterative hard thresholding (IHT) algorithm proposed in [134]. Similar to the RFPI algorithm, the BIHT method considers a zero-level quantization threshold. However, in contrast to the RFPI algorithm, it exploits the knowledge of the sparsity level K of the signal of interest. In other words, the BIHT algorithm is designed to tackle the following counterpart of $\mathcal{P}_0$:

$$\mathcal{P}_3: \quad \min_{\boldsymbol{x}\in\mathbb{R}^n} \ \|\rho\left(\boldsymbol{R}\boldsymbol{\Phi}\boldsymbol{x}\right)\|_1, \quad \text{s.t.} \quad \|\boldsymbol{x}\|_0 = K, \ \ \|\boldsymbol{x}\|_2 = 1, \tag{4.23}$$

where $\rho(\boldsymbol{c}) = \max\{-\boldsymbol{c}, \boldsymbol{0}\}$ and $\boldsymbol{R} = \text{Diag}(\boldsymbol{r})$ as before. Note that the one-sided ℓ_1 objective function above (also related to the hinge-loss) enforces the consistency principle previously introduced in (Equation 4.6), and that by solving the above optimization problem, we are working to achieve *maximal consistency* with the one-bit measurements $\boldsymbol{r}$. It is worth mentioning that one can also consider different objective functions, and not necessarily an ℓ_1 objective, as long

as it promotes the data consistency principle (e.g., ℓ_2 norm). For a detailed analysis of different candidates for the objective function and their properties, see [134].

The BIHT iterations are described as follows. Let $\boldsymbol{c} = \boldsymbol{R}\boldsymbol{\Phi}\boldsymbol{x}$, and define $\mathcal{F}(\boldsymbol{c}) = \|\rho(\boldsymbol{c})\|_1$. Furthermore, let $\partial\mathcal{F}(\boldsymbol{x}) := \{\boldsymbol{f} : \mathcal{F}(\boldsymbol{x}) - \mathcal{F}(\boldsymbol{y}) \geq \langle \boldsymbol{f}, \boldsymbol{x} - \boldsymbol{y}\rangle\}$ denotes the sub-differential set of $\mathcal{F}$ at the point $\boldsymbol{x} \in \mathbb{R}^n$. Then, given an initial point $\boldsymbol{x}_0$, the sparsity level K, and one-bit measurements $\boldsymbol{r}$ (or equivalently $\boldsymbol{R}$), at the i-th iteration, the BIHT algorithm updates the current estimate of the signal $\boldsymbol{x}_i$ through the following steps:

$$\boldsymbol{u}_{i+1} = \boldsymbol{x}_i - \delta \boldsymbol{f}_i = \boldsymbol{x}_i + \frac{\delta}{2}\boldsymbol{\Phi}^T \left(\boldsymbol{r} - \operatorname{sign}(\boldsymbol{\Phi}\boldsymbol{x}_{i-1})\right), \tag{4.24a}$$

$$\boldsymbol{x}_{i+1} = \mathcal{H}_K \left(\boldsymbol{u}_i\right), \tag{4.24b}$$

where $\boldsymbol{f}_{i-1} \in \partial\mathcal{F}(\boldsymbol{x})$ denotes a sub-gradient of the one-side ℓ_1 objective function in $\mathcal{P}_3$ at $\boldsymbol{x}_i$, $\delta > 0$ governs the fixed gradient step-size, and the projection operator $\mathcal{H}_K(\boldsymbol{x})$ is defined such that it retains the largest K elements (in magnitude) of the vector argument, and set the rest of the elements to zero. Note that the term $(-1/2)\boldsymbol{\Phi}^T(\boldsymbol{r} - \operatorname{sign}(\boldsymbol{\Phi}\boldsymbol{x}))$ is a sub-gradient of the objective function at point $\boldsymbol{x}$ (more details is provided in subsection B.1 below).

The step (Equation 4.24a) can be interpreted as taking a descent step using the computed sub-gradient of the objective function (Equation 4.23), while the projection step in (Equation 4.24b) can be viewed as a projection of $\boldsymbol{u}_i$ onto the support set of K-sparse signals. Once the above iterations terminate either by fully satisfying the consistency principle (i.e., obtaining $\boldsymbol{x}^*$ such that $\mathcal{F}(\boldsymbol{x}^*) = 0$), or by achieving a maximum number of iterations,

the ultimate step to be taken is projecting the final estimate $\boldsymbol{x}^*$ onto the unit-sphere, viz. $\boldsymbol{x}^* \leftarrow \boldsymbol{x}^*/\|\boldsymbol{x}^*\|_2$. Note that this is in contrast to the RFPI algorithm as the BIHT iterations does not require a normalization step as in (Equation 6.8) at each iteration.

B.1. Extending the BIHT framework to non-zero quantization thresholds:

The extension of the BIHT iterations to incorporate the non-zero thresholds vector $\boldsymbol{b}$ is straight-forward. In the case of non-zero quantization thresholds, we cast the signal recovery problem as

$$\min_{\boldsymbol{x}\in\mathbb{R}^n} \ \mathcal{F}(\boldsymbol{x}) \equiv \|\rho\left(\boldsymbol{R}(\boldsymbol{\Phi}\boldsymbol{x}-\boldsymbol{b})\right)\|_1, \quad \text{s.t.} \quad \|\boldsymbol{x}\|_0 = K, \tag{4.25}$$

where $\boldsymbol{R} = \text{Diag}(\boldsymbol{r})$ and $\boldsymbol{r} = \text{sign}(\boldsymbol{\Phi}\boldsymbol{x}-\boldsymbol{b})$. Further note that the unit-ball constraint has been dropped due to the fact that the amplitude ambiguity is resolved by employing non-zero quantization thresholds.

In order to tackle the optimization problem (Equation 4.25), we make use of the well-known iterative hard thresholding algorithm (IHT) extensively used in the literature for general CS problems. In particular, the updating steps of the iterative hard thresholding algorithm is consist of taking a descent step on the objective function in (Equation 4.25) followed by a projection operator denoted by $\mathcal{H}_k$. Starting from a K-sparse initial point $\boldsymbol{x}_0$, the overall updating equation of the IHT algorithm for recovering the underlying K-sparse solution is given by:

$$\boldsymbol{x}_{i+1} = \mathcal{H}_k\left(\boldsymbol{x}_i - \delta\boldsymbol{f}_i\right), \tag{4.26}$$

where $\boldsymbol{f}_i \in \partial\mathcal{F}(\boldsymbol{x})$ at $\boldsymbol{x}_i$, and $\delta > 0$ denotes the step-size. Similar to the steps we took in (Equation 4.9)-(Equation 4.22), and extending the analysis provided in [104], we define $\mathcal{F}(\boldsymbol{x}) = \sum_i f_i(\boldsymbol{x})$, where the convex sub-function f_i is given by:

$$f_i(\boldsymbol{x}) = |[\rho\left(\boldsymbol{R}\left(\boldsymbol{\Phi}\boldsymbol{x} - \boldsymbol{b}\right)\right)]_i| \tag{4.27}$$

$$= \begin{cases} |\boldsymbol{\phi}_i^T\boldsymbol{x} - b_i| & \text{if } r_i\left(\boldsymbol{\phi}_i^T\boldsymbol{x} - b_i\right) < 0, \\ 0 & \text{else.} \end{cases} \tag{4.28}$$

Assuming $(\boldsymbol{\phi}_i^T\boldsymbol{x} - b_i) \neq 0$, the gradient of the sub-function f_i can be expressed as follows:

$$\nabla_{\boldsymbol{x}} f_i(\boldsymbol{x}) = -\frac{1}{2}\left(r_i - \text{sign}(\boldsymbol{\phi}_i^T\boldsymbol{x} - b_i)\right)\boldsymbol{\phi}_i \tag{4.29}$$

$$= \begin{cases} \text{sign}(\boldsymbol{\phi}_i^T\boldsymbol{x} - b_i)\boldsymbol{\phi}_i & \text{if } r_i\left(\boldsymbol{\phi}_i^T\boldsymbol{x} - b_i\right) < 0, \\ 0 & \text{else.} \end{cases}$$

In the case of $(\boldsymbol{\phi}_i^T\boldsymbol{x} - b_i) = 0$, since the sub-function f_i is convex, we have the sub-gradients of $f_i(\boldsymbol{x})$ given by the set

$$\partial f_i(\boldsymbol{x}) = \left\{-\frac{\lambda}{2}(r_i - \text{sign}(\phi_i^T\boldsymbol{x} - b_i))\boldsymbol{\phi}_i : \lambda \in [0,1]\right\} \tag{4.30}$$

$$\ni -\frac{1}{2}\left(r_i - \text{sign}(\boldsymbol{\phi}_i^T\boldsymbol{x} - b_i)\right)\boldsymbol{\phi}_i. \tag{4.31}$$

Hence, considering that $\mathcal{F}(\boldsymbol{x}) = \sum_i f_i(\boldsymbol{x})$, we have

$$\boldsymbol{f} = -\frac{1}{2}\boldsymbol{\Phi}^T \left(\boldsymbol{r} - \text{sign}(\boldsymbol{\Phi}\boldsymbol{x} - \boldsymbol{b})\right) \in \partial\mathcal{F}(\boldsymbol{x}). \tag{4.32}$$

Consequently, the updating steps of the IHT algorithm defined in (Equation 4.26) for solving the optimization problem (Equation 4.25) can be expressed as:

The Proposed Generalized BIHT Iterations:

$$\boldsymbol{u}_i = \boldsymbol{x}_{i-1} + \frac{\delta}{2}\boldsymbol{\Phi}^T \left(\boldsymbol{r} - \text{sign}(\boldsymbol{\Phi}\boldsymbol{x}_{i-1} - \boldsymbol{b})\right), \tag{4.33a}$$

$$\boldsymbol{x}_i = \mathcal{H}_K \left(\boldsymbol{u}_i\right), \tag{4.33b}$$

Note the exception that in the proposed generalized BIHT iterations, there is no need for the normalization of the obtained estimate of the signal $\boldsymbol{x}^*$ after the update steps terminate. This is due to the fact that a non-zero quantization threshold vector is employed at time of the encoding, and hence, the amplitude information is not fully lost. In the rest of this paper, we refer to the above iterations as *Generalized BIHT* (G-BIHT) algorithm.

Although simple and powerful, the BIHT algorithm requires a careful choice of the gradient step-size δ for convergence and stability, and there is no straight-forward method to properly choose the gradient step-size. Moreover, it only utilizes a fixed step-size through all iterations. This motivates the development of a methodology by which one can design a decoder function that exploits adaptive gradient step-sizes, which can result in a significant improvement of the performance of the BIHT algorithm.

In the next section, we discuss a slight over-parametrization of the iterations of RFPI, G-RFPI, BIHT, and G-BIHT algorithms that paves the way for the design of our proposed one-bit compressive AE and for jointly designing the parameters of the encoder function defined in (Equation 4.3) parametrized on the sensing matrix $\boldsymbol{\Phi}$, the quantization thresholds $\boldsymbol{b}$, and the design of a set of decoder functions based on the discussed iterative optimization algorithms.

4.3 The Proposed Model-Based Deep Learning Models for One-Bit CS

We pursue the design of a novel *model-driven one-bit compressive sensing-based autoencoder* deep architecture that facilitates the joint design of the parameters of both the encoder and the decoder module when one-bit quantizers with non-zero thresholds are employed in the data acquisition process (i.e., the encoding module) for a K-sparse input signal $\boldsymbol{x} \in \mathbb{R}^n$. In particular, the decoder module can be seen as model-based deep architecture which is derived upon unfolding the iterations of first-order optimization techniques provided above onto the layers of a deep neural network.

In general terms, a AE is a generative model comprised of an encoder and a decoder module that are sequentially connected together. The purpose of an AE is to learn an abstract representation of the input data, while providing a powerful data reconstruction system through the decoder module. The input to such a system is a set of signals following a certain distribution, i.e. $\boldsymbol{x} \sim \mathcal{D}(\boldsymbol{x})$, and the output is the recovered signal from the decoder module $\hat{\boldsymbol{x}}$. Hence, the goal is to jointly learn an abstract representation of the underlying distribution of the signals through the encoder module, and simultaneously, learning a decoder module allowing for reconstruction of the compressed signals from the obtained abstract representations. Therefore,

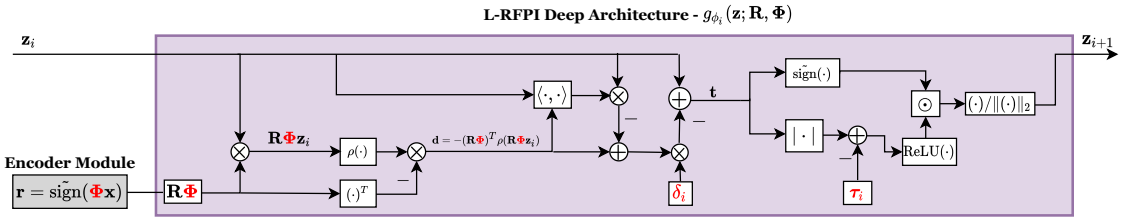
an AE can be defined by two main functions: *i*) an encoder function $f^{\text{Encoder}}_{\boldsymbol{\Upsilon}_1} : \mathbb{R}^n \mapsto \mathbb{R}^m$, parameterized on a set of variables $\boldsymbol{\Upsilon}_1$ that maps the input signal into a new vector space, and *ii*) a decoder function $f^{\text{Decoder}}_{\boldsymbol{\Upsilon}_2} : \mathbb{R}^m \mapsto \mathbb{R}^n$ parameterized on $\boldsymbol{\Upsilon}_2$, which maps the output of the encoder module back into the original signal space. Hence, the governing dynamics of a general auto-encoder can be expressed as $\hat{\boldsymbol{x}} = f^{\text{Decoder}}_{\boldsymbol{\Upsilon}_2} \circ f^{\text{Encoder}}_{\boldsymbol{\Upsilon}_1}(\boldsymbol{x})$, where $\hat{\boldsymbol{x}}$ denotes the reconstructed signal.

In light of the above, we seek to interpret a one-bit CS system as an AE module facilitating not only the design of the sensing matrix $\boldsymbol{\Phi}$ and the quantization thresholds $\boldsymbol{b}$ that best captures the information of a K-sparse signal when one-bit quantizers are employed, but also to learn the parameters of an iterative optimization algorithm specifically designed for the task of signal recovery. To this end, we modify and unfold the iterations of the proposed G-RFPI algorithm defined in (Equation 4.22), and the GBIHT method defined in (Equation 4.33) onto the layers of a deep neural network and later use the deep learning tools to tune the parameters of the proposed one-bit compressive AE.

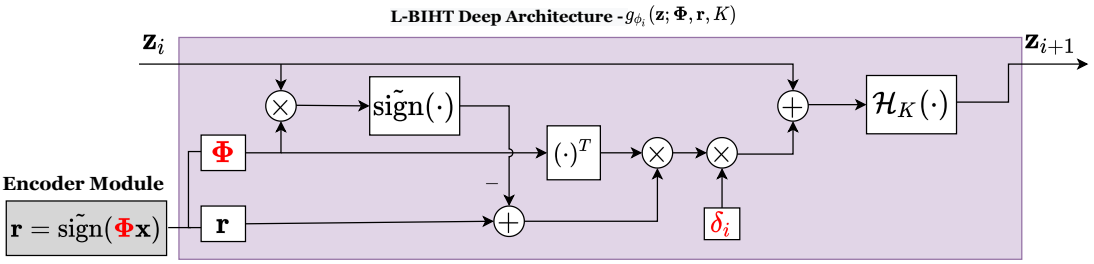
4.3.1 Structure of the Encoding Module

In its most general form, we define the encoder module of the proposed AE based on our data-acquisition model defined in (Equation 4.3), as follows:

$$f^{\text{Encoder}}_{\boldsymbol{\Upsilon}_1}(\boldsymbol{x}) = \tilde{\text{sign}}(\boldsymbol{\Phi}\boldsymbol{x} - \boldsymbol{b}), \tag{4.34}$$



(a)



(b)

Figure 12. An illustration of the computation dynamics in i-th layer of the proposed (a) L-RFPI and (b) L-BIHT deep architectures, in which the trainable parameters are highlighted in red color.

where $\mathbf{\Upsilon}_1 = \{\mathbf{\Phi}, \boldsymbol{b}\}$ denotes the set of learnable parameters of the encoder function, and $\tilde{\text{sign}}(\boldsymbol{x}) = \tanh(t \cdot \boldsymbol{x})$, for a large $t > 0$ (t was set to 50 in numerical investigations). Note that we replaced the original sign function with a smooth differentiable approximation of it based on the hyperbolic tangent function due to the fact that the sign function is not continuous and its gradient is zero everywhere except at the origin. Hence, the use of it would cripple stochastic gradient-based optimization methods (later used in back-propagation method for deep learning).

4.3.2 Structure of the Decoding Module

In this part, we describe the different scenarios under which we pursue the design of our decoder function by using the RFPI, BIHT, and the suggested G-RFPI and G-BIHT iterations. In particular, we fix the total complexity of our decoding module by fixing the total number of iterations allowed for the mentioned optimization iterations. Next, we slightly over-parameterize each iteration/step of the mentioned algorithms to increase the per-iteration degrees-of-freedom of each method and to further account for the learnable latent variables in the system. Finally, we unfold the iterations of each algorithm onto the layers of a deep architecture such that each layer of the deep network resembles one iteration of the base-line algorithm. We then seek to learn the parameters of both the decoder and encoder function using the training tools already developed for deep learning. We consider the following cases to design our decoder function:

• **Learned RFPI (L-RFPI)**: We consider the RFPI iterations defined in (Equation 4.8) as our base-line but slightly over-parametrize its iterations by introducing a gradient step-size δ_i and a *shrinkage* thresholds vector $\boldsymbol{\tau}_i$ *for each iteration* i. This is in contrast to the original RFP iterations where a fixed gradient step-size δ, and shrinkage threshold $\tau = (\delta/\alpha)\mathbf{1}$ were employed for all iterations. Hence, the proposed unfolded over-parametrized iterations are much more expressive. The decoder function will be parameterized on $\boldsymbol{\Upsilon}_2 = \{\delta_i, \boldsymbol{\tau}_i\}_{i=0}^{L-1}$, and the encoder function will be parametrized on the set $\boldsymbol{\Upsilon}_1 = \{\boldsymbol{\Phi}\}$ (note that $\boldsymbol{b} = \mathbf{0}$).

• **Learned BIHT (L-BIHT)**: We consider the unfolding of the iterations of the BIHT defined in (Equation 4.24) similar to the previous case and by introducing per-iteration gradient step-sizes δ_i in lieu of a fixed gradient-step size along all iterations. In this case, the decoder function

will be parametrized on the set $\boldsymbol{\Upsilon}_2 = \{\delta_i\}$, while the set of parameters of the encoding module is $\boldsymbol{\Upsilon}_1 = \{\boldsymbol{\Phi}\}$; both are to be learned.

• **Learned G-RFPI (LG-RFPI)**: We consider the unfolding of the proposed Generalized RFPI iterations in (Equation 4.22) in a non-zero quantization thresholds setting. We over-parameterize the iterations of the proposed G-RFPI by parametrizing the decoder function on the set $\boldsymbol{\Upsilon}_2 = \{\delta_i, \boldsymbol{\tau}_i\}_{i=0}^{L-1}$, and this time, by parameterizing the encoder function on both the sensing matrix and the quantization thresholds vector, i.e. $\boldsymbol{\Upsilon}_1 = \{\boldsymbol{\Phi}, \boldsymbol{b}\}$.

• **Learned G-BIHT (LG-BIHT)**: We consider the unfolding of the G-BIHT iterations defined in (Equation 4.33) in a similar manner, i.e. by parameterizing the decoder function on $\boldsymbol{\Upsilon}_2 = \{\delta_i\}_{i=0}^{L-1}$. However, similar to the previous case, we further parametrize the encoder function on the quantization thresholds vector in conjunction with the sensing matrix, i.e. $\boldsymbol{\Upsilon}_1 = \{\boldsymbol{\Phi}, \boldsymbol{b}\}$.

4.3.3 The Proposed One-Bit Compressive Autoencoding Approach

In the following, we describe the design of four novel deep architectures based on the above mentioned structures and discuss the governing dynamics of the proposed one-bit compressive sensing-based AE.

C.1. L-RFPI-Based Compressive Autoencoding:

In this case, we consider the following parameterized encoder function:

$$f_{\boldsymbol{\Upsilon}_1}^{\text{Encoder}}(\boldsymbol{x}) = \tilde{\text{sign}}(\boldsymbol{\Phi}\boldsymbol{x}), \quad \text{where } \boldsymbol{\Upsilon}_1 = \{\boldsymbol{\Phi}\}. \tag{4.35}$$

As for the decoder function, and based on the RFPI iterations in (Equation 4.8), define $g_{\phi_i} : \mathbb{R}^m \mapsto \mathbb{R}^n$ as follows:

$$g_{\phi_i}(\boldsymbol{z}; \boldsymbol{\Phi}, \boldsymbol{R}) = \frac{\boldsymbol{v}}{\|\boldsymbol{v}\|_2}, \quad \text{with} \tag{4.36a}$$

$$\boldsymbol{v} = \tilde{\text{sign}}\left(\boldsymbol{t}\right) \odot \left(|\boldsymbol{t}| - \boldsymbol{\tau}_i\right), \tag{4.36b}$$

$$\boldsymbol{t} = \left(1 + \delta_i \boldsymbol{d}^T \boldsymbol{z}\right) \boldsymbol{z} - \delta_i \boldsymbol{d}, \tag{4.36c}$$

$$\boldsymbol{d} = -\left(\boldsymbol{R}\boldsymbol{\Phi}\right)^T \rho\left(\boldsymbol{R}\boldsymbol{\Phi}\boldsymbol{z}\right), \tag{4.36d}$$

where $\phi_i = \{\boldsymbol{\tau}_i, \delta_i\}$ represents the parameters of the function g_{ϕ_i}, and $\boldsymbol{\tau}_i \in \mathbb{R}^n$ denotes the sparsity inducing *shrinkage* thresholds vector, and δ_i represents the gradient step-size at iteration i. Next, we define the proposed L-RFPI composite decoder function as follows:

$$f^{\text{Decoder}}_{\boldsymbol{\Upsilon}_2}(\boldsymbol{z}_0) = g_{\phi_{L-1}} \circ g_{\phi_{L-2}} \circ \cdots \circ g_{\phi_1} \circ g_{\phi_0}(\boldsymbol{z}_0; \boldsymbol{\Phi}, \boldsymbol{R}), \tag{4.37}$$

where $\boldsymbol{\Upsilon}_2 = \{\phi_i\}_{i=0}^{L-1}$ represents the learnable (tunable) parameters of the decoder function, and $\boldsymbol{z}_0$ is some initial point of choice. Note that we have over-parameterized the iterations of the RFPI algorithm by introducing the new variable $\boldsymbol{\tau}_i$ at each iteration for the sparsity inducing step in (Equation 4.36b). Moreover, in contrast with the original RFPI iterations, we have introduced a new step-size δ_i at each step of the iteration as well (see Eq. (Equation 4.36c)). Therefore, the above decoder function can be interpreted as performing L iterations of the original RFPI algorithm with an additional $L(n+1)-2$ degrees of freedom (as compared to the

base algorithm) expressed in terms of the set of the shrinkage thresholds $\boldsymbol{\tau}_i$ and the gradient step-sizes δ_i, i.e. $\{\boldsymbol{\tau}_i, \delta_i\}_{i=0}^{L-1}$. As a reslt, the proposed decoder function is much more expressive than that of the iterations of RFPI algorithm. A depiction of the computation dynamics of the i-th layer of the proposed L-RFPI deep architecture is provided in Fig. 12(a).

Remark: Note that the above encoder and decoder function, once cascaded together, can be viewed as a deep neural network with $(L+1)$ layers, where the dynamics of the first layer is described by the encoder function defined in (Equation 4.35), and the governing dynamics of the succeeding layers is described by computations of the form (Equation 4.36a)-(Equation 4.36d). Equivalently, such a deep architecture can be viewed as a computational graph with shared variables among the computation nodes, and thus, its parameters can be efficiently optimized by utilizing known deep learning tools such as back-propagation. Hence, the goal is to *jointly* learn the parameters of such a cascaded network (i.e., $\boldsymbol{\Upsilon}_1 \cup \boldsymbol{\Upsilon}_2$) in an end-to-end manner by using the available data at hand coming from the underlying distribution of the source signal $\boldsymbol{x}$. ■

C.2. L-BIHT-Based Compressive Autoencoding:

Similar to the previous case, we consider the same encoding function parametrized only on the learnable sensing matrix $\boldsymbol{\Phi}$ in a zero quantization thresholds setting, i.e. $\boldsymbol{\Upsilon}_1 = \{\boldsymbol{\Phi}\}$. The

governing equations for the decoder function in the case of the proposed Learned BIHT are as follows. We re-define $g_{\phi_i} : \mathbb{R}^m \mapsto \mathbb{R}^n$ as:

$$g_{\phi_i}(\boldsymbol{z}; \boldsymbol{\Phi}, \boldsymbol{r}, K) = \mathcal{H}_K(\boldsymbol{v}), \quad \text{for } i < L, \text{where} \tag{4.38a}$$

$$\boldsymbol{v} = \boldsymbol{z} + \delta_i \boldsymbol{\Phi}^T \left(\boldsymbol{r} - \tilde{\text{sign}}(\boldsymbol{\Phi}\boldsymbol{z})\right), \tag{4.38b}$$

with $\phi_i = \{\delta_i\}_{i=0}^{L-1}$, and where we have an added final layer $i = L$, to renormalize the reconstructed signal as,

$$g_{\phi_L}(\boldsymbol{z}; \boldsymbol{\Phi}, \boldsymbol{r}) = \frac{\boldsymbol{z}}{\|\boldsymbol{z}\|_2}. \tag{4.39}$$

Therefore, similar to the previous case, the proposed L-BIHT-based decoder function is defined as:

$$f_{\boldsymbol{\Upsilon}_2}^{\text{Decoder}}(\boldsymbol{z}_0) = g_{\phi_L} \circ g_{\phi_{L-1}} \circ \cdots \circ g_{\phi_1} \circ g_{\phi_0}(\boldsymbol{z}_0; \boldsymbol{\Phi}, \boldsymbol{R}, K). \tag{4.40}$$

We again observe the slight over-parametrization of the L-BIHT algorithm during the unfolding process. Namely, at each iteration we are introducing the per-iteration step-sizes δ_i to be learned which further enhances the performance of our iterations (see (Equation 4.38)). In this case, the decoder function is parameterized only on the gradient step-sizes, i.e. $\boldsymbol{\Upsilon}_2 = \{\delta_i\}_{i=0}^{L-1}$. The L-BIHT iterations have an additional $(L-1)$ degrees of freedom compared to that of the

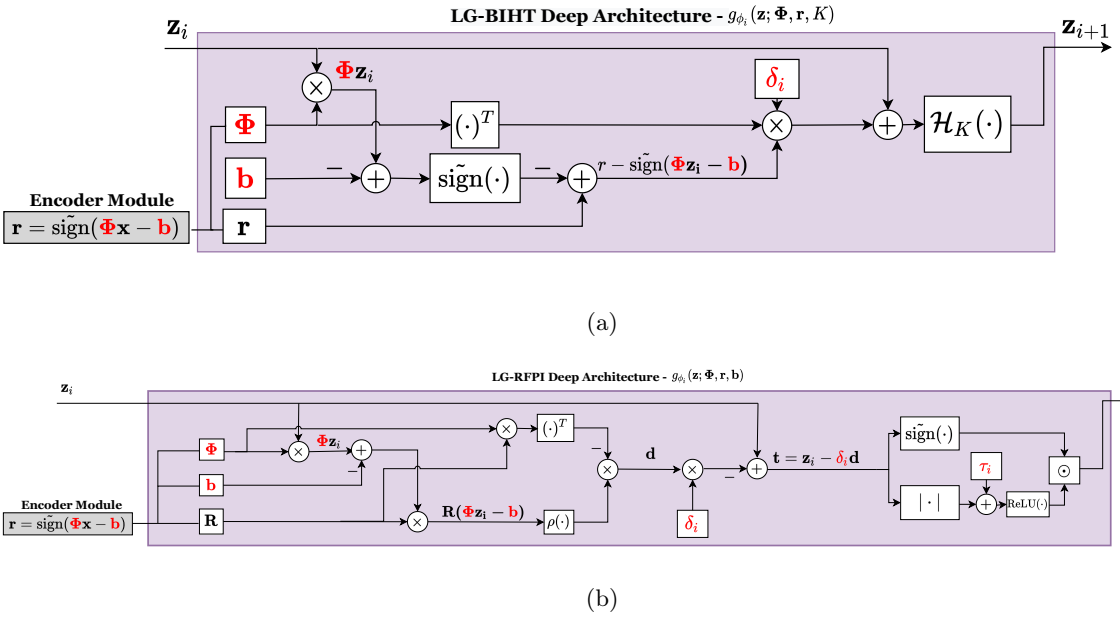


Figure 13. An illustration of the computation dynamics in the i-th layer of the proposed (a) LG-BIHT and (b) LG-RFPI deep architectures, in which the trainable parameters are highlighted in red color.

original BIHT iterations. Fig. 12(b) illustrates the computation dynamics of the i-th layer of the proposed L-BIHT deep architecture.

C.3. LG-RFPI-Based Compressive Autoencoding:

We consider the unfolding of iterations of the Learned Generalized RFPI method according to (Equation 4.22). As previously discussed, in the generalized iterations of both the RFPI and BIHT algorithms, the encoder module can be expressed as:

$$f_{\mathbf{\Upsilon}_1}^{\text{Encoder}}(\boldsymbol{x}) = \tilde{\text{sign}}(\mathbf{\Phi}\boldsymbol{x} - \boldsymbol{b}), \tag{4.41}$$

where $\boldsymbol{\Upsilon}_2 = \{\boldsymbol{\Phi}, \boldsymbol{b}\}$, and $\boldsymbol{b}$ represents the tunable vector of quantization thresholds. We follow a similar approach to the proposed L-RFPI-Based deep architecture and slightly over-parameterize the iterations in (Equation 4.22a)-(Equation 4.22c), leading to the design of the decoder function:

$$g_{\phi_i}(\boldsymbol{z}; \boldsymbol{\Phi}, \boldsymbol{R}, \boldsymbol{b}) = \tilde{\text{sign}}\left(\boldsymbol{v}\right) \odot \left(|\boldsymbol{v}| - \boldsymbol{\tau}_i\right), \quad \text{with} \tag{4.42a}$$

$$\boldsymbol{v} = \boldsymbol{z} - \delta_i \boldsymbol{d}, \tag{4.42b}$$

$$\boldsymbol{d} = -\left(\boldsymbol{R}\boldsymbol{\Phi}\right)^T \rho\left(\boldsymbol{R}\left(\boldsymbol{\Phi}\boldsymbol{z} - \boldsymbol{b}\right)\right), \tag{4.42c}$$

where $\phi_i = \{\boldsymbol{\tau}_i, \delta_i\}$ represents the parameters of the function g_{ϕ_i}, $\boldsymbol{\tau}_i \in \mathbb{R}^n$ denotes the sparsity inducing thresholds vector, and δ_i represents the gradient step-size at iteration i. Hence, the proposed decoder function $f^{\text{Decoder}}_{\boldsymbol{\Upsilon}_2}(\boldsymbol{z}_0)$ can be represented in the same way as in (Equation 4.37), with $\boldsymbol{\Upsilon}_2 = \{\phi_i\}_{i=0}^{L-1}$. Note that by incorporating the non-zero quantization thresholds, there is no need for an additional normalization term at each iteration. The above iterations (comprising the decoder function) have the same degree of freedom as L-RFPI iterations—an additional $L(n+1) - 2$ model parameters compared to that of the base-line G-RFPI iterations. Also, note the additional m degrees of freedom that the encoder function offers in terms of tunable quantization thresholds vector $\boldsymbol{b}$ (in addition to the sensing matrix). Fig. 13(b) illustrates the computation dynamics of the i-th layer of the proposed L-BIHT deep architecture.

C.4. LG-BIHT-Based Compressive Autoencoding:

We consider an encoder function $f^{\text{Encoder}}_{\boldsymbol{\Upsilon}_1}$ of the form (Equation 4.41), where $\boldsymbol{\Upsilon}_1 = \{\boldsymbol{\Phi}, \boldsymbol{b}\}$

denotes the learnable sensing matrix and arbitrary quantization thresholds. Additionally, we present an over-parameterization of the Genralized BIHT iterations (see Eqs. (Equation 4.33)) and consider the resulting unfolded network as the blueprint of our decoder. Namely, we define $g_{\phi_i} : \mathbb{R}^m \mapsto \mathbb{R}^n$ as:

$$g_{\phi_i}(\boldsymbol{z}; \boldsymbol{\Phi}, \boldsymbol{r}, \boldsymbol{b}, K) = \mathcal{H}_K(\boldsymbol{v}), \quad \text{with} \tag{4.43a}$$

$$\boldsymbol{v} = \boldsymbol{z} + \delta_i \boldsymbol{\Phi}^T \left(\boldsymbol{r} - \tilde{\text{sign}}(\boldsymbol{\Phi}\boldsymbol{z} - \boldsymbol{b})\right), \tag{4.43b}$$

where $\phi_i = \{\delta_i\}$ denotes the set of parameters of the function g_{ϕ_i}. Note that due to employing a non-zero thresholds vector, we do not need the additional normalization layer as in (Equation 4.39) for this case. Consequently, the decoder function $f^{\text{Decoder}}_{\boldsymbol{\Upsilon}_2}$ can be expressed in a similar manner as in (Equation 4.40), with $\boldsymbol{\Upsilon}_2 = \{\delta_i\}_{i=0}^{L-1}$. These iterations, similar to L-BIHT case, have an additional $(L-1)$ degrees of freedom compared to that of the base-line G-BIHT iterations; whereas, the encoder function has an additional m tunable parameters in terms of the one-bit quantization thresholds compared to that of the L-BIHT-based AE. Fig. 13(a) illustrates the computation dynamics of the i-th layer of the proposed LG-BIHT deep architecture.

In the next section, we discuss the training process of the above proposed one-bit compressive autoencoders. Particularly, we formulate a proper loss function that facilitates the training of such unfolded deep architectures, and for each model, we seek to jointly learn the set of

parameters of the entire network (i.e., the encoder and decoder function) in a end-to-end manner using the available deep learning techniques.

4.3.4 Loss Function Characterization and Training Method

The output of an autoencoder is the reconstructed signal from the compressed measurements, i.e.

$$\hat{\boldsymbol{x}} = f_{\boldsymbol{\Upsilon}_2}^{\text{Decoder}} \circ f_{\boldsymbol{\Upsilon}_1}^{\text{Encoder}}(\boldsymbol{x}),$$

where $\boldsymbol{x} \sim \mathcal{D}(\boldsymbol{x})$ and $\hat{\boldsymbol{x}}$ denotes the input and output of the AE, respectively. The training of an AE should be carried out by defining a proper loss function $\mathcal{G}\left(\boldsymbol{x}, f_{\boldsymbol{\Upsilon}_2}^{\text{Decoder}} \circ f_{\boldsymbol{\Upsilon}_1}^{\text{Encoder}}(\boldsymbol{x})\right)$ that provides a measure of the similarity between the input and the output of the AE. The goal is to minimize the distance between the input target signal $\boldsymbol{x}$ and the recovered signal $\hat{\boldsymbol{x}}$ according to a similarity criterion. A widely-used option for the loss function is the output MSE loss, i.e., $\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}(\boldsymbol{x})}\left\{\|\boldsymbol{x}-\hat{\boldsymbol{x}}\|_2^2\right\}$, and hence, the training loss of such a system can be formulated as:

$$\mathcal{G}(\boldsymbol{x};\hat{\boldsymbol{x}}) = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}(\boldsymbol{x})}\left\{\|\boldsymbol{x} - f_{\boldsymbol{\Upsilon}_2}^{\text{Decoder}} \circ f_{\boldsymbol{\Upsilon}_1}^{\text{Encoder}}(\boldsymbol{x})\|_2^2\right\}$$

that is to be minimized over $\boldsymbol{\Upsilon}_1$ and $\boldsymbol{\Upsilon}_2$. Nevertheless, in deep architectures with a high number of layers and parameters, such a simple choice of the loss function makes it difficult to back-propagate the gradients; in fact, the vanishing gradient problem arises. Therefore, for the training of the proposed AE, a better choice for the loss function is to consider the cumulative MSE loss of the layers. As a result, one can also feed-forward the decoder function for only $l < L$ layers (a lower complexity decoding), and consider the output of the l-th layer as a good

approximation of the target signal. For training, one needs to consider the constraint that the gradient step-sizes $\{\delta_i\}_{i=0}^{L-1}$, and the shrinkage thresholds $\{\boldsymbol{\tau}_i\}_{i=0}^{L-1}$ must be non-negative. By parameterizing the decoder function on the step-sizes and the shrinkage step thresholds, we need to regularize the training loss function ensuring that the network chooses positive step sizes and shrinkage thresholds at each layer. With this in mind, we suggest the following loss function for training the proposed one-bit compressive AE. Let $\tilde{g}_i = g_{\phi_i} \circ g_{\phi_{i-1}} \circ \cdots \circ g_{\phi_0} \circ f_{\boldsymbol{\Upsilon}_1}^{\text{Encoder}}(\boldsymbol{x})$, and define the loss function for training as

$$\mathcal{G}_L(\boldsymbol{x}; \hat{\boldsymbol{x}}) = \underbrace{\sum_{i=0}^{L-1} w_i ||\boldsymbol{x} - \tilde{g}_i(\boldsymbol{x}_i)||_2^2}_{\text{accumulated MSE loss of all layers}} + \underbrace{\lambda \sum_{i=0}^{L-1} (-[\boldsymbol{\delta}]_i) + \lambda \sum_{i=0}^{nL-1} (-[\boldsymbol{\tau}]_i)}_{\text{regularization term for the step-sizes and shrinkage thresholds}}, \tag{4.44}$$

where w_i denotes the importance weight of choice for the output of each layer, $\lambda > 0$, $[\boldsymbol{\delta}]_i = \delta_i$, and $\boldsymbol{\tau} = [\boldsymbol{\tau}_0^T, \ldots, \boldsymbol{\tau}_{L-1}^T]$. Note that as the information flows through the network, one expects that as we progress layer by layer, the reconstruction shows improvement. A reasonable weighting scheme for designing the importance weights w_i is to gradually increase the importance weights as we proceed through the layers. In this work, we consider a logarithmic weighting scheme, i.e. $w_i = \log(i+2)$, for $i \in \{0, \ldots, L-1\}$. Moreover, in training the autoencoders based on the BIHT algorithm, we exclude the last term in (Equation 4.44) as there is no shrinkage thresholds required for these models.

As for the training procedure, our numerical investigations showed that an incremental learning approach is most effective for training of the proposed networks. The details of the incremental learning method that we employed are as follows. During the l-th increment round (for $l \in \{0, \ldots, L-1\}$), we seek to optimize the cost function $\mathcal{G}_l(\boldsymbol{x}, \hat{\boldsymbol{x}})$ by learning the set of parameters $\boldsymbol{\Upsilon}_l = \boldsymbol{\Upsilon}_1 \cup \{\phi_i\}_{i=0}^{l}$. At each round l, we perform a batch learning with mini-batches of size B. After finishing the l-th round of training, the $(l+1)$-th layer will be added to the network, and the objective function will be changed to $\mathcal{G}_{l+1}(\boldsymbol{x}, \hat{\boldsymbol{x}})$. Next, the entire network will go through another batch-learning phase. Interestingly, in this method of training, the learned parameters from the l-th round $\boldsymbol{\Upsilon}_l$ will be used as the initial values of the same parameters in the next round.

4.4 Numerical Results

In this section, we present various simulation results to investigate the performance of the proposed one-bit compressive AEs and to further show the effectiveness of our training. For training purposes, we randomly generate K-sparse signals of length $n = 128$, i.e. $\boldsymbol{x} \in \mathbb{R}^{128}$ where the non-zero elements are sampled from $\mathcal{N}(0, 1)$. Furthermore, we fix the total number of layers of the decoder function to $L = 30$; equivalent of performing only 30 optimization iterations of the form (Equation 4.36), (Equation 4.38), (Equation 4.42), and (Equation 4.43). As for the sensing matrix (to be learned), we assume $\boldsymbol{\Phi} \in \mathbb{R}^{512 \times 128}$. The results presented here are averaged over 128 realizations of the system parameters. Similar to [102], we consider the case that $m > n$, due to the focus of this study on one-bit sampling where usually a large number of one-bit samples are available, as opposed to the usual infinite-precision CS settings.

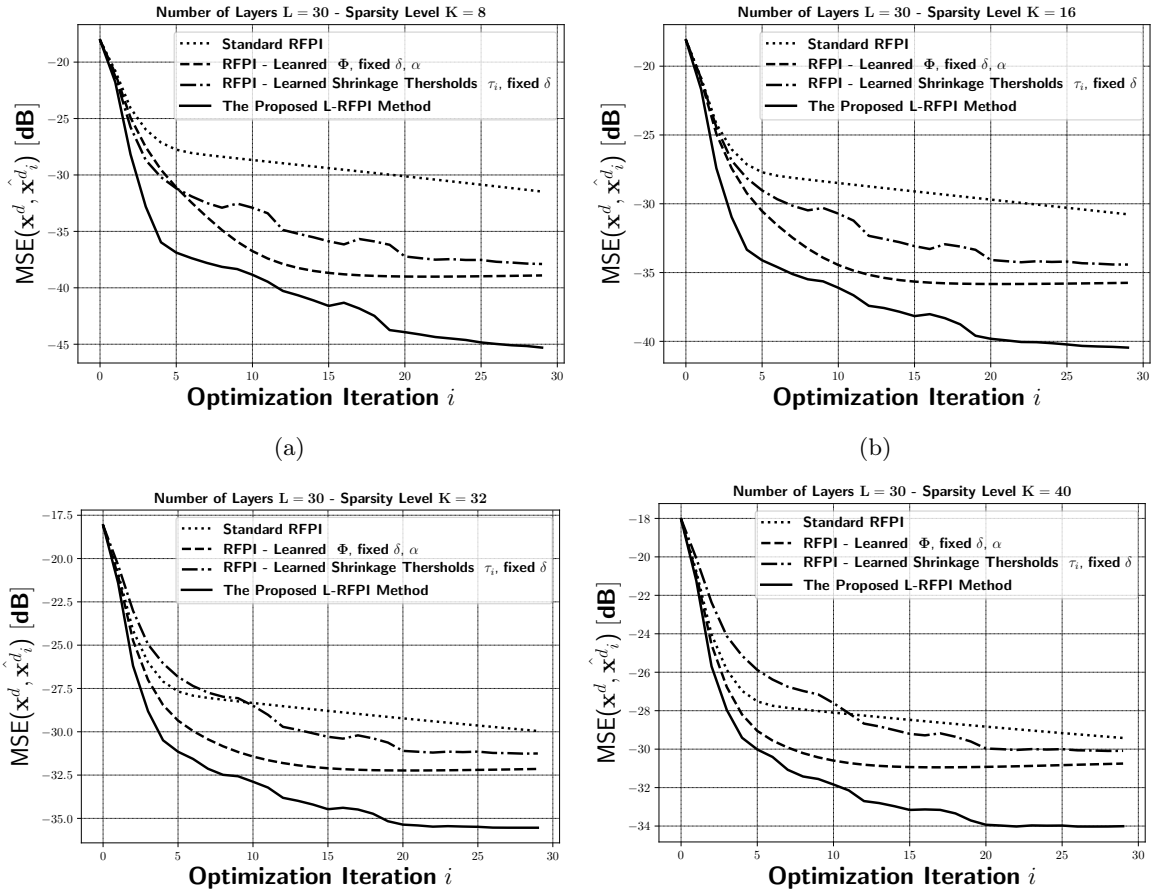


Figure 14. The performance of the proposed L-RFPI method compared to the RFPI algorithm for sparsity levels: (a) $K = 8$, (b) $K = 16$, (c) $K = 32$, and (d) $K = 40$.

The proposed one-bit CS AEs are implemented using the PyTorch library [9]. The Adam optimizer [84] with a learning rate of 10^{-3} is utilized for optimization of parameters of the

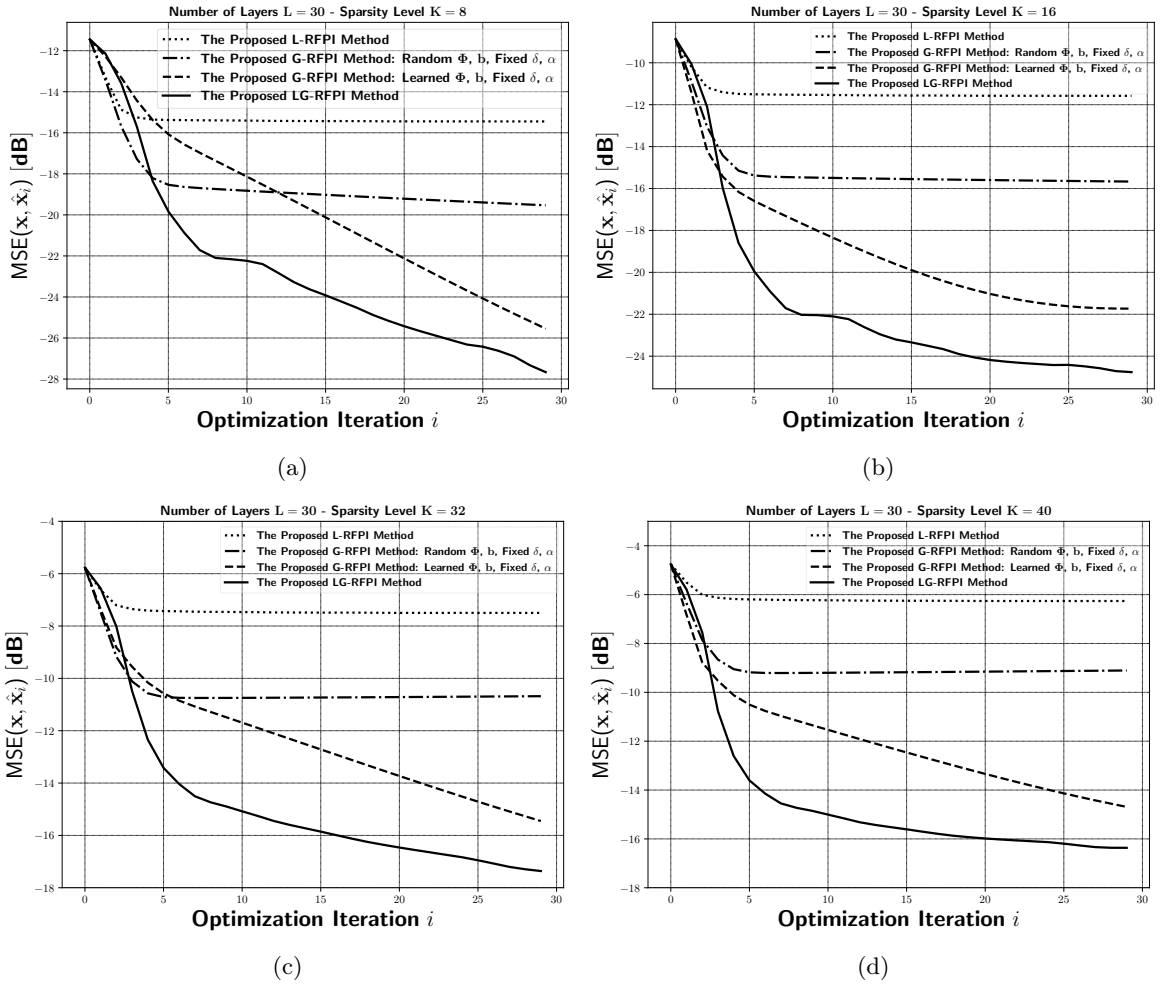


Figure 15. The performance of the proposed LG-RFPI AE and the proposed G-RFPI method in recovering the amplitude information of the K- sparse signals for sparsity levels: (a) $K = 8$, (b) $K = 16$, (c) $K = 32$, and (d) $K = 40$.

proposed deep architectures. Due to the importance of reproducible research, we have made all the codes implemented publicly available along with this paper.[1]

As it was previously discussed in Sec. 4.3.4, we employ an incremental batch-learning approach with mini-batches of size 64 at each round $l < 30$, and a total number of 200 epochs per round. For training of the the proposed AEs that are based on the RFPI iterations, i.e., the L-RFPI and LG-RFPI deep architectures, we uniformly sample the sparsity level of the source signal from the set $K \in \{16, 24, 32\}$ for each training point in the mini-batch. We evaluate the performance of the proposed methods on target signals with $K \in \{16, 32\}$, as well as $K \in \{8, 40\}$ (which was not presented to the network during the training phase). Moreover, due to the fact that the BIHT method and the corresponding one-bit AEs (L-BIHT and LG-BIHT) require the knowledge of the sparsity level of the source signal a priori, there is no need to train the network on various sparsity levels; i.e., the corresponding deep architectures can be trained for a particular K. Hence, for the L-BIHT and LG-BIHT deep architectures, we train the network for source signals with $K = 16$, and evaluate the performace of the resulted networks on $K \in \{16, 24\}$.

In the sequel, we refer to $\boldsymbol{s}^{\mathrm{d}} = \boldsymbol{s}/\|\boldsymbol{s}\|_2$ as the *normalized* version of the vector $\boldsymbol{s}$. In all scenarios, in order to have a fair comparison between the algorithms, the initial starting point $\boldsymbol{z}_0$ of the optimization algorithms are the same.

---

[1]The code is also available at: `https://github.com/skhobahi/deep1bitVAE`

Performance of the proposed L-RFPI AE:

In this part, we investigate the performance of the proposed L-RFPI-based AE in recovering the normalized source signal $\boldsymbol{x}$, i.e., recovering $\boldsymbol{x}_i^{\mathrm{d}}$.

Fig. Figure 14 illustrates Mean Squared Error (MSE) for normalized version of the recovered signal $\hat{\boldsymbol{x}}_i^{\mathrm{d}}$ versus total number of optimization iterations i, for $i \in \{0, \ldots, 29\}$, and for sparsity levels (a) $K = 8$, (b) $K = 16$, (c) $K = 32$, and (d) $K = 40$. We compare the performance of the proposed L-RFPI algorithm with the standard RFPI iterations in (Equation 4.8a)-(Equation 6.8), in the following scenarios:

• *Case 1*: The RFPI algorithm with a randomly generated sensing matrix whose elements are i.i.d. and sampled from $\mathcal{N}(0, 1)$, and fixed values for δ and α.

• *Case 2:* The RFPI algorithm where the learned $\boldsymbol{\Phi}$ is utilized, and the values for δ and α are fixed as in the previous case.

• *Case 3:* The RFPI algorithm with a randomly generated $\boldsymbol{\Phi}$ (same as Case 1), however, the learned shrinkage thresholds vector $\{\boldsymbol{\tau}_i\}_{i=0}^{L-1}$ is utilized with a fixed step-size.

• *Case 4:* The proposed one-bit L-RFPI AE method corresponding to the iterations of the form (Equation 4.36a)-(Equation 4.36d), with learned $\boldsymbol{\Phi}$, $\{\delta_i\}_{i=0}^{L-1}$, and $\{\boldsymbol{\tau}_i\}_{i=0}^{L-1}$.

To have a fair comparison, we fine-tuned the parameters of the standard RFPI method (Case 1), i.e., the step-size δ and the shrinkage threshold α, using a grid-search method. It can be seen from Fig. Figure 14 that in all cases of $K \in \{8, 16, 32, 40\}$, the proposed L-RFPI method demonstrates a significantly better performance than that of the RFPI algorithm (described in Case 1)—an improvement of ~ 10 times in MSE outcome. Furthermore, the effectiveness of

the learned $\mathbf{\Phi}$ (Case 2), and the learned $\{\boldsymbol{\tau}_i\}$ (Case 3) compared to the base algorithm (Case 1), are clearly evident, as both algorithms with learned parameters significantly outperform the original RFPI. Finally, although we trained the network for $K \in \{16, 24, 32\}$ sparse signals, it still shows very good generalization properties even for $K \in \{8, 40\}$ (see Fig. Figure 14 (a) and (d)). This is presumably due to the fact that the proposed L-RFPI-based AE is a *hybrid* model-based data-driven approach that exploits the existing domain knowledge of the problem as well as the available data at hand. Furthermore, note that the proposed method achieves a high accuracy very quickly and does not require solving (Equation 4.7) for several instances as opposed to the original RFPI algorithm—thus showing great potential for usage in real-time applications.

Performance of the proposed LG-RFPI AE :

Next, we investigate the performance of the proposed LG-RFPI AE (see Eqs. (Equation 4.42a)-(Equation 4.42c)) and the G-RFPI algorihtm (see Eqs. (Equation 4.22a)-(Equation 4.22c)) that we specifically designed for incorporating arbitrary quantization thresholds at data acquisiton. We investigate the performance of the proposed method in both cases of recovering the amplitude information as well as the normalized signal.

Fig. Figure 15 illustrates the MSE between the source signal $\boldsymbol{x}$ and the recovered signal $\hat{\boldsymbol{x}}_i$ versus total number of optimization iterations i, for $i \in \{0, \ldots, 29\}$, and for sparsity levels (a) $K = 8$, (b) $K = 16$, (c) $K = 32$, and (d) $K = 40$. Similar to the previous case, we consider the following scenarios:

- *Case 1*: The proposed G-RFPI algorithm with a randomly generated sensing matrix and

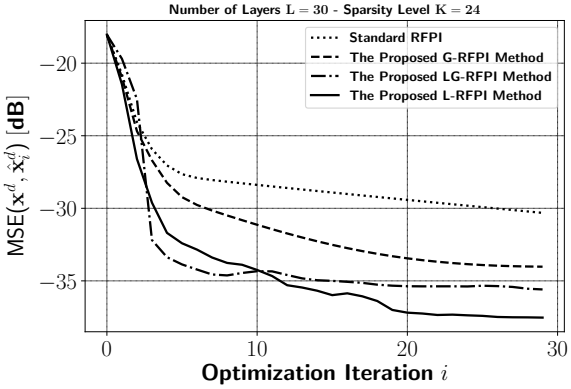


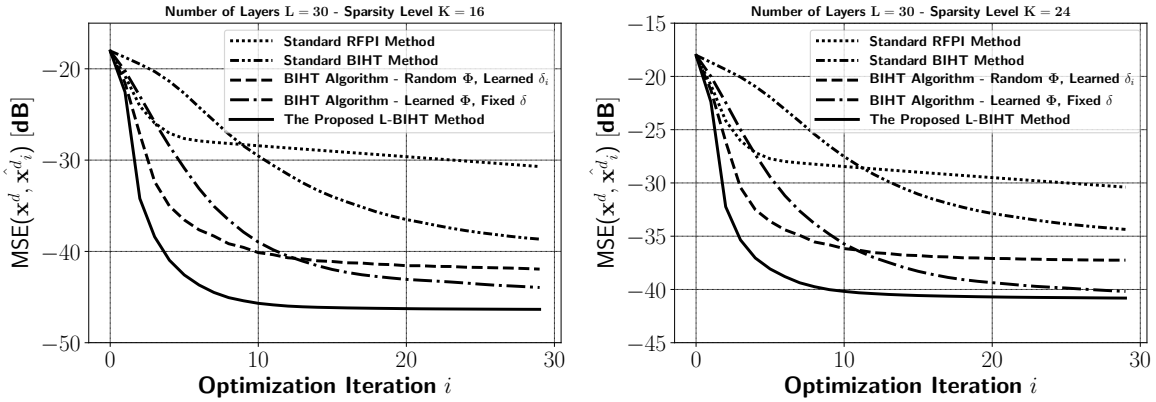
Figure 16. The performance of the proposed LG-RFPI AE and the proposed G-RFPI method in recovering the normalized K-sparse signals for sparsity level $K = 24$.

quantization threhsolds vector, whose elements are i.i.d. and sampled from $\mathcal{N}(0, 1)$, and fixed values for δ and α.

• *Case 2*: The proposed G-RFPI algorithm where the learned sensing matrix $\boldsymbol{\Phi}$ and quantization thresholds vector $\boldsymbol{b}$ are utilized, and the values for δ and α are fixed as in the previous case.

• *Case 3*: The proposed one-bit LG-RFPI AE method corresponding to the iterations of the form (Equation 4.42a)-(Equation 4.42c), with the learned $\boldsymbol{\Phi}$, $\boldsymbol{b}$, $\{\delta_i\}_{i=0}^{L-1}$, and $\{\boldsymbol{\tau}_i\}_{i=0}^{L-1}$.

Note that the focus of this part is on recovering the amplitude information of the underlying K-sparse signal by means of using arbitrary quantization thresholds. Although the RFPI method and the proposed L-RFPI AE can only recover the normalized signal $\boldsymbol{x}^d = \boldsymbol{x}/\|\boldsymbol{x}\|$, we further provide the performance of the L-RFPI method (that significantly outperforms the RFPI method) in recovering the amplitude information for comparison purposes. It can be observed from Fig. Figure 15 that the proposed G-RFPI algorithm with randomly generated sensing ma-



(a) (b)

Figure 17. The performance of the proposed L-BIHT method compared to the base-line BIHT algorithm for sparsity levels: (a) $K = 16$ and (b) $K = 24$.

trix and quantization thresholds (Case 1) provides good accuracy in recovering the amplitude information of the true signal for sparsity levels $K \in \{8, 16, 32, 40\}$. This is in contrast to the RFPI algorithm and the corresponding L-RFPI AE where the amplitude information is lost due to zero quantization thresholds. More precisely, the proposed G-RFPI algorithm outperforms the RFPI and the L-RFPI algorithm in terms of recovering the amplitude information of the signal. One can observe that even with a randomly generated quantization thresholds (i.e., without learning them), the proposed G-RFPI method achieve a significantly lower MSE in terms of recovering the amplitude information of the source signal as compared to the RFPI and the proposed L-RFPI method. Hence, the proposed G-RFPI method can be used as an stand-alone algorithm for one-bit compressive sensing settings with non-zero quantization thresholds, where both finding the direction of the source signal and the amplitude information is of great interest.

Next, we explore the effect of learning the distribution-specific (data-driven) sensing matrix and the quantization thresholds (Case 2). It is evident from Fig. Figure 15 that compared to the vanilla G-RFPI method, one can significantly achieve a lower MSE in terms of recovering the amplitude information by learning a proper sensing matrix and the quantization thresholds and utilizing them during the data-acquisition process. Finally, it can be seen from Fig. Figure 15 that the proposed LG-RFPI AE (Case 3) significantly outperforms its counterparts by achieving a much lower MSE very quickly. Moreover, the proposed LG-RFPI AE shows strong generalization properties for unseen sparsity levels $K \in \{8, 40\}$ (see Fig. Figure 15 (b) and (d)). The fact that such architectures show great performance in generalization is due to the model-driven nature of the proposed deep networks.

We conclude this part by comparing the performance of the proposed LG-RFPI, G-RFPI, and L-RFPI AEs in recovering the normalized version of the signal $\boldsymbol{x}$. Fig. Figure 16 illustrates the MSE between the normalized source signal and the recovered signal versus number of iterations i, i.e. MSE($\boldsymbol{x}^d$,$\hat{\boldsymbol{x}}_i^d$), for a sparsity level of $K = 24$. It can be observed from Fig. Figure 15 that the proposed methods outperform the standard RFPI iterations and achieve a high accuracy in recovering $\boldsymbol{x}^d$. Moreover, the proposed L-RFPI AE shows a slightly better performance than that of the LG-RFPI method. This is presumably due to the fact that the L-RFPI iterations and the corresponding deep architecture are specifically designed and tuned for recovering the normalized source signal while the proposed G-RFPI and LG-RFPI algorithms are designed for recovering the amplitude information of the source signal. Nevertheless, the MSE difference between the LG-RFPI and L-RFPI methods in recovering $\boldsymbol{x}^d$ is negligible, and hence, in a

non-zero quantization thresholds setting, it is beneficial to use the proposed LG-RFPI AE as it shows significant improvement in the performance of recovering the amplitude information while maintaining a high performance in recovering $\boldsymbol{x}^{\mathrm{d}}$ as well.

Performance of the proposed L-BIHT AE:

In this part, we investigate the performance of the proposed L-BIHT AE, and compare our results with the standard BIHT algorithm. Note that similar to the RFPI method and the proposed L-RFPI AE, the BIHT algorithm considers $\boldsymbol{b} = \mathbf{0}$ at the time of data acquisition. Hence, we investigate the performance of the proposed method in recovering the normalized source signal, i.e. $\boldsymbol{x}^{\mathrm{d}}$. In particular, we provide the simulation results for the following cases:

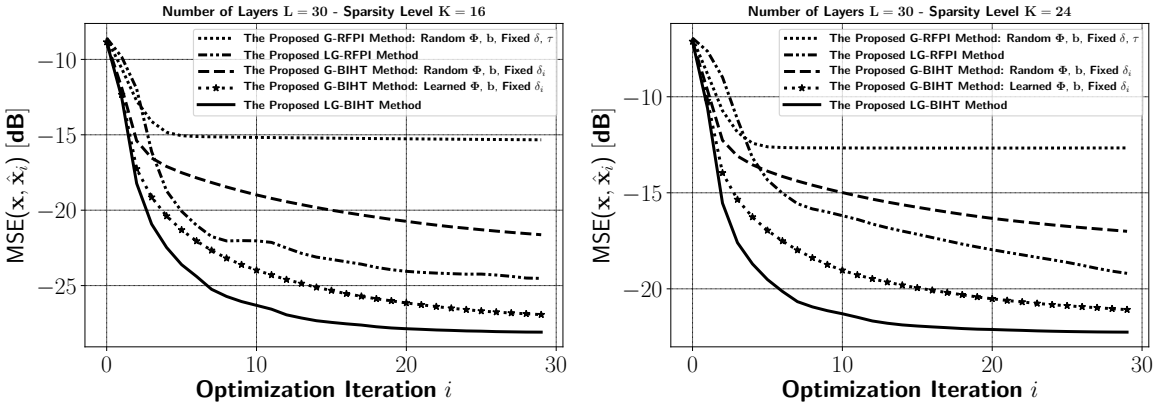
• *Case 1*: The BIHT algorithm with a randomly generated sensing matrix whose elements are i.i.d. and sampled from $\mathcal{N}(0, 1)$, and fixed value for δ.

• *Case 2*: The BIHT algorithm with a randomly generated $\boldsymbol{\Phi}$ (same as Case 1); however, learned gradient step-sizes δ_i are used at each iteration i.

• *Case 3*: The BIHT algorithm where the learned $\boldsymbol{\Phi}$ is utilized and the value for the step-size δ is fixed as in Case 1.

• *Case 4*: The proposed one-bit L-BIHT AE method corresponding to the iterations of the form (Equation 4.38a)-(Equation 4.38b), with the learned $\boldsymbol{\Phi}$ and $\{\delta_i\}_{i=0}^{L-1}$.

Fig. Figure 17 demonstrates the MSE between normalized source signal $\boldsymbol{x}^d$, and the recovered signal $\hat{\boldsymbol{x}}_i^{\mathrm{d}}$ versus the number of optimization iterations i, for signals with sparsity levels (a) $K = 16$ and (b) $K = 24$. Note that for learning the parameters of the proposed L-BIHT algorithm, we trained the corresponding deep architecture on the sparsity level $K = 16$, and



(a) (b)

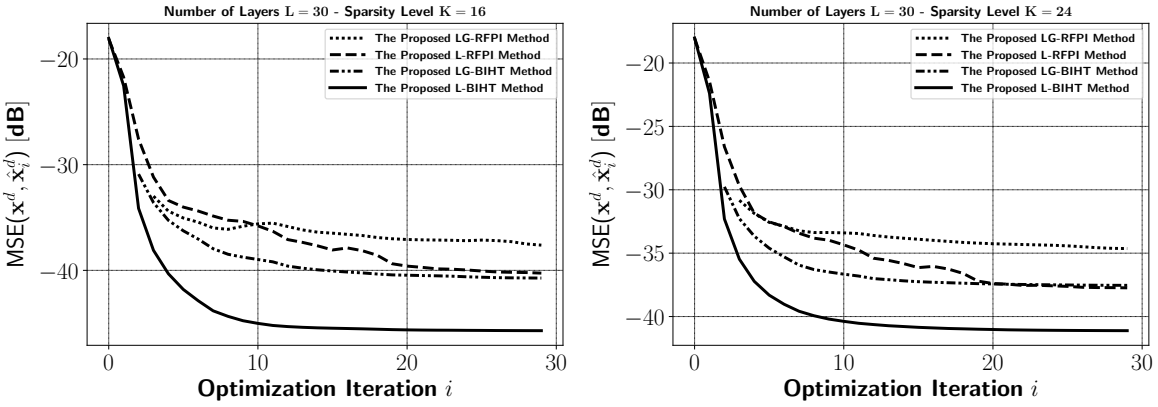
Figure 18. The performance of the proposed G-BIHT and the corresponding LG-BIHT AE in recovering the amplitude information of the signal for sparsity levels: (a) $K = 16$, and (b) $K = 24$.

we check the generalization performance of the learned parameters for the case of $K = 24$. It can be seen from Fig. Figure 17 that in both cases of $K \in \{16, 24\}$ the proposed L-BIHT algorithm demonstrates a significantly better performance than that of the standard BIHT algorithm (Case 1). Moreover, the effectiveness of the learned step-sizes $\{\delta_i\}_{i=0}^{L-1}$ (Case 2), and the learned sensing matrix $\mathbf{\Phi}$ (Case 3) compared to the base-line vanilla BIHT algorithm (Case 1) are evident. In particular, the learned step-sizes (Case 2) results in a fast descent while the learned $\mathbf{\Phi}$ (Case 3) leads to a lower MSE compared to Case 2. In addition, we provided the performance of the standard RFPI algorithm for comparison purposes. It can be seen from Fig. Figure 17 that the BIHT algorithm with and without the learned parameters achieves a better accuracy in recovering the direction of the source signal compared to the RFPI method. Also, a comparison between Fig. Figure 17 (a) and Fig. Figure 14 (b) reveals the fact that the proposed

L-BIHT AE demonstrates a far better performance than that of the proposed L-RFPI AE. This is due to the fact that the BIHT algorithm and the corresponding proposed L-BIHT AE, exploits the knowledge of the sparsity level K of the source signal (note the mapping function $\mathcal{H}_K$ used in (Equation 4.38a) and (Equation 4.24b)). One can further observe that even for the unseen case of $K = 24$, the proposed method generalizes very well and maintains its accuracy. This is due the model-driven nature of the proposed L-BIHT AE architecture. It is worth mentioning that it can be observed from Fig. Figure 17 that the proposed L-BIHT method converges very fast (in 10 iterations), achieving a high accuracy—making it a great candidate for real-time applications. Of course, the trade-off between using the L-RFPI and L-BIHT is implicit in the knowledge of the sparsity level of the signal. For applications where K is known beforehand, the proposed L-BIHT can be used in that it shows higher accuracy compared to the other methods. However, the L-RFPI methodology is more flexible as it does not require knowing the sparsity level of the signal a priori.

Performance of the proposed LG-BIHT AE:

Finally, we investigate the performance of the proposed G-BIHT method (see Eqs. (Equation 4.33a)-(Equation 4.33b)) and the corresponding one-bit compressive LG-BIHT AE (see Eqs. (Equation 4.43a)-(Equation 4.43b)) that are specifically designed to handle non-zero quantization thresholds $\boldsymbol{b}$. In particular, we are interested in evaluating the performance of the proposed methods in recovering the amplitude information of the source K-sparse signal. Hence, for this part, we check the MSE between the true signal $\boldsymbol{x}$, and the recovered signal $\hat{\boldsymbol{x}}_i$ from the G-BIHT and LG-BIHT methods for each iteration i. In addition, we provide the results for recovering the direction



(a) (b)

Figure 19. The performance of the proposed G-BIHT and the corresponding LG-BIHT AE in recovering the normalized signal, i.e. $\boldsymbol{x}^{\mathrm{d}}$, for sparsity levels: (a) $K = 16$, and (b) $K = 24$.

of the source signal $\boldsymbol{x}^{\mathrm{d}}$ as well. Specifically, we provide the simulation results for the following cases:

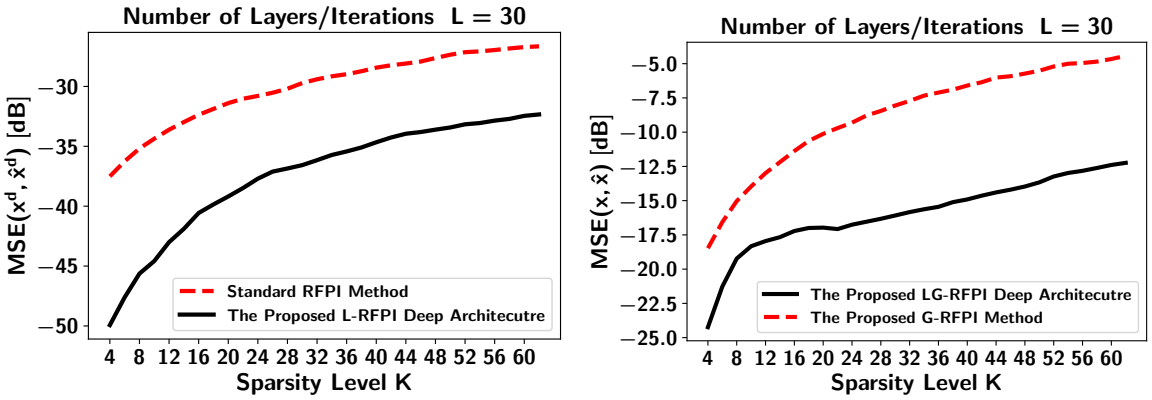
• *Case 1*: The proposed G-BIHT algorithm with a randomly generated sensing matrix and quantization thresholds vector where the elements of both are i.i.d. and sampled from $\mathcal{N}(0, 1)$, and fixed value for $\{\delta_i\}_{i=0}^{L-1}$.

• *Case 2*: The proposed G-BIHT algorithm, where the learned sensing matrix $\boldsymbol{\Phi}$ and quantization thresholds $\boldsymbol{b}$ are utilized and the values for $\{\delta_i\}_{i=0}^{L-1}$ are fixed as in the previous case.

• *Case 3*: The proposed one-bit LG-BIHT AE method corresponding to the iterations of the form (Equation 4.43a)-(Equation 4.43b), with learned $\boldsymbol{\Phi}$, $\boldsymbol{b}$ and $\{\delta_i\}_{i=0}^{L-1}$.

Fig. Figure 18 illustrates the MSE between the true signal $\boldsymbol{x}$ and the recovered signal $\hat{\boldsymbol{x}}_i$ versus optimization iteration i for sparsity levels (a) $K = 16$ and (b) $K = 24$. We further provide

the numerical results for the proposed LG-RFPI AE and the proposed G-RFPI iterations for comparison. It can be seen from Fig. Figure 18 that the proposed G-BIHT algorithm with randomly generated latent-variables (Case 1) significantly outperforms its G-RFPI counterpart, and achieves a high accuracy very quickly. On the other hand, the proposed LG-RFPI still achieves a lower MSE compared to the vanilla G-RFPI method. In addition, a comparison between the performance of the proposed G-BIHT algorithm with learned $\boldsymbol{\Phi}$ and $\boldsymbol{b}$ (Case 2) and the proposed LG-RFPI AE and vanilla G-BIHT (Case 1) reveals the effectiveness of the learned parameters and the power of the proposed G-BIHT algorithm. Namely, by utilizing only the learned $\boldsymbol{\Phi}$ and $\boldsymbol{b}$ and by using a fixed step size for the G-BIHT algorithm, one can achieve a superior performance than that of the LG-RFPI (where all of the learned variables are in use) and the vanilla G-BIHT method. Finally, it can be observed from Figure 17(a)-(b) that the proposed LG-BIHT algorithm (Case 3) significantly outperforms the other methods as it achieves a much lower MSE very quickly, specifically, compared to the proposed LG-RFPI AE. The superior performance of the LG-BIHT algorithm and the corresponding LG-BIHT AE is due the fact that we are exploiting the knowledge of the sparsity level K present in the signal. As discussed before, if the sparsity level is known a priori, it is beneficial to use either the G-BIHT algorithm (when one do not wish to perform any learning) or the proposed LG-BIHT methodology. It is worth mentioning that similar to the previously investigated methods, the proposed LG-BIHT generalizes very well for $K = 24$ (see Fig. Figure 18(b)) even though the sparsity level was not revealed to the network during the training phase.



(a) (b)

Figure 20. The generalization performance of (a) the proposed L-RFPI deep architecture compared to the standard RFPI method in recovering the direction information, and (b) the proposed LG-RFPI deep architecture and the proposed G-RFPI method in recovering the amplitude information of the K-sparse signals for sparsity levels $K \in \{4, 5, 6, \cdots, 64\}$.

Fig. Figure 19 demonstrate the MSE between the direction of the source signal, i.e. $\boldsymbol{x}^{\mathrm{d}}$, and the recovered direction $\hat{\boldsymbol{x}}^{\mathrm{d}}$ versus optimization iteration i, for sparsity levels of (a) $K = 16$ and (b) $K = 24$. It can be seen from Fig. Figure 19 that the proposed LG-BIHT method outperforms both the LG-RFPI method, and furthermore, it achieves a similar MSE to that of the proposed L-RFPI method. However, the convergence of LG-BIHT is much faster than that of the L-RFPI method. Furthermore, the proposed L-BIHT algorithm still achieves a superior performance than that of the other methods both in terms of convergence speed and accuracy. This is presumably due to the fact that the L-BIHT method is specifically designed and learned to have a high accuracy in finding normalized true signal $\boldsymbol{x}^{\mathrm{d}}$.

Generalization of the LG-RFPI and L-RFPI Methods:

TABLE I

NUMERICAL ANALYSIS OF $\mu(\boldsymbol{\Phi})$ AND THE GRAM MATRIX $\boldsymbol{M}$.

Metrics	**L-BIHT**	**LG-BIHT**	**L-RFPI**	**LG-RFPI**	**Random $\boldsymbol{\Phi}$**
$\mu(\boldsymbol{\Phi})$	0.0415	0.0904	0.0760	0.0852	0.2444
$\|\boldsymbol{M}-\boldsymbol{I}\|_F^2$	2.1716	8.7711	6.6117	8.0244	31.3032

In this part, we analyze the generalization performance of the proposed LG-RFPI and L-RFPI deep architectures. We consider the same simulation setup as in the previous cases, i.e. both architectures are assumed to have $L = 30$ layers. We performed the training of both architectures on a dataset consisting of K-sparse signals where the sparsity level is sampled uniformly from the set $K \in \{4, 8, 12, 16, 20\}$, and we evaluate the generalization performance of both architectures on K-source signals with sparsity levels $K \in \{4, 5, 6, \cdots, 64\}$.

Fig. 20(a) demonstrates the MSE between the direction of the true source signal, i.e. $\boldsymbol{x}^{\mathrm{d}}$, and the recovered direction $\hat{\boldsymbol{x}}^{\mathrm{d}}$ versus the sparsity level $K = \|\boldsymbol{x}\|_0$ for the proposed L-RFPI deep architecture. Moreover, Fig. 20(b) demonstrates the MSE between the the true source signal $\boldsymbol{x}$ and the recovered signal $\hat{\boldsymbol{x}}$ versus the sparsity level K for the proposed LG-RFPI deep architecture, and the proposed base-line G-RFPI algorithm (provided here for comparison purposes), both specifically designed to recover the amplitude information of the source signal. It can be seen from both Figs. 20(a) and 20(b) that the performance of the proposed L-RFPI and LG-RFPI methodologies is far superior to that of the standard model-based RFPI methods and the proposed G-RFPI algorithm across all sparsity levels. Interestingly, although the proposed deep architectures have been trained only on the sparsity levels $K \in \{4n\}_{n=1}^{5}$, they generalized

very well to higher sparsity levels as well, while outperforming the model-based algorithms. Such a good generalization performance is expected due to the model-based nature of the proposed architecture. This is in contrast to the conventional black-box deep learning models where the generalization performance usually degrades significantly as the input deviates from the distribution of the data points considered for training.

Coherence analysis of the learned sensing matrices:

In this part, we give a detailed analysis of the quality of the learned sensing matrices obtained by employing each of the proposed methodologies.

In order to quantify the quality of the learned task-specific sensing matrices, we make use of the mutual coherence metric defined in (Equation 4.2) as a figure of merit for the learned sensing matrices by the proposed methodologies. Fig. Figure 21 demonstrates the distribution of the mutual coherence coefficients of the sensing matrices obtained by the proposed deep architecture as well as the mutual coherence coefficients of a randomly generated sensing matrix (used in the previous numerical results). Furthermore, a detailed numerical analysis of $\mu(\mathbf{\Phi})$ and the Gram matrix $\boldsymbol{M}$ is provided in Table Table I. It can be clearly observed from Fig. Figure 21 and Table Table I that the proposed methodologies result in task-specific sensing matrices with considerably lower coherence coefficients as compared to that of a randomly generated one. In Particular, the mutual coherence $\mu(\mathbf{\Phi})$ is significantly lower for the proposed methodologies as compared to a purely random $\mathbf{\Phi}$. These observations also support the superior performance of the proposed methodologies in terms of signal reconstruction accuracy. In addition, the Gram matrix associated with the learned sensing matrices admits a structure far closer to the identity

as compared to a random matrix (see Table Table I). As it was previously mentioned, the design of sensing matrices with low mutual coherence is the subject of numerous works in various fields and directly carrying out such a design is a difficult task in general. Interestingly, *although the framework does not rely on explicit regularization or a tailored optimization objective to reduce the mutual coherence, the proposed methodology learns sensing matrices with a very low mutual coherence, i.e., the proposed methodology is implicitly biased towards learning high-quality task-specific sensing matrices.* This is presumably due to the model-based nature of the proposed deep architectures.

4.5 Conclusion

In this paper, we considered the problem of one-bit compressive sensing and proposed a novel hybrid *model-driven* and *data-driven* autoencoding scheme that allows us to *jointly* learn the parameters of the measurment module (i.e., the sensing matrix and the quantization thresholds) and the latent-variables of the decoder (estimator) function, based on the underlying distribution of the data. In broad terms, we proposed a novel methodology that combines the traditional compressive sensing techniques with model-based deep learning—resulting in interpretable deep architectures for the problem of one-bit compressive sensing. In addition, the proposed method can handle the recovery of the amplitude information of the signal using the learned and optimized quantization thresholds. Our simulation results demonstrated that the proposed hybrid methodology is superior to the state-of-the-art methods for the problem of one-bit CS in terms of both computional efficiency and accuracy.

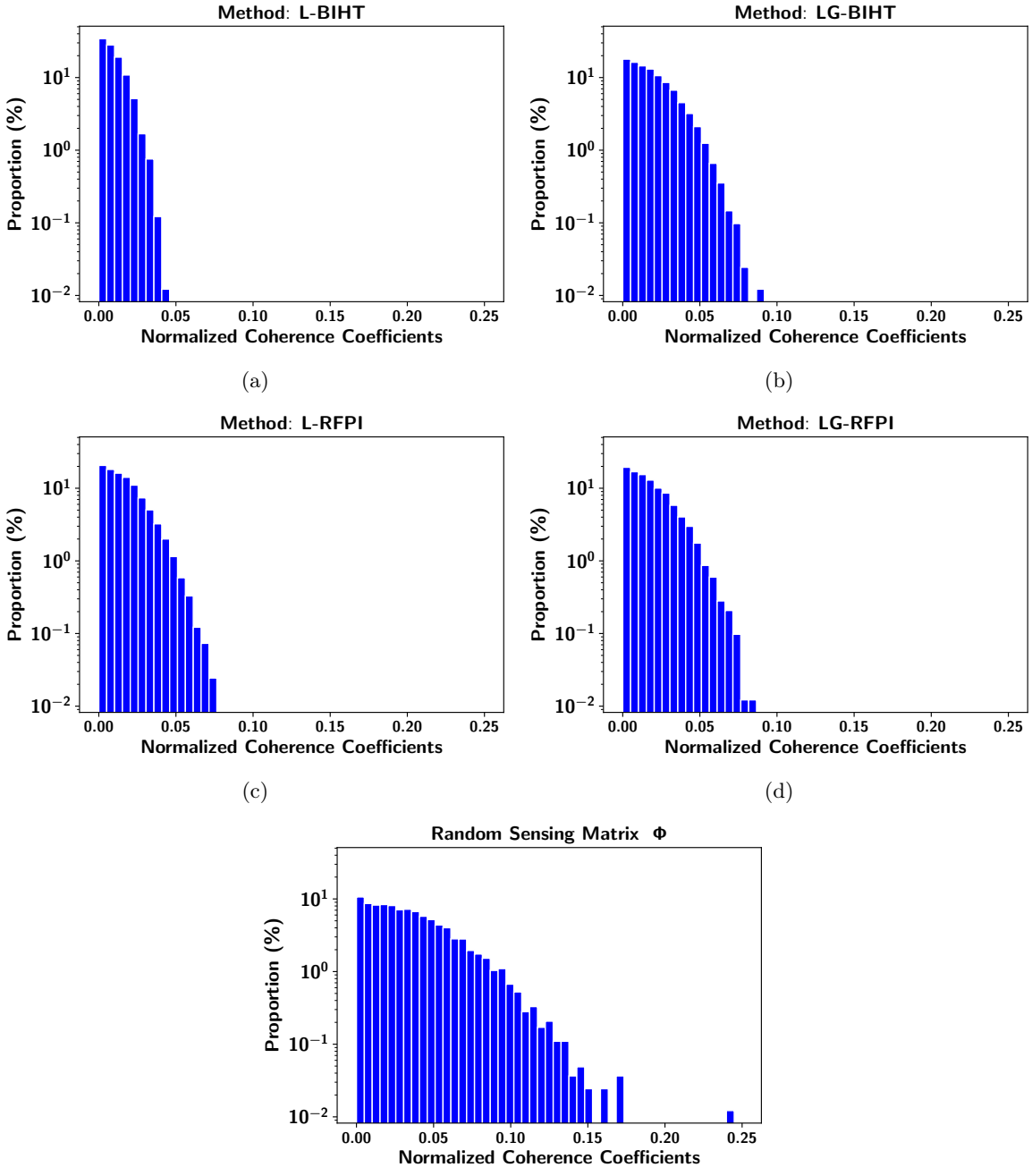


Figure 21. Distribution of the mutual coherence coefficients associated with the sensing matrix learned through (a) the proposed L-BIHT method (b) the proposed L-GBIHT method, (c) the proposed L-RFPI and (d) the proposed L-GBRFPI method, where (e) demonstrates the distribution of the mutual coherence coefficient of a random generated sensing matrix $\mathbf{\Phi}$ used in the numerical sections. It can be observed that the proposed methodologies implicitly learn sensing matrices with very low mutual coherence as compared to that of a randomly generated $\mathbf{\Phi}$.

I-C

Model-Based Deep Learning for Phase Retrieval

CHAPTER 5

UNFOLDED ALGORITHMS FOR DEEP PHASE RETRIEVAL

Overview: Exploring the idea of phase retrieval has been intriguing researchers for decades, due to its appearance in a wide range of applications. The task of a phase retrieval algorithm is typically to recover a signal from linear phase-less measurements. In this paper, we approach the problem by proposing a hybrid model-based data-driven deep architecture, referred to as *U*nfolded *P*hase *R*etrieval (*UPR*), that exhibits significant potential in improving the performance of state-of-the-art data-driven and model-based phase retrieval algorithms. The proposed method benefits from versatility and interpretability of well-established model-based algorithms, while simultaneously benefiting from the expressive power of deep neural networks. In particular, our proposed model-based deep architecture is applied to the conventional phase retrieval problem (via the incremental reshaped Wirtinger flow algorithm) and the sparse phase retrieval problem (via the sparse truncated amplitude flow algorithm), showing immense promise in both cases. Furthermore, we consider a joint design of the sensing matrix and the signal processing algorithm and utilize the deep unfolding technique in the process. Our numerical results illustrate the effectiveness of such hybrid model-based and data-driven frameworks and showcase the untapped potential of data-aided methodologies to enhance the existing phase retrieval algorithms. relative

---

Keywords: Deep learning, deep unfolding, IRWF, model-based deep learning, phase retrieval, SPARTA.

5.1 Introduction

The task of phase retrieval is concerned with recovering a complex or real-valued signal of interest, $\boldsymbol{x} \in \mathbb{R}^n/\mathbb{C}^n$, from m linear phase-less measurements of the form

$$y_i = |\langle \boldsymbol{a}_i, \boldsymbol{x} \rangle|, \text{ for } i \in \{1, 2, \cdots, m\}, \tag{5.1}$$

where the set of sensing vectors $\{\boldsymbol{a}_i \in \mathbb{R}^n/\mathbb{C}^n\}_{i=1}^m$, are assumed to be known *a priori*. The journey of solving the decades old phase retrieval problem has led to numerous algorithms and methodologies. This is no surprise given the many applications of phase retrieval, including those in areas such as crystallography, optics, and imaging [137, 138]. These extensive practical applications have made their way into deep space as well, where phase retrieval plays an important role in signal processing for space telescopes such as NASA's Hubble Space Telescope [139–141]. With the growing number of applications in various fields, the developed methodologies continue to increase in number and efficiency. Note that a large number of methods in the literature have their roots in the seminal works by Gerchberg, Saxton, and Fienup [142–146]. Their extensive body of work was integral to the implementation of phase retrieval algorithms in NASA's Hubble Space Telescope [139, 140], and is still widely referenced in modern phase retrieval research. However, the Gerchberg-Saxton algorithm's shortcomings in terms of finding the optimal solution in an efficient manner has resulted in numerous new

directions. While some convex formulations have been proposed in the literature [147–149], most of the existing phase retrieval algorithms view the problem through a non-convex lens. As a case in point, methodologies such as Wirtinger flow (WF), truncated Wirtinger flow (TWF), reshaped Wirtinger flow (RWF/IRWF), and incremental truncated Wirtinger flow (ITWF) have all shown promise in addressing the problem in an efficient and accurate manner [152–155]. WF was the first approach to display the significance of spectral initialization and updating. Since then, other approaches such as RWF/IRWF improved the performance further by implementing their own spectral initialization and updating method [152, 155].

In many practical applications, the signal of interest is naturally sparse or can be made sparse by design [156, 157]. This has resulted in approaches that reduce the number of required measurements via block-sparse phase retrieval solvers, such as the TWF [154]. In addition, more robust approaches such as sparse truncated amplitude flow (SPARTA) have been formulated and shown to further improve performance [157]. SPARTA's two-stage approach of simple power iterations for initializations, and truncated gradient calculations via thresholds, makes it a particularly interesting. Despite the enormous progress made, there remains a number of obstacles including the reduced applicability of algorithms developed based on the Gaussian noise assumption for other kinds of disturbances [158]. Particularly, developing methods that can deal with outliers is especially important. In this context, algorithms such as AltIRLS and AltGD [158] can handle the existence of impulsive noise using an ℓ_p-fitting based estimator. Such algorithms are of particular interest due to their ability to outperform versatile algorithms such as TWF. Moreover, most established phase retrieval algorithms struggle with internal parameter

optimization, such as determining the optimal step size for signal recovery while avoiding to fall into local minima when sample sizes are small. This is still the case even with the improved algorithms such as RWF [155]. In fact, computational inefficiency is still a major obstacle in applying such phase retrieval algorithms in large-scale or real-time scenarios.

Along with the convex and non-convex divide, the phase retrieval algorithms can also be categorized based on whether they are model-based or data-driven: model-based methodologies, like those discussed above, seek to design algorithms through a preliminary modeling of the problem and incorporating the reasoning that emerge from the model, whereas data-driven methodologies rely primarily on data to solidify the workings of algorithms. Data driven approaches can help with some of the shortcomings mentioned earlier by making use of the expressive power of deep neural networks and training them in a manner that the resulting network acts as an estimator of the true signal given the measurements vector $\boldsymbol{y} = |\boldsymbol{A}^H \boldsymbol{x}|$. Within the realm of phase retrieval, deep learning has been only used to design neural nets for algorithms such as hybrid-input-output (HIO) and Fienup's method [159, 160]. Such works are limited due to their inability to deal with multiple types of models, as well as the more recent phase retrieval algorithms that are more complex in nature. Additional body work has been done using convolutional neural nets, such as prDeep [161], leading to a separate class of architectures not versatile enough to improve the existing algorithms. In fact, some of Fienup's recent work for space telescopes take advantage of smart initializations obtained from convolutional neural nets for phase retrieval [162]. The said limitations directly relate to the two major pitfalls of data-driven approaches, i.e., (i) their need to a relatively large amount of data for training purposes and (ii)

their inherent lack of interpretability, even after training. Still, data-driven approaches, such as deep learning techniques, have become immensely useful in recent years for handling complex signal processing problems. As a result, it is not difficult to see why a hybrid model-based and data-driven model has the potential to improve the data acquisition model even further by effectively dealing with each approach's weaknesses. For complex systems, such a hybrid model is promising due to its ability to integrate parameterized and non-parameterized mathematical models. In addition, the employment of particular activation functions for model-based deep architectures allows them to be differentiated from conventional deep learning methodologies. Specifically, the activation functions can be designed to mimic non-linearities already present in traditional optimization algorithms.

The *deep unfolding* technique is an effective amalgam of model-based and data-driven approaches. LeCun et al. first introduced this notion in [163]; which was extended thoroughly by Hershey et al. in [75]. Deep unfolding facilitates the design of model-aware deep architectures based on well-established iterative signal processing techniques. Deep unfolded networks have repeatedly displayed great promise in the field of signal processing and can make use of the immense amounts of data, along with the domain knowledge gleaned from the underlying problem at hand [36, 67, 68, 73, 74, 78, 130, 164]. Furthermore, they have the ability to utilize the adaptability and reliability of model-based methods, while also taking advantage of the expressive power of deep neural networks. As a result, they are an ideal candidate for problems such as phase retrieval, particularly in non-convex settings where researchers struggle with bounding the complexity of signal processing algorithms while keeping them effective. This problem

particularly arises when applying iterative optimization techniques. In this context, first-order methods are widely used as iterative optimization techniques with low *per-iteration* complexity. The drawback, on the other hand, stems from their tendency to suffer from a slow speed of convergence, since they usually require many iterations to converge. It should also be noted that predicting the number of iterations required for convergence is generally a difficult task. As a result, they may not be recommended for low-latency and reliable signal processing in real-time applications. It is therefore logical to consider fixing the total number of iterations of such algorithms (say L iterations), while at the same time, seeking to optimize the (parameters of the) iterations in order to achieve the best improvement in the underlying objective function at hand. Accordingly, our goal in this paper is to improve the existing first-order methods by *meta optimizing* the SPARTA and IRWF iterations when the total computational budget is fixed. This is done by formulating the meta-optimization problem in a deep learning setting, and interpreting the resulting unrolled iterations as a neural network with L layers where each layer is designed to imitate one iteration of the original optimization method. Eventually, such a deep neural network can be trained using a small data-set and the resulting network can be used as an enhanced first-order method for solving the underlying problem at hand.

A particular concern with regards to phase retrieval is the usual assumption that the sensing matrix is known *a priori*. In the existing literature, designing the phase retrieval algorithms has received the primary attention. Designing the measurement matrix, on the other hand, is considerably unique. Random matrices have been shown to be effective in related areas such as compressive sensing, but typically need to satisfy certain criteria to guarantee recovery. A

particularly well-researched criterion that can guarantee said recovery is the Restricted Isometry Property (RIP) [165], which is known to be a sufficient condition for noise-robust sparse signal recovery [166]. Alternatively, deterministic design of the sensing matrix presents a set of advantages that can counteract the pitfalls that stem from the random selection of matrices. Although the RIP is a well-known criterion for random matrices, testing the matrices for RIP compliance may not be efficient from a computational viewpoint. Furthermore, such random matrices themselves tend to become large and unmanageable when the signal size grows large. Such issues with random matrices naturally translate to the design of measurement matrices in phase retrieval as well [152, 167, 168], but the corresponding solutions have not been looked into thoroughly. Deterministic designs allow for judicious matrix constructions that can mitigate the efficiency and storage issues that are associated with random matrix designs. In addition, they can account for the underlying distribution of the signal and system parameters, which is the main shortcoming of a typical random matrix approach. As such, an architecture that could design the sensing vectors/matrices in a deterministic setup would be extremely valuable in practical scenarios [169]. Whether it is astronomy, X-ray imaging, or any other application, designing a sensing matrix for each signal is a difficult task [170, 171].

We note that there already have been a number of research works on designing the sensing matrix, including designs based on learning methodologies [172–178]. The interest in learning techniques such as convolutional neural nets and stacked denoised autoencoders [175, 176, 178] has increased greatly. This has led to works such as [179], where the measurement matrix is designed based on mutual information, but the ideas are not exploited for recovery. A deep

architecture that could handle the design of the sensing matrix along with the recovery task would be able improve the accuracy and efficiency of phase retrieval in a way unmatched by previous methods. As result, we undertake a deterministic task-specific and data-specific design of the sensing matrix that can be cascaded to the recovery algorithm, resulting in an immense improvement in the recovery performance.

Contributions Overview: In this paper, we propose model-aware deep architectures, referred to as Unfolded Phase Retrieval (UPR)—as a new approach to the problem of phase retrieval. In particular, we focus on two scenarios: the conventional phase retrieval problem (with IRWF) and the sparse phase retrieval problem (with SPARTA), which have both shown great promise. Furthermore, we propose a general framework for designing task-specific sensing matrices for improving the performance of the underlying recovery algorithm and to outperform numerous state-of-the-art algorithms as a supplementary feature. Deep unfolding has the unique distinction of falling into the category of a hybrid model-based and data-driven methodology. More explicitly, we consider a joint design of the sensing matrix and the signal processing algorithm and utilize the deep unfolding technique in the process. As a result, our hybrid methodology can adopt the advantages of both methodologies as well. Specifically, we have the capability to exploit the data for better accuracy and performance, the resulting interpretability leads to trusted outcomes, we require less data due to having less parameters to train, and obtain an enhanced convergence rate. UPR allows for a unification of the optimization process for system parameters in an end-to-end manner. Interestingly, UPR acts as a unified framework for both optimal sensing and recovery. As mentioned before, such a framework can be cascaded to

the recovery algorithm which allows for noticeable recovery accuracy improvement. Additionally, UPR has a remarkable compatibility with gradient based algorithms, which is displayed both in our description of the methodology and in our implementation with SPARTA and IRWF. Finally, we compare our results with the said baseline algorithms and different variants of the RWF algorithm that have already shown good performance in the context of phase retrieval. We show that UPR significantly outperforms the baseline algorithms from which it has emerged.

Organization of the Paper: Section II is devoted to a technical overview of the system model and a brief overview of deep unfolding in the context of phase retrieval. Section III introduces the UPR framework, its baseline algorithms and its initialization method, as well as our methodology for the design of the sensing matrix. We examine the performance of UPR in Section IV. Finally, Section V concludes the paper.

Notations: Throughout the paper, we use boldface lowercase letters to denote vectors, and boldface capital letters to denote matrices. Calligraphic letters are reserved for sets. The superscripts $(\cdot)^*$, $(\cdot)^T$, and $(\cdot)^H$ represent the conjugate, the transpose, and the Hermitian operators, respectively. $\|\cdot\|_2$ represents the Euclidean norm of the vector argument. The operator symbol $\odot$ represents the Hadamard product of matrices.

5.2 System Model and Problem Formulation

In this section, we present a mathematical formulation of the phase retrieval problem and provide a brief overview of the existing classical model-based phase retrieval algorithms which will lay the ground for the proposed hybrid model-based and data-driven methodology.

As discussed earlier, a phase retrieval *system* can be mathematically formalized by considering the following encoding module (i.e., the data-acquisition system):

$$\text{Encoding Module:} \quad \boldsymbol{y} = |\boldsymbol{A}\boldsymbol{x}|\,, \tag{5.2}$$

where $\boldsymbol{x} \in \mathbb{C}^n$ denotes the underlying signal of interest, $\boldsymbol{A} = [\boldsymbol{a}_1, \boldsymbol{a}_2, \cdots, \boldsymbol{a}_m] \in \mathbb{C}^{n\times m}$ denotes the sensing matrix, and $\boldsymbol{y} \in \mathbb{R}^m$ represents the captured phase-less measurements of the signal $\boldsymbol{x}$.

Traditionally, in a phase retrieval problem we seek to retrieve the measurement phase by recovering the signal of interest $\boldsymbol{x}$ from the embedded phase-less measurements $\boldsymbol{y}$ given the knowledge of the sensing matrix $\boldsymbol{A}$. Specifically, the phase retrieval problem under the noise-free assumption can be formally stated as:

$$\mathcal{P}_0: \quad \text{find} \quad \boldsymbol{x} \quad \text{s.t.} \quad \boldsymbol{y} = |\boldsymbol{A}\boldsymbol{x}|\,, \tag{5.3}$$

where the constraint in $\mathcal{P}_0$ represents the feasible set of the problem corresponding to the data consistency principle. Clearly, the above program is non-convex due to its non-convex constraint. Moreover, observe that if $\boldsymbol{x}^\dagger$ is a feasible point of the problem $\mathcal{P}_0$, then $e^{-j\phi}\boldsymbol{x}^\dagger$ is also a solution to the problem for an arbitrary phase constant ϕ. Hence, it is only possible to recover the underlying signal of interest up to a phase shift factor in a phase retrieval problem. With that in mind, let $\boldsymbol{x}^\dagger$ be any solution to the phase retrieval problem defined in (Equation 5.3) and

let $\boldsymbol{x}^\star$ represent the true signal of interest. A meaningful quantifying metric of closeness of the recovered signal $\boldsymbol{x}^\star$ to the true signal can then be defined as follows:

$$\mathrm{D}(\boldsymbol{x}^\star, \boldsymbol{x}^\dagger) = \min_{\phi \in [0, 2\pi)} \|\boldsymbol{x}^\star - e^{-j\phi}\boldsymbol{x}^\dagger\|_2^2, \tag{5.4}$$

which was first proposed in [152] and can be interpreted as measuring the Eucledian distance of two complex vectors up to a global phase difference.

In this work, we consider the phase retrieval problem from the perspectives of *system design* and *reconstructive algorithm development*. A system design approach for a phase retrieval problem is mainly concerned with finding the best set of parameters of the encoder module that facilitate the recovery of the underlying signal of interest from the phase-less measurements captured through the encoder module (i.e., the data acquisition system). In particular, a system design perspective for a phase retrieval data acquisition system can be defined as designing the sensing matrix in a deterministic fashion according to a performance criterion. As an example, [179] has considered the design of the sensing matrix in a deterministic manner such that it maximizes the mutual information between the signal of interest and the acquired phase-less measurements. However, the development of effective signal reconstruction techniques that can harness this maximal mutual information obtained by a judicious design of the sensing matrix is still an open problem. Hence, a more natural approach to this problem is to consider a joint design of the sensing matrix and the recovery algorithm.

We seek to jointly design the sensing matrix (encoder module) and the reconstruction algorithm (decoder module), in contrast to the existing methodologies that either consider the development of the reconstruction algorithm or the design of the sensing matrix. The proposed methodology can be viewed as a unification of both approaches. Specifically, we propose a hybrid model-based and data-driven methodology to this problem and utilize the notions of auto-encoders and the deep unfolding technique as the main tools to develop our proposed framework. By implementing such a methodology, while the sensing matrix will be designed to maximize the signal reconstruction accuracy at the decoder module, the decoder module itself will be trained to perform the recovery by accounting for the learned sensing matrix—leading to the promised joint design.

In order to make the joint design possible, we seek to identify sensing matrices that make the recovery of the underlying signal of interest feasible and to identify structures that are amenable to a specific class of reconstruction algorithms and vice versa. Let $\mathcal{D}_{\Phi} : \mathbb{R}^n \mapsto \mathbb{C}^m$ denote the class of decoder functions parameterized on a set of parameters Φ. Further assume that for a given measurement matrix $\boldsymbol{A}$ and the corresponding measurements vector $\boldsymbol{y}$, $\hat{\boldsymbol{x}}$ represents an estimation of the signal of interest provided by a certain characterization of the decoder function, viz.

$$\hat{\boldsymbol{x}} = \mathcal{D}_{\phi}\left(\boldsymbol{y}; \boldsymbol{A}\right), \tag{5.5}$$

where $\phi \in \Phi$ denotes a characterization of the decoder module from the original class. Recall that for a fixed sensing matrix $\boldsymbol{A}$, the dynamics of the data-acquisition model of a phase retrieval system is given by $\boldsymbol{y} = |\boldsymbol{A}\boldsymbol{x}|$. Accordingly, for a fixed characterization of the decoder function $\mathcal{D}_{\phi\in\Phi}$, the design of the sensing matrix $\boldsymbol{A} \in \mathbb{C}^{m\times n}$ can be considered according to the following optimization problem:

$$\min_{\boldsymbol{A}\in\mathbb{C}^{m\times n}} \ \mathrm{E}_{\boldsymbol{x}\sim P(\boldsymbol{x})} \left\{ \mathrm{D}\left(\boldsymbol{x}, \mathcal{D}_{\phi}(|\boldsymbol{A}\boldsymbol{x}|; \boldsymbol{A})\right)\right\}, \tag{5.6}$$

where $P(\boldsymbol{x})$ represents the distribution of the underlying signal of interest. For simplicity, we drop the notation $\boldsymbol{x} \sim P(\boldsymbol{x})$, and unless mentioned otherwise, all expectations are taken with respect to the input distribution in the sequel. The aforementioned approach for the design of the sensing matrix considers the signal reconstruction accuracy as a design criterion with respect to a particular realization of the decoder functions. Thus, the obtained matrix $\boldsymbol{A}$ can be viewed as a task-specific encoding matrix which not only considers the underlying distribution of the signal, but also the considered class of decoder functions resulting in a superior performance. Once a solution to (Equation 5.6) is obtained, the sensing matrix can be utilized for data-acquisition purposes while the fixed decoder module $\mathcal{D}_{\phi}$ can be used to carry out the reconstruction of the signal of interest. However, starting our design from (Equation 5.6) presents us with an inherent performance bottleneck. To see why, let $\mathcal{D}_{\phi_1\in\Phi}$ and $\mathcal{D}_{\phi_2\in\Phi}$ represent two realizations of the decoder module from the same class, and let $\boldsymbol{A}$ denote a fixed sensing matrix that will

be the solution to (Equation 5.6) for $\phi = \phi_2$. Then, for a signal of interest $\boldsymbol{x} \sim P(\boldsymbol{x})$ and the corresponding phase-less measurements $\boldsymbol{y}$, it might be the case that

$$\mathbb{E}\left\{\mathrm{D}(\boldsymbol{x}, \mathcal{D}_{\phi_1}(\boldsymbol{y}; \boldsymbol{A}))\right\} \leq \mathbb{E}\left\{\mathrm{D}(\boldsymbol{x}, \mathcal{D}_{\phi_2}(\boldsymbol{y}; \boldsymbol{A}))\right\}. \tag{5.7}$$

In other words, the realization of the decoder module on the set of parameters $\phi_1 \in \Phi$, results in a more accurate reconstruction algorithm as compared to the alternative realization. As a result, it is better to consider the design of the sensing matrix with respect to $\mathcal{D}_{\phi_1}$ in such a case. Nonetheless, finding an optimal $\phi^\star \in \Phi$ such that

$$\mathbb{E}\left\{\mathrm{D}(\boldsymbol{x}, \mathcal{D}_{\phi^\star}(\boldsymbol{y}; \boldsymbol{A}))\right\} \leq \mathbb{E}\left\{\mathrm{D}(\boldsymbol{x}, \mathcal{D}_{\phi}(\boldsymbol{y}; \boldsymbol{A}))\right\}, \quad \forall \phi \in \Phi,$$

is a particularly challenging task due to the non-convex nature of the problem. If such a characterization $\phi^\star$ is known, the design of the sensing matrix matrix can easily be carried out. The above observation is further evidence for the paramount importance of developing a unified framework that allows for a joint optimization of the reconstruction algorithm (i.e. finding an optimal or sub-optimal $\phi^\star$) and the sensing matrix. In the following, we consider the joint design of the sensing matrix and the signal processing algorithm and propose a novel framework based on the deep unfolding technique that allows for an optimization over the set of both design parameters $\{\Phi, \boldsymbol{A}\}$. The resulting framework can be viewed as a hybrid data-driven model-based deep neural network that achieves a far superior performance when compared to the existing model-based and data-driven methodologies. Furthermore, the obtained deep architecture is

interpretable due to the incorporation of the domain knowledge and assumes far less parameters when compared to existing black-box neural networks. Due to the fewer training parameters, the proposed architecture requires far fewer training samples for optimization of the network when compared to the existing data-driven methodologies. We will provide a high-level description of the proposed methodology and proceed deeper into the details of each module in the next section.

We conclude this section by mathematically formalizing the problem at hand. We cast the problem of jointly designing the sensing matrix and the reconstruction algorithm as the following optimization problem:

$$\min_{\{\boldsymbol{A}\in\mathbb{C}^{m\times n},\phi\in\boldsymbol{\Phi}\}} \mathbb{E}\{\mathrm{D}\left(\boldsymbol{x},\mathcal{D}_{\phi}\left(|\boldsymbol{A}\boldsymbol{x}|;\boldsymbol{A}\right)\right)\}. \tag{5.8}$$

Our preliminary step is to mathematically formulate a proper parameterized class of the decoder and encoder modules that facilitate the incorporation of domain knowledge. To this end, we consider the realization of the decoder module by first formulating the problem of phase retrieval as an optimization problem, and then, we resort to first-order optimization techniques that lay the groundwork for obtaining a rich class of parameterized decoder functions. Once such a class is formalized, we make the connection between deep neural networks and first-order optimization techniques and propose two novel model-based deep architectures that encode the domain knowledge in their respective architecture and task-specific computations. Next, we

show how to tackle the above program using the prominent deep learning techniques by taking advantage of the available data.

5.3 UPR: The Proposed Framework

In this section, we present the proposed hybrid data-driven and model-based ***U***nfolded ***P***hase ***R***etrieval (UPR) framework. We begin our presentation by mathematically defining the encoder and decoder modules which take the form of a model-based deep architecture, highly tailored to the problem of phase retrieval. Then, we interpret the overall dynamics of the proposed methodology as a single auto-encoder module whose training procedure takes the form of (Equation 5.8), which can then be tackled using the well-established deep learning techniques such as back-propagation. Once the training of the auto-encoder module is complete, the optimized sensing matrix can be extracted from the encoder module for data-acquisition purposes, while the optimized decoder module can be used as an enhanced signal reconstruction algorithm to carry out the signal estimation task from the phase-less measurements obtained through the optimized sensing matrix.

5.3.1 Architecture of the Encoder Module

We take this opportunity to formally define the class of encoder functions for the phase retrieval problem. Recall that the governing dynamics of the data-acquisition system in a phase retrieval problem is given by

$$\boldsymbol{y} = |\boldsymbol{A}\boldsymbol{x}|, \tag{5.9}$$

where the measurement vector $\boldsymbol{y}$ and the sensing matrix $\boldsymbol{A}$ are used as the input to a signal reconstruction algorithm to obtain $\hat{\boldsymbol{x}}$. Now, we use the above dynamics as a blue-print to design our encoder module parametrized on the sensing matrix. To this end, we define the hypothesis class $\mathcal{H}_e$ of the possible encoder functions (module) as follows:

$$\mathcal{H}_e = \{f_{\boldsymbol{A}} : \boldsymbol{x} \mapsto |\boldsymbol{A}\boldsymbol{x}| \ : \boldsymbol{A} \in \mathbb{C}^{m \times n}\}, \tag{5.10}$$

where w $f_{\boldsymbol{A}}(\boldsymbol{x}) \triangleq |\boldsymbol{A}\boldsymbol{x}|$. The above hypothesis class encapsulates the possible encoder modules whose computation dynamics mimics the behaviour of the data-acquisition model in a phase retrieval model. Alternatively, the above hypothesis class can be interpreted as a *one-layer neural network* with n input neurons and m output neurons, where the activation function is given by $|\cdot|$ with the trainable weight matrix $\boldsymbol{A}$. Specifically, the input to such a neural network is $\boldsymbol{x} \in \mathbb{C}^n$, while the output is $|\boldsymbol{A}\boldsymbol{x}| \in \mathbb{R}^m$. This paves the way for viewing the design of the sensing matrix from a statistical learning theory perspective. For instance, the problem of designing the sensing matrix for a given realization of a decoder function $\mathcal{D}_\phi$ with respect to the hypothesis class $\mathcal{H}_e$ can be re-expressed as the following *learning problem*:

$$\min_{\mathcal{E}_{\boldsymbol{A}} \in \mathcal{H}_e} \ \mathbb{E}\{\mathrm{D}\left(\boldsymbol{x}, \mathcal{D}_\phi \circ \mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x})\right)\}. \tag{5.11}$$

The above learning problem seeks to find an encoder function $\mathcal{E}_{\boldsymbol{A}}$ such that the resulting functions maximizes the reconstruction accuracy according to the decoder function $\mathcal{D}_\phi$. Equivalently, (Equation 5.11) can be expressed as learning the weights of a one-layer neural net-

work $\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}) = |\boldsymbol{A}\boldsymbol{x}|$, where for any choice of $\boldsymbol{A}$ we have $\mathcal{E}_{\boldsymbol{A}} \in \mathcal{H}_e$. Then, the problem in (Equation 5.11) can be re-expressed as a deep learning problem by considering the training of $\mathcal{E}_{\boldsymbol{A}}$ according to a loss function that is the same as objective function in (Equation 5.11), i.e.,

$$\min_{\boldsymbol{A}\in\mathbb{C}^{m\times n}} \; \mathbb{E}\left\{\mathrm{D}(\boldsymbol{x}, \mathcal{D}_{\phi} \circ \mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}))\right\}. \tag{5.12}$$

Hereafter, we seek to formulate the joint design of the sensing matrix and the reconstruction algorithm (i.e., see (Equation 5.8)) from a deep learning and statistical learning point-of-view, and eventually, formulate (Equation 5.8) as a purely deep learning problem and tackle it by utilizing the well-established deep learning techniques. The following subsection, in particular, is devoted to mathematically defining two parameterized classes of decoder functions; one specialized for sparse phase retrieval, and the other, specialized for the conventional class of phase retrieval problems. Moreover, we will formally introduce the UPR framework and its training procedure.

5.3.2 Architecture of the Decoding Module

In this subsection, we propose two model-based deep architectures that are not only highly-tailored to the problem of phase retrieval but also allow for learning task-specific superior reconstruction algorithms. Next, we cascade the decoder and the encoder module to obtain an auto-encoder deep architecture that will facilitate the joint training of the sensing matrix and the reconstruction algorithm.

Deep Unfolded Networks for Phase Retrieval. We begin with a brief introduction to deep unfolded networks and their relation with mathematical optimization. Note that the goal of a decoder module is to obtain an estimate of the underlying signal of interest from the phase-less measurements of the form $\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}) = |\boldsymbol{A}\boldsymbol{x}|$, for a given sensing matrix $\boldsymbol{A}$. Equivalently, a decoder function seeks to tackle the following problem:

$$\text{find} \quad \boldsymbol{z} \quad \text{s.t.} \quad \mathcal{E}_{\boldsymbol{A}}(\boldsymbol{z}) = \mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}). \tag{5.13}$$

In lieu of directly tackling (Equation 5.13), a more practical approach is a reformulation of (Equation 5.13) as an optimization problem to approximate of the true signal of interest. Accordingly, (Equation 5.13) can be reformulated as:

$$\min_{\boldsymbol{z}\in\mathbb{C}^n} \quad \ell(\boldsymbol{z}) \quad \text{s.t.} \quad \boldsymbol{z} \in \Omega, \tag{5.14}$$

where Ω represents the search space of the underlying signal of interest. For instance, if it is known *a priori* that the signal of interest is s-sparse, this information can be encoded in Ω, i.e., $\Omega = \{\boldsymbol{x} \in \mathbb{C}^n : \|\boldsymbol{x}\|_0 = s\}$. On the other hand, for a conventional phase retrieval problem where no prior structural information is assumed on $\boldsymbol{x}$, we set $\Omega = \mathbb{C}^m$.

The ultimate formalization of a phase retrieval problem, as we move from (Equation 5.13) to (Equation 5.14), boils down to a proper realization of the loss function $\ell(\boldsymbol{x})$ and the feasible set Ω. In particular, for a given signal of interest $\boldsymbol{x}$ and sensing matrix $\boldsymbol{A}$, if $\boldsymbol{z}^\star \in \operatorname{argmin}_{\boldsymbol{z}\in\Omega} \ell(\boldsymbol{z})$ then we must have $\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{z}^\star) = \mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x})$. Note that different choices of the loss function $\ell(\boldsymbol{x})$ and

the feasible set Ω give rise to various flavors of recovery algorithms for a phase retrieval, as has been previously discussed. In this work, we consider the development of the class of decoder functions $\mathcal{H}_d$ such that for any realization of the decoder function $D_\phi \in \mathcal{H}_d$, the function $\mathcal{D}_\phi$ becomes a good approximator of the solution to (Equation 5.14). Note that there exist alternative approaches to tackling the phase retrieval problem. For example, the direct development of a class of decoder functions can be considered which are model-agnostic. One particular class of such model-agnostic methodologies can be derived by utilizing black-box deep neural networks. Assume that the set $\{\boldsymbol{x}_i, \mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}_i)\}_{i=0}^{B-1}$ represents our available data from the phase retrieval system, i.e., we have B realizations of the input-output relationship of our phase retrieval system for a given $\boldsymbol{A}$. Furthermore, let

$$\mathcal{F}_{\sigma,L} = \{f_\Gamma : \mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}) \mapsto \boldsymbol{x} \,|\, f_\Gamma(\boldsymbol{x}) = \sigma_{\gamma_L}^L \circ \cdots \circ \sigma_{\gamma_1}^1(\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}))\} \tag{5.15}$$

represent the class of all L-layer deep neural networks with activation function $\sigma(\cdot)$, where γ_i denotes the weights of the i-th layer, and $\Gamma = \{\gamma_i\}_{i=1}^L$. Then, a decoder function $\hat{f}_{\Gamma^\star} \in \mathcal{F}_{\sigma,L}$ can be found by training the neural network; e.g., the realization of a decoder function with respect to some fixed sensing matrix $\boldsymbol{A}$ is given by:

$$\hat{f}_{\Gamma^\star} = \underset{f_\Gamma \in \mathcal{F}_{\sigma,L}}{\operatorname{argmin}} \frac{1}{B} \sum_{i=0}^{B-1} \|\boldsymbol{x}_i - f_\Gamma(\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}_i))\|_2^2. \tag{5.16}$$

The above model-agnostic decoder function is only justifiable in scenarios where the input-output relationship of the system is unknown (i.e. in applications such as computer vision and

natural language processing problems). Furthermore, due to the model-agnostic nature of the above methodology, the obtained decoder function is not interpretable and acts as a black-box decoder in which the decision making process of the decoder function cannot be examined by the user. In addition, the above black-box methodology for solving the inverse problem results in a very large number of training parameters Γ, which begs for a very large number of training samples B to optimize the network. We note that there exists a rich literature on phase retrieval, and hence, choosing such a black-box approach for this problem would not be appropriate. Most prudent would be to develop a more mature deep learning model for this problem which incorporates of domain knowledge, while still harnessing the expressive power of deep neural network. In the following, we show how a model-aware deep architecture can be obtained for phase retrieval. In particular, we employ the deep unfolding approach to obtaining a class of decoder modules based on an optimization problem of the form (Equation 5.14).

Generally speaking, finding a closed-form solution to (Equation 5.14) for a given loss function is either a very difficult task or even mathematically intractable. Thus, after formalizing the objective function and the search space for a phase retrieval problem, first-order mathematical optimization techniques are utilized to tackle (Equation 5.14). Iterative optimization techniques are a popular choice for both convex and non-convex programming. Specifically, first-order methods are among the most popular and well-established iterative optimization techniques due to their low per-iteration complexity and efficiency in complex scenarios. One of the most prominent first-order optimization techniques suitable for our problem in (Equation 5.14) is the projected gradient descent algorithm (PGD) [180–185]. In particular, assuming $\boldsymbol{z}^0$ is the initial

point for the algorithm, the l-th iteration of PGD to approximate the solution to (Equation 5.14) can be defined as follows:

$$\boldsymbol{z}^{l+1} = \mathcal{P}_{\Omega}\left(\boldsymbol{z}^{l} - \alpha^{l}\nabla_{\boldsymbol{z}}\ell(\boldsymbol{z}^{l})\right), \tag{5.17}$$

where $\mathcal{P}_{\Omega} : \mathbb{C}^n \mapsto \Omega \subseteq \mathbb{C}^m$ denotes a mapping function of the vector argument to the feasible set Ω, $\nabla_{\boldsymbol{z}}\ell(\boldsymbol{z}^l)$ denotes the gradient of the objective function obtained at the point $\boldsymbol{z}^l$, and α^l represents the step-size of the PGD algorithm at the l-th iteration. We note that first-order methods generally suffer from a slow speed of convergence and predicting the number of iterations they require for convergence is a difficult task. As a result, they are not ordinarily recommended for real-time signal processing applications. A sensible approach to circumvent this issue is to fix the total number of iterations of such algorithms (e.g., L), and to follow up with a proper choice of the parameters for each iteration (i.e., the step-sizes) that results in the best improvement in the objective function, while allowing only L iterations.

Let $\boldsymbol{g}_{\boldsymbol{\gamma}_l} : \mathbb{C}^n \mapsto \mathbb{C}^n$, be a parameterized mapping operator defined as

$$\boldsymbol{g}_{\boldsymbol{\gamma}_1}(\boldsymbol{z}) = \mathcal{P}_{\Omega}\left(\boldsymbol{z} - \boldsymbol{G}^{l}\nabla_{\boldsymbol{z}}\ell(\boldsymbol{z})\right), \tag{5.18}$$

where $\boldsymbol{\gamma}_l = \{\boldsymbol{G}^l\}$ denotes the set of parameters of the mapping function $\boldsymbol{g}_{\boldsymbol{\gamma}l}$, and $\boldsymbol{G}^l$ denotes a positive semi-definite matrix. The above mapping can then be utilized to model various first-order optimization techniques for the problem at hand, and in particular, the considered PGD

algorithm in this work. Consequently, performing L iterations of the form (Equation 5.17) can be modeled as follows:

$$\mathcal{G}_{\boldsymbol{\Gamma}}^{L}(\boldsymbol{x}) = \boldsymbol{g}_{\boldsymbol{\gamma}_L} \circ \boldsymbol{g}_{\boldsymbol{\gamma}_{L-1}} \circ \cdots \circ \boldsymbol{g}_{\boldsymbol{\gamma}_1}(\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}); \boldsymbol{x}_0), \tag{5.19}$$

where the above function corresponds to the PGD in all scenarios, be it when we have a fixed step-size $\boldsymbol{G}^l = \alpha \boldsymbol{I}$, time-varying step-sizes $\boldsymbol{G}^l = \alpha^l \boldsymbol{I}$, or the general preconditioned PGD algorithm with an arbitrary choice of $\boldsymbol{G}^l \succeq 0$. Note that the rate of convergence for the mapping function $\mathcal{G}_{\boldsymbol{\Gamma}}^{L}$ depends heavily on the choice of the parameter set $\boldsymbol{\Gamma}$, i.e., the preconditioning matrices. These parameters are usually heuristically selected in the literature and there is no straightforward methodology to obtain the best set of parameters $\boldsymbol{\Gamma}$ such that $\mathcal{G}_{\boldsymbol{\Gamma}}$ results in the best improvement in the objective function for applying exactly L iterations. Furthermore, under some mild conditions and for a proper choice of the step-size α, it can be shown that as $L \to \infty$, the output of the mapping function $\mathcal{G}_{\boldsymbol{\Gamma}=\alpha}^{L}$ converges to a first-order optimal point of the objective function.

In light of the above, we now formally give a high-level definition of the considered hypothesis classes of the decoder function in accordance with (Equation 5.19). Note that for a fixed realization of the set of parameters $\boldsymbol{\Gamma}$ and L, the realized mapping function $\mathcal{G}_{\boldsymbol{\Gamma}}^{L}$ can be viewed as a problem-specific decoder function. In particular, after the formalization of the objective function $\ell(\boldsymbol{x})$ for a phase retrieval problem, and with an initial point of $\boldsymbol{x}^0$, the output of the

mapping function $\hat{\boldsymbol{x}} = \mathcal{G}_{\boldsymbol{\Gamma}}^{L}(\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}); \boldsymbol{x}^0)$ can be considered to be an estimate of the signal of interest. Accordingly, we define the hypothesis class of possible decoder functions as follows:

$$\mathcal{H}_{d,L} = \{\mathcal{D}_{\boldsymbol{\Gamma}} : \mathbb{C}^n \mapsto \mathbb{C}^n | \boldsymbol{\Gamma} = \{\boldsymbol{\gamma}_i = \boldsymbol{G}^i \in \mathbb{S}_+\}_{i=1}^{L}\}, \tag{5.20}$$

where $\boldsymbol{\Gamma} = \{\boldsymbol{\gamma}_i = \boldsymbol{G}^i \in \mathbb{S}_+\}_{i=1}^{L}$ denotes the set of parameters of the proposed class of decoder functions, $\mathbb{S}_+$ denotes the set of all positive semi-definite matrices, and $\mathcal{D}_{\boldsymbol{\Gamma}}(\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}); \boldsymbol{x}_0) = \boldsymbol{g}_{\boldsymbol{\gamma}_L} \circ \boldsymbol{g}_{\boldsymbol{\gamma}_{L-1}} \circ \cdots \circ \boldsymbol{g}_{\boldsymbol{\gamma}_1}(\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}); \boldsymbol{x}_0)$. Observe that for a fixed realization $\mathcal{G}_{\boldsymbol{\Gamma}^\star} \in \mathcal{H}_{d,L}$, one can interpret $\mathcal{G}_{\boldsymbol{\Gamma}^\star}$ as an L-layer deep neural network, whose computation dynamics at each layer mimics the behaviour of one iteration of a first-order optimization algorithm as given in (Equation 5.17). Specifically, due to the incorporation of domain knowledge in the design of the deep unfolded network, the resulting deep neural network $\mathcal{G}_{\boldsymbol{\Gamma}}$ is interpretable and has far fewer training parameters as compared to its black-box counterparts. Moreover, the hypothesis class $\mathcal{H}_{d,L}$ encompasses the class of gradient method realizations with a varying preconditioning matrix. We seek to find the best set of parameters of a first-order optimization algorithm with only L-iterations, or equivalently, to find $\mathcal{G}_{\Gamma} \in \mathcal{H}_{d,L}$ such that $\|\boldsymbol{x} - \mathcal{G}_{\Gamma} \circ \mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x})\|$ is minimized. In the case of the phase retrieval problem and an Euclidean metric $\mathrm{D}(\cdot, \cdot)$, we are concerned with the following optimization problem:

$$\min_{\mathcal{G}_{\Gamma} \in \mathcal{H}_{d,L}} \mathbb{E}_{\boldsymbol{x} \sim P(\boldsymbol{x})}\{\mathrm{D}(\boldsymbol{x}, \mathcal{G}_{\boldsymbol{\Gamma}} \circ \mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}))\}. \tag{5.21}$$

The above program can be re-written in the more familiar form of optimizing the set of parameters Γ containing the weights of each layer of the considered deep neural network defined in (Equation 5.19). Note that our goal is to perform a joint design over the set of parameters of both the decoder and encoder module which we address in the next part. However, for a fixed $\boldsymbol{A}$, the learning of the deep decoder module with respect to a realization of the encoder function can be expressed as the following reformulation of (Equation 5.21):

$$\min_{\boldsymbol{\Gamma}=\{\boldsymbol{G}^i\in\mathbb{S}_+\}_i} \mathbb{E}_{\boldsymbol{x}\sim P(\boldsymbol{x})}\{\mathrm{D}(\boldsymbol{x}, \mathcal{G}_{\boldsymbol{\Gamma}}\circ\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}))\}, \tag{5.22}$$

where the above optimization problem formulation corresponds to training an L-layer deep neural network $\mathcal{G}_{\boldsymbol{\Gamma}}$ with a task-specific architecture as defined in (Equation 5.19), and the input to such a model-based deep architecture is given by a realization of the encoder module. In the following, we build upon the aforementioned ideas and propose two model-based deep architectures as decoder modules for both the conventional phase retrieval problem and a sparse phase retrieval settings. Specifically, we consider the unfolding of iteration of two well-established signal processing techniques for the phase retrieval problem to derive the considered decoder modules.

- ***UPR-SPARTA: Sparse Phase Retrieval***. We now present the structure of the proposed UPR-SPARTA deep decoder module for the problem of sparse phase retrieval. As mentioned earlier, one particular area of interest in phase retrieval is when the signal of interest is sparse in nature. The applications of sparse phase retrieval emerge across different fields

of engineering, and are seen especially in domains such as imaging [138]. Several algorithms and frameworks readily exist for sparse phase retrieval. The most notable one is the sparse truncated amplitude flow (SPARTA) methodology which seeks to recover a s-sparse signal from its phase-less measurements [157]. Inspired by SPARTA, the following discussion is aimed at providing the iterations of a first-order gradient-based optimizer that lays the groundwork for the proposed UPR-SPARTA model-based decoder module.

We begin our work by formalizing the problem in a non-convex form similar to (Equation 5.14). Let $\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x})$ represent the encoder module providing the phase-less measurements of an underlying s-sparse signal $\boldsymbol{x}$, i.e., $\boldsymbol{x} \in \Omega := \{\boldsymbol{x} \in \mathbb{C}^n | \|\boldsymbol{x}\|_0 = s\}$. Then, decoding the signal $\boldsymbol{x}$ from the measurement vector $\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x})$ can be formulated as the following non-convex optimization problem:

$$\min_{\boldsymbol{z}} \ \ell(\boldsymbol{z}; \mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x})) \equiv \frac{1}{m}\|\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}) - \mathcal{E}_{\boldsymbol{A}}(\boldsymbol{z})\|_2^2, \ \text{ s.t. } \|\boldsymbol{z}\|_0 = s. \tag{5.23}$$

Note that not only both the objective function and the constraint in the above are non-convex, the problem is deemed to be NP-hard in its general form. The SPARTA algorithm [157] was proposed to efficiently approximate the solution to (Equation 5.23). In particular, the SPARTA solver works through a two-stage process. The first stage is concerned with a sparse orthogonality-promoting initialization and employs iterations based on truncated gradients. Specifically, the initialization utilizes power iterations on the estimated support of the sparse signal to solve a PCA-type problem. The second stage then makes use of truncated gradients for thresholding iterations, except for s signal elements with the largest magnitudes.

Starting from an initial guess $\boldsymbol{z}_0$, the SPARTA algorithm is tasked with approximating the solution to (Equation 5.23) in an iterative manner, via update equations as follows:

$$\boldsymbol{z}^{l+1} = \mathcal{H}_s\left(\boldsymbol{z}^l - \alpha \nabla_{\boldsymbol{z}} \ell_{\text{tr}}(\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}), \boldsymbol{z}^l)\right), \tag{5.24}$$

where the above iterations can be viewed as performing the gradient descent algorithm with a per-iteration step-size α, on the truncated loss function objective ℓ_{tr} whose gradient is defined as:

$$\nabla_{\boldsymbol{z}} \ell_{\text{tr}}(\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}), \boldsymbol{z}) \triangleq \sum_{i \in \mathcal{I}_{l+1}} \left(\boldsymbol{a}_i^T \boldsymbol{z} - [\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x})]_i \frac{\boldsymbol{a}_i^T \boldsymbol{z}}{|\boldsymbol{a}_i^T \boldsymbol{z}|}\right) \boldsymbol{a}_i, \tag{5.25}$$

and where $\mathcal{I}_{l+1} = \{1 \leq i \leq m | |\boldsymbol{a}_i^T \boldsymbol{z}^l| \geq [\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x})]_i/(1+\tau)\}$, τ represents the truncation threshold, $\mathcal{H}_s(\boldsymbol{z}) : \mathbb{R}^m \mapsto \mathbb{R}^m$ sets all entries of $\boldsymbol{u}$ to zero except for the s entries with the largest magnitudes. In particular, $\boldsymbol{z}^0$ will be initialized as $\sqrt{\sum_{i=1}^m [\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x})]_i^2/m}\hat{\boldsymbol{z}}^0$ where $\hat{\boldsymbol{z}}^0 \in \mathbb{R}^n$ is created by placing zeros in $\hat{\boldsymbol{z}}^0_{\hat{\mathcal{S}}}$ where the indices are not in $\hat{\mathcal{S}}$. The principal eigenvector $\hat{\boldsymbol{z}}^0_{\hat{\mathcal{S}}} \in \mathbb{R}^s$ is determined by performing power method iterations on the matrix:

$$\boldsymbol{\Lambda} \triangleq \frac{1}{|\mathcal{I}^0|} \sum_{j \in \mathcal{I}^0} \left(\frac{\boldsymbol{a}_{i,\hat{\mathcal{S}}} \boldsymbol{a}_{i,\hat{\mathcal{S}}}^T}{\|\boldsymbol{a}_{i,\hat{\mathcal{S}}}\|_2^2}\right). \tag{5.26}$$

With the structure of the iterations of SPARTA provided in (Equation 5.24) and the structure provided in (Equation 5.17) in mind, we can now easily derive the corresponding deep architecture by introducing a preconditioning matrix $\boldsymbol{G}^i$ at each iteration in lieu of α, and cast it as

a trainable parameter. Ergo, UPR-SPARTA can be presented via the mathematical structure of the proposed UPR architecture in the previous part, as we define the computational dynamics of the l-th layer of the UPR-SPARTA architecture according to (Equation 5.18). Furthermore, the dynamics of the overall network with L layers will be the same as (Equation 5.19) and the primary trainable parameters for the proposed decoder module become $\mathbf{\Gamma} = \{\boldsymbol{G}_0, \boldsymbol{G}_1, \cdots, \boldsymbol{G}_{L-1}\}$, as described in the previous section.

For the sake of completeness, the inner computation dynamics of UPR-SPARTA is summarized below:

UPR-SPARTA Computation Dynamics:

Initialize $\boldsymbol{z}_0$ according to the discussion above (Equation 5.26).

Every layer $i \in \{0, 1, \ldots, L-1\}$ is tuned to compute:

$$\boldsymbol{g}_{\boldsymbol{\gamma}_i}(\boldsymbol{z}) = \mathcal{H}_s \left(\boldsymbol{z} - \boldsymbol{G}^i \sum_{j \in \mathcal{I}_{k+1}} \boldsymbol{u}_j \boldsymbol{a}_j \right), \tag{5.27}$$

and

$$\boldsymbol{u}_j = \left(\boldsymbol{a}_j^T \boldsymbol{z} - [\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x})]_j \frac{\boldsymbol{a}_j^T \boldsymbol{z}}{|\boldsymbol{a}_j^T \boldsymbol{z}|} \right), \tag{5.28}$$

where $\boldsymbol{z}$ is the input to the l-th layer, and $\boldsymbol{\gamma}_i = \{\boldsymbol{G}_i \in \mathbb{S}_+\}$. The overall dynamics of UPR-SPARTA is given by:

$$\mathcal{G}_{\mathbf{\Gamma}}^L(\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}); \boldsymbol{z}_0) = \boldsymbol{g}_{\gamma_{L-1}} \circ \boldsymbol{g}_{\boldsymbol{\gamma}_{L-2}} \circ \cdots \circ \boldsymbol{g}_{\boldsymbol{\gamma}_0}(\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}); \boldsymbol{z}_0). \tag{5.29}$$

• ***UPR-IRWF: Conventional Phase Retrieval.*** We now consider the conventional phase retrieval problem, and similar to our previous approach, consider the unfolding of the IRWF algorithm for such problems. In addition, we propose the UPR-IRWF deep architecture as a decoder module to perform the signal recovery task for a conventional phase retrieval problem. WF based algorithms have had an unparalleled influence on the world of phase retrieval and IRWF is no exception. In fact, IRWF's performance made it of particular interest to further explore. Specifically, as an immediate extension from RWF, IRWF was developed to tackle the following optimization problem for the recovery purposes:

$$\min_{\boldsymbol{z}\in\mathbb{C}^m} \ell(\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x});\boldsymbol{z}) \equiv \frac{1}{2m}\Big\|\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}) - \mathcal{E}_{\boldsymbol{A}}(\boldsymbol{z})\Big\|_2^2. \tag{5.30}$$

The iterations of the IRWF algorithm for finding the critical points of the non-convex problem in (Equation 5.30) can be simply explained as follows. Starting from a proper initial point $\boldsymbol{z}^0$ (more on this below), the IRWF algorithm generates a sequence of points $\{\boldsymbol{z}^0, \boldsymbol{z}^1, \boldsymbol{z}^2, \cdots\}$ according to the following update rule:

$$\boldsymbol{z}^{k+1} = \boldsymbol{z}^k - \alpha\nabla_{\boldsymbol{z}}\ell(\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x});\boldsymbol{z}), \tag{5.31}$$

where α is some positive step-size and the gradient of the objective function is given by:

$$\nabla_{\boldsymbol{z}}\ell(\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x});\boldsymbol{z}) = \boldsymbol{A}^H\left(\boldsymbol{A}\boldsymbol{z} - \mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}) \odot \mathrm{Ph}(\boldsymbol{A}\boldsymbol{z})\right), \tag{5.32}$$

where the function $\mathrm{Ph}(\boldsymbol{z})$ is applied element-wise and captures the phase of the vector argument; e.g., for real valued signals $\mathrm{Ph}(\boldsymbol{z}) = \mathrm{sign}(\boldsymbol{z})$.

For initialization, [155] implements a method different than the commonly used spectral initialization, which benefits from a lower-complexity than that of the spectral method. In particular, the starting point is initialized as $\boldsymbol{x}_0 = \lambda_0 \boldsymbol{z}$, where $\lambda_0 \approx \sqrt{\pi/2}$, and $\boldsymbol{z}$ is the leading eigenvector of the matrix $(1/m)\sum_{i=1}^{m}[\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x})]_i \boldsymbol{a}_i \boldsymbol{a}_i^H$.

In light of the above description, we can view the mathematical structure of the proposed UPR architecture in a similar fashion as we did for UPR-SPARTA. We define the computational dynamics of the UPR-IRWF architecture as follows:

UPR-IRWF Computation Dynamics:

Initialize $\boldsymbol{z}_0$ according to the discussion below (Equation 5.32).

Every layer $i \in \{0, 1, \ldots, L-1\}$ is tuned to compute:

$$\boldsymbol{g}_{\boldsymbol{\gamma}_i}(\boldsymbol{z}) = \boldsymbol{z} - \boldsymbol{G}^i \boldsymbol{u}, \tag{5.33}$$

and

$$\boldsymbol{u} = \boldsymbol{A}^H \left(\boldsymbol{A}\boldsymbol{z} - \mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}) \odot \mathrm{sign}(\boldsymbol{A}\boldsymbol{z}) \right), \tag{5.34}$$

where $\boldsymbol{z}$ is the input to the l-th layer, and $\boldsymbol{\gamma}_i = \{\boldsymbol{G}^i \in \mathbb{S}_+\}$. The overall dynamics of UPR-IRWF is given by:

$$\mathcal{G}^L_{\boldsymbol{\Gamma}}(\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}); \boldsymbol{z}^0) = \boldsymbol{g}_{\boldsymbol{\gamma}_{L-1}} \circ \boldsymbol{g}_{\boldsymbol{\gamma}_{L-2}} \circ \cdots \circ \boldsymbol{g}_{\boldsymbol{\gamma}_0}(\mathcal{E}_{\boldsymbol{A}}(\boldsymbol{x}); \boldsymbol{z}_0). \tag{5.35}$$

5.4 Numerical Results

In this section, we investigate the performance of the proposed UPR-SPARTA and UPR-IRWF frameworks for the task of phase retrieval through various numerical experiments and compare their performance relative to their base-line state-of-the-art SPARTA [157] and IRWF [155] algorithms. The numerical experiments are conducted for real-valued Gaussian signals per convention, and are averaged over 100 Monte-Carlo trials. The proposed deep architectures are implemented using the *PyTorch* library [9], and the optimization (training) of the networks are carried out using the Adam stochastic optimizer [84] with a learning rate of 10^{-4} and for 100 epochs. For a fair comparison, the parameters of all competing algorithms are initialized to their suggested values as reported in [157] and [155]. As a figure of merit for evaluating the performance of the proposed methodologies, we make use of the empirical success rate (ESR) and the relative mean-square error metrics. If a signal $\boldsymbol{x}$ is to be recovered, we define the relative MSE metric for the recovered signal $\boldsymbol{y}$ as:

$$\text{Relative MSE} \triangleq \frac{\mathrm{D}(\boldsymbol{y}, \boldsymbol{x})}{\|\boldsymbol{x}\|_2}, \tag{5.36}$$

where we declare success for the recovery trial if the estimated signal assumes a relative MSE less than 10^{-5}. For both UPR frameworks, we consider a diagonal structure on the pre-conditioning matrices, i.e., we set $\boldsymbol{G}_i = \boldsymbol{W}_i \boldsymbol{W}_i^T$, where $\boldsymbol{W}_i = \text{Diag}(\alpha_0, \alpha_1, \cdots, \alpha_{n-1})$, and perform the training over $\boldsymbol{W}_i$, to ensure the positive-definiteness of the resulting $\boldsymbol{G}_i$.

5.4.1 Performance of the Proposed UPR-SPARTA

In this part, we evaluate the performance of the proposed UPR-SPARTA framework for the case of recovering a k-sparse real-valued Gaussian signal. For training purposes, we generate a training data-set of size 2048 where each data point $\boldsymbol{x} \in \mathbb{R}^n$ in the data-set follows a standard Gaussian distribution whose $(n-k)$ entries set to zero are chosen randomly and uniformly. Accordingly, a testing data-set of size 2048 with the same setting is generated using which we evaluate the performance of the proposed UPR-SPARTA and the base-line SPARTA algorithm. For all experiments in this part, the UPR-SPARTA is implemented with $L = 20$ layers (iterations) and we let the base-line SPARTA algorithm run for the same number of iterations.

We first provide the exact recovery performance of the UPR-SPARTA in terms of the ESR metric defined previously, based on conducting 100 Monte-Carlo trials. For this purpose, we set the signal dimension to $n = 100$ and the sparsity level to $k = 5$, while the ratio (m/n) increases from 0.1 to 3.0. In order to show the effectiveness of the learned parameters, we compare the performance of the proposed approach with the standard SPARTA algorithm in the following scenarios:

• *Case 1*: The standard SPARTA algorithm with a randomly generated measurement matrix $\boldsymbol{A} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ and a fixed step-size for the iterations as reported in [157].

• *Case 2*: The UPR-SPARTA algorithm with a learned measurement matrix $\boldsymbol{A}$ and a fixed step-size whose value is similar to the previous case.

• *Case 3*: The UPR-SPARTA algorithm with the learned pre-conditioning matrices $\{\boldsymbol{G}_i\}_i$ and for a fixed randomly generated $\boldsymbol{A}$, same as *Case 1.*

• *Case 4*: The UPR-SPARTA algorithm with the learned measurement matrix $\boldsymbol{A}$ and pre-conditioning matrices $\{\boldsymbol{G}_i\}_i$.

Figure 22 demonstrates the ESR versus the number of measurements (m/n) for the above scenario. It can be observed from Figure 22 that the proposed methodology significantly outperforms the standard state-of-the-art SPARTA algorithm for recovering a sparse signal in all considered cases, which is further evidence for the effectiveness of the learned parameters. Moreover, a comparison between the UPR-SPARTA with learned preconditioning matrices (Case 2) and the standard SPARTA algorithm reveals that for applications where the measurement matrix is imposed by the physics of system, the learning of the preconditioning matrices can significantly increase the performance of the recovery. On the other hand, considering a learning over the measurement matrix $\boldsymbol{A}$ only and employing fixed step-sizes (Case 3) further indicates the effectiveness of learning task-specific measurement matrices tailored to the application at hand can significantly increase the performance of the recovery algorithm. Finally, it can be observed from Figure 22 that the proposed UPR-SPARTA, while allowing a learning over all system parameters, outperforms all other cases and achieves a significantly higher ESR metric along all measurement ratios (m/n). We note that such a significant gain achieved by the pro-

posed approach for Cases 2-4 is due to the hybrid model-based and data-driven nature of the proposed methodology.

We further note that learning the preconditioning matrices must lead to accelerating the convergence of the proposed methodology. In order to numerically validate this claim, we perform a per-layer analysis of the proposed UPR-SPARTA framework in terms of the achieved relative MSE. Furthermore, such an analysis is driven possible due to the model-driven nature of the proposed approach which results in interpretable deep architectures as opposed to conventional black-box data-driven approaches. Figure 23 demonstrates the relative MSE versus number of iterations/layers for the case of recovering a $k = 5$ sparse signal $\boldsymbol{x} \in \mathbb{R}^n$, where $n = m = 300$. Observing Figure 23, it can be deduced that the proposed methodology achieves a far lower relative MSE much faster than that of the standard SPARTA.

Our final experiment in this part is an analysis of the ESR versus the sparsity level k, when the signal dimension and number of measurements are set to $n = m = 150$. Figure 24 illustrates the ESR metric versus the sparsity level k for the considered scenario. It can be observed from Figure 24 that the proposed methodology has a far superior performance than that of the base-line SPARTA algorithm.

5.4.2 Performance of the Proposed UPR-IRWF

In this part, we investigate the performance of the proposed UPR-IRWF specifically tailored for conventional phase retrieval application where no prior knowledge is assumed on the underlying signal of interest (e.g., sparsity). Similar to the previous subsection, we focus on the recovering of a real-valued Gaussian signal $\boldsymbol{x} \in \mathbb{R}^n$ from the phase-less measurements. In

particular, we employ the UPR-IRWF framework with $L = 50$ layers (iterations), and we let the base-line IRWF algorithm run for the same number of iterations for a fair comparison.

For the first experiment, we evaluate the performance of the proposed UPR-IRWF in terms of its ESR versus number of measurements to signal-length ratio (m/n) and compare its performance with the standard state-of-the-art IRWF algorithm in the following scenarios:

• *Case 1*: The standard IRWF algorithm with a randomly generated measurement matrix $\boldsymbol{A} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ and a fixed step-size for the iterations as reported in [155].

• *Case 2*: The UPR-IRWF algorithm with a learned measurement matrix $\boldsymbol{A}$ and a fixed step-size whose value is similar to the previous case.

• *Case 3*: The UPR-IRWF algorithm with the learned preconditioning matrices $\{\boldsymbol{G}_i\}_i$ and for a fixed randomly generated $\boldsymbol{A}$, same as *Case 1*.

• *Case 4*: The UPR-IRWF algorithm with the learned measurement matrix $\boldsymbol{A}$ and preconditioning matrices $\{\boldsymbol{G}_i\}_i$.

Figure 25 demonstrates the ESR versus measurement to signal-length ratio (m/n) when the signal length is fixed and set to $n = 100$. It can be clearly observed from Figure 25 that the proposed methodology significantly outperforms the base-line IRWF algorithm in terms of ESR in all scenarios. As for Case 3, in which we consider the scenario where the measurement matrix is fixed (i.e. imposed by the physics of the system) and is known *a priori*, the learning of the preconditioning matrices significantly improves the performance of the underlying signal recovery algorithm. This supports the proposition that a judicious design of the preconditioning

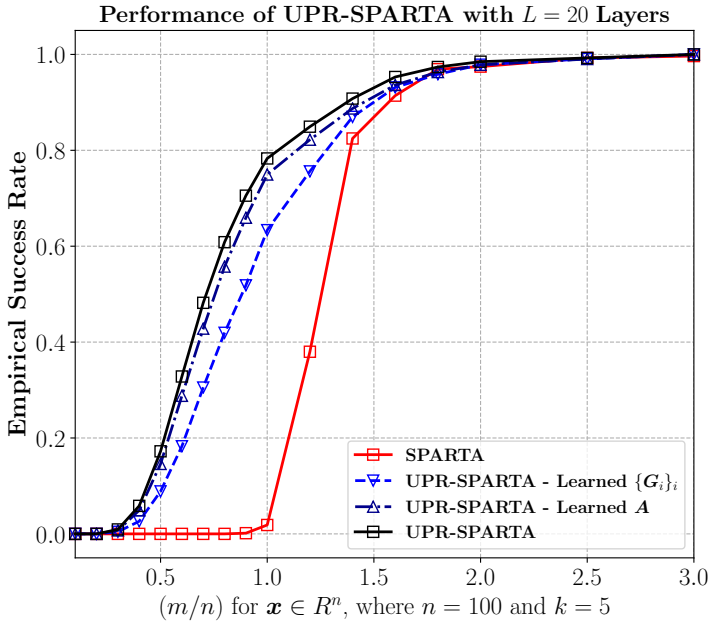


Figure 22. Empirical success rate versus the measurement to signal-length ratio (m/n) for $\boldsymbol{x} \in \mathbb{R}^n$ with $n = 100$ and $k = 5$ nonzero entries.

matrices, when the number of layers (iterations) are fixed, can indeed result in an accelerated convergence, thus immensely improving the performance. On the other hand, focusing on the Case 2 in which we only learn the measurement matrix while employing a fixed scalar step-size, further reveals the effectiveness of learning a proper task-specific data-driven measurement matrix akin to the problem of interest. Finally, it is evident that the proposed UPR-IRWF with both learned $\boldsymbol{A}$ and $\{\boldsymbol{G}_i\}_i$ outperforms all other cases.

Now, we numerically investigate the convergence properties of the proposed UPR-IRWF as compared to the standard IRWF algorithm via per-layer relative MSE analysis. We again stress that such an evaluation is possible due to the model-based nature of the proposed methodology—

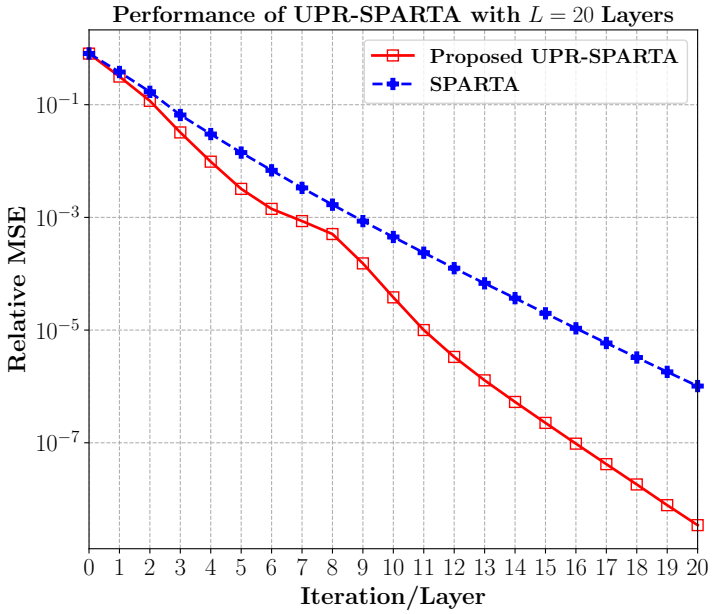


Figure 23. Convergence behavior of the proposed UPR-SPARTA as compared to the original SPARTA algorithm for the case of $\boldsymbol{x} \in \mathbb{R}^n$, where $n = m = 300$.

something that cannot be achieved when using conventional purely data-driven black-box techniques. For this experiment, we set the signal dimension and number of measurements as $(n, m) = (100, 600)$. Figure 26 demonstrates the relative MSE versus iteration (layer) number. It can be clearly observed from Figure 26 that the proposed methodology benefits from accelerated convergence properties and is able to converge to an optimal point of the objective function with as few as $L = 30$ layers. This in turn reveals that the proposed methodology can be further truncated to have much fewer iterations, which will reduce the computational cost of the overall algorithm.

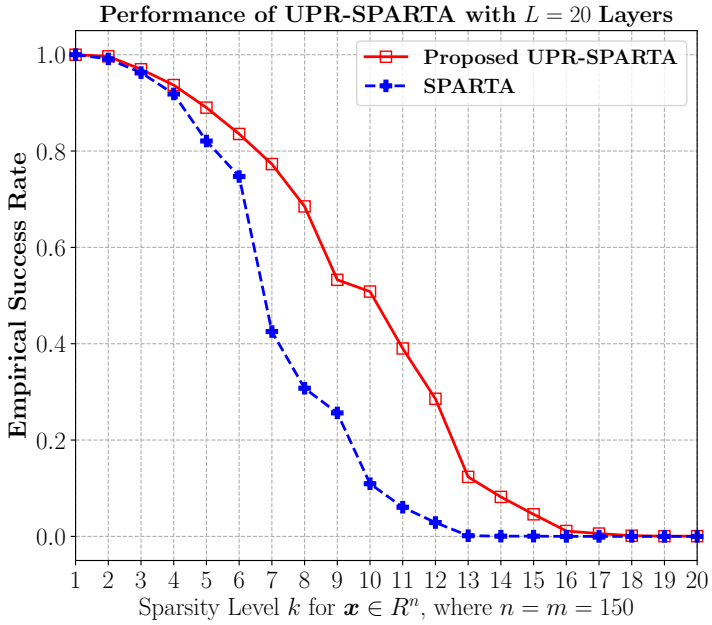


Figure 24. Empirical success rate versus the sparsity level k for $\boldsymbol{x} \in \mathbb{R}^n$, where $n = m = 150$.

5.5 Conclusion

In this paper, we proposed a new approach to the phase retrieval paper via developing hybrid model-aware and data-driven deep architectures, referred to as Unfolded Phase Retrieval (UPR). Specifically, we focused on the conventional phase retrieval problem and the sparse phase retrieval problem. We considered a joint design of the sensing matrix and the signal recovery algorithm, while utilizing the deep unfolding technique in the process. Such an approach allowed us to exploit the data for better accuracy and performance, and attain trusted results owing to UPR's model-based roots and the resulting interpretability. The UPR framework required less data for effective training due to having fewer parameters, and enjoyed an enhanced convergence

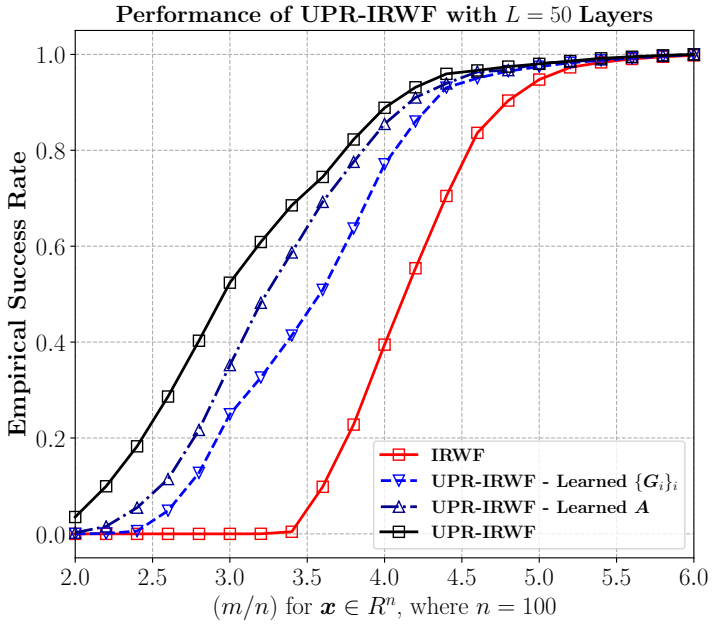


Figure 25. Empirical success rate versus the measurement to signal-length ratio (m/n) for $\boldsymbol{x} \in \mathbb{R}^n$ with $n = 100$.

rate. The unique capability of UPR to allow for designing task-specific sensing matrices further enhanced the performance of the system.

The performance of UPR was compared with the state-of-the-art model-based phase retrieval algorithms. Our results displayed a significant improvement in performance compared to such algorithms thanks to UPR's hybrid model-aware and data-driven nature.

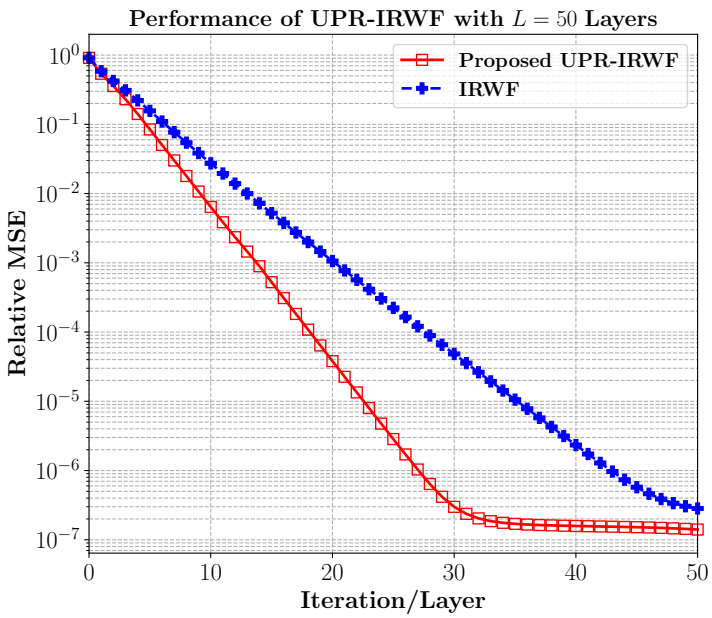


Figure 26. Convergence behavior of the proposed UPR-IRWF as compared to the original IRWF algorithm for the case of $\boldsymbol{x} \in \mathbb{R}^n$, with $(n, m) = (100, 600)$.

I-D

Model-Based Deep Learning for Autonomous Vehicles

CHAPTER 6

DEEP RADAR WAVEFORM DESIGN FOR EFFICIENT AUTOMOTIVE RADAR SENSING

Overview: In radar systems, unimodular (or constant-modulus) waveform design plays an important role in achieving better clutter/interference rejection, as well as a more accurate estimation of the target parameters. The design of such sequences has been studied widely in the last few decades, with most design algorithms requiring sophisticated *a priori* knowledge of environmental parameters which may be difficult to obtain in real-time scenarios. In this paper, we propose a novel hybrid model-driven and data-driven architecture that adapts to the ever changing environment and allows for adaptive unimodular waveform design. In particular, the approach lays the groundwork for developing extremely low-cost waveform design and processing frameworks for radar systems deployed in autonomous vehicles. The proposed model-based deep architecture imitates a well-known unimodular signal design algorithm in its structure, and can quickly infer statistical information from the environment using the observed data. Our numerical experiments portray the advantages of using the proposed method for efficient radar waveform design in time-varying environments.

Keywords: automotive radar, deep learning, deep unfolding, data-driven approaches, model-based signal processing, unimodular waveform design

---

6.1 Introduction

Waveform design for active sensing has been of interest to engineers, system theorists and mathematicians in the last sixty years. In the last decade, however, civilian radar applications such as the use of radar in autonomous cars have attracted much-deserved attention towards enhanced resolvability for advanced safety. In vehicular applications, the radar technology offers excellent resolvability and immunity to bad weather conditions in comparison to visible and infrared imaging techniques. However, the cost overheads of ultra-high frequency radar signal processors is excessive, which limits a mass deployment of radar-based advanced vehicular safety features. Reliable and low-cost deep learning-based algorithms and hardware may promise a solution to such difficulties.

Automotive radar sensors usually operate at the 24-77 GHz frequency domain, and are able to measure the target range, radial velocity and azimuth angle simultaneously even in multi-target situations [190]. The quality of these measurements, however, depends strongly on the transmit waveform design process. There exist several approaches to tackle the task of waveform design in such radar systems [32, 191–193, 193–219], which rely on known radar models. In such model-based approaches, one only considers a simplified mathematical model and often do not take into account the intricate interactions innate to the kind of complex information systems that are common in real world. On the other hand, in a purely data-driven approach, including deep learning techniques, one do not need an explicit mathematical model of the problem, and should be able to use the available data at hand for designing the waveforms. The major shortcoming of the data-driven approach stems from the fact that it is unclear how

to incorporate the existing knowledge of the system model in the processing stage. Namely, purely data-driven approaches have a wider applicability at the cost of interpretability, and in some cases, reliability [36, 223]. In this paper, we seek to bridge the gap between the model-based and data-driven approaches, and propose a novel methodology in order to design efficient waveforms for automotive radars by making use of the *deep unfolding framework* [31,31,36,224]. Specifically, the purpose of performing waveform design for a radar systems is to capture as much as information possible from the the environment, where in fact, *the transmitted waveform can now be interpreted as a channel that retrieves and collects information.* In light of this, we employ the deep unfolding framework that aims to take the well-established iterative approaches, and design a deep architecture for waveform design in radar systems under different unimodular signal constraint, and boost the performance of the underlying inference optimization algorithm in terms of speed of convergence and effectiveness.

6.2 Radar Model— and Signal Design Formulation

Consider a radar system transmitting unimodular codes used to modulate a train of sub-pulses. Let $\boldsymbol{s} = [s_1 \ s_2 \ \cdots \ s_N]^T \in \mathbb{C}^N$ denote the complex-valued probing sequence to be designed. Under some mild assumptions, the received discrete-time base-band signal $\boldsymbol{y}$, after signal compression and range focusing, can be modeled as follows [193]:

$$\boldsymbol{y} = \boldsymbol{A}^H \boldsymbol{\alpha} + \boldsymbol{\epsilon}, \tag{6.1}$$

where

$$\boldsymbol{A}^H = \begin{bmatrix} s_1 & 0 & \cdots & 0 & s_N & s_{N-1} & \cdots & s_2 \\ s_2 & s_1 & & \vdots & 0 & s_N & & \vdots \\ \vdots & \vdots & \ddots & 0 & \vdots & \vdots & \ddots & s_N \\ s_N & s_{N-1} & \cdots & s_1 & 0 & 0 & \cdots & 0 \end{bmatrix}, \tag{6.2}$$

$$\boldsymbol{\alpha} = [\alpha_0 \ \alpha_1 \ \cdots \ \alpha_{N-1} \ \alpha_{-N+1} \ \cdots \ \alpha_{-1}]^T \in \mathbb{C}^{2N-1.} \tag{6.3}$$

Here, the parameter α_0 is the scattering coefficient of the current range cell, while $\{\alpha_k\}_{k\neq 0}$ are that of the neighboring cells adding to clutter, and $\boldsymbol{\epsilon}$ is the signal independent interference term. The main goal in a radar system given the measurement model in (Equation 6.1) is typically to design the probing signal $\boldsymbol{s}$ such that it allows for an accurate recovery of the target scattering coefficient α_0.

Note that, in model-based radar waveform design, the statistics of the interference and noise is usually assumed to be known, e.g., through stand-alone prescan procedures. Under such conditions, the waveform design boils down to constrained quadratic or fractional quadratic program as detailed in previous work [193, 214, 215, 217, 219]. An example for waveform design criteria comes from the waveform's merit for resolvability along with clutter rejection.

Namely, using a matched filter (MF) in the pulse compression stage, one can look for codes that maximize the following criterion:

$$f(\boldsymbol{s}) = \frac{n(\boldsymbol{s})}{d(\boldsymbol{s})} \triangleq \frac{|\boldsymbol{s}^H \boldsymbol{y}|^2}{\sum_{k\neq 0} |\boldsymbol{s}^H \boldsymbol{J}_k \boldsymbol{y}|^2} = \frac{\boldsymbol{s}^H \boldsymbol{A} \boldsymbol{s}}{\boldsymbol{s}^H \boldsymbol{B} \boldsymbol{s}}, \tag{6.4}$$

where $\boldsymbol{A} = \boldsymbol{y}\boldsymbol{y}^H$, $\boldsymbol{B} = \sum_{k\neq 0} \boldsymbol{J}_k \boldsymbol{A} \boldsymbol{J}_k^H$, and $\{\boldsymbol{J}_k\}$ are shift matrices satisfying $[\boldsymbol{J}_k]_{l,m} = [\boldsymbol{J}_{-k}^H]_{l,m} \triangleq \delta_{m-l-k}$, with $\delta_{(\cdot)}$ denotes the Kronecker delta function. Note that the above function can be interpreted as an oracle to a signal-to-interference-noise (SINR) ratio as the numerator represents the signal power and the denominator represents the combined interference and noise power after applying the matched filter. We further note that, to lower the implementation cost, it is desirable to use unimodular codes, i.e. $s_k = e^{j\phi_k},\ \phi_k \in [0, 2\pi),\ k \in \{1, \ldots, N\}$, that attain the smallest peak-to-average ratio possible for transmit signals. As a result, one can consider the following fractional program in its general form for radar waveform design:

$$\max_{\boldsymbol{s}} \ \frac{\boldsymbol{s}^H \boldsymbol{A} \boldsymbol{s}}{\boldsymbol{s}^H \boldsymbol{B} \boldsymbol{s}}, \ \text{ s.t. } \ |s_k| = 1, \ \ k \in \{1, \ldots, N\} \tag{6.5}$$

Note that evaluating the objective function in (Equation 6.5), i.e. computing $f(\boldsymbol{s})$, only requires the knowledge of the transmit sequence $\boldsymbol{s}$ and the observed vector $\boldsymbol{y}$ at the receiver. Nevertheless, solving the above optimization program is still NP-hard and very hard to tackle in general. In order to approximate the solution, one can resort to power method-like iterations specifically designed to tackle unimodular quadratic programs (UQPs) [214]. In what follows, we reformulate the problem of (Equation 6.5) as a UQP, and present the corresponding

power method-like iterations that lays the groundwork for our proposed hybrid model-aware and data-driven *adaptive* waveform design framework.

Observe that both the numerator $n(\boldsymbol{s})$ and the denominator $d(\boldsymbol{s})$ of the objective function $f(\boldsymbol{s})$ are quadratic in $\boldsymbol{s}$. Hence, in order to tackle the maximization of (Equation 6.4) (or equivalently tackling (Equation 6.5)) we resort to fractional programming techniques [225, 226]. Since $f(\boldsymbol{s})$, the SINR, is finite, we must have that $d(\boldsymbol{s}) = \boldsymbol{s}^H \boldsymbol{B}\boldsymbol{s} > 0$. In addition, let $\boldsymbol{s}_\star$ denote the current value of the code sequence $\boldsymbol{s}$. Then, we define

$$e(\boldsymbol{s}) \triangleq n(\boldsymbol{s}) - f(\boldsymbol{s}_\star) d(\boldsymbol{s}), \tag{6.6}$$

$$\boldsymbol{s}_\dagger = \arg\max_{\boldsymbol{s}} \; e(\boldsymbol{s}). \tag{6.7}$$

Henceforth, it can be easily verified by the virtue of (Equation 6.7) that $e(\boldsymbol{s}_\dagger) \geq e(\boldsymbol{s}_\star) = 0$. As a result, we have that $e(\boldsymbol{s}_\dagger) = n(\boldsymbol{s}_\dagger) - f(\boldsymbol{s}_\star) d(\boldsymbol{s}_\dagger) \geq 0$ implying that

$$f(\boldsymbol{s}_\dagger) \geq f(\boldsymbol{s}_\star), \tag{6.8}$$

as $d(\boldsymbol{s}_\dagger) > 0$. In other words, we can argue that with respect to $\boldsymbol{s}_\star$, the $\boldsymbol{s}_\dagger$ increases the objective function $f(\boldsymbol{s})$. It is noteworthy to mention that for the criteria in (Equation 6.8) to hold, it is sufficient for $\boldsymbol{s}_\dagger$ to satisfy $e(\boldsymbol{s}_\dagger) \geq e(\boldsymbol{s}_\star)$ and that $\boldsymbol{s}_\dagger$ shall not necessarily be the maximizer of $e(\boldsymbol{s})$.

For a given $\boldsymbol{s}_\star$ maximizer of (Equation 6.5) we have that:

$$e(\boldsymbol{s}) = \boldsymbol{s}^H \boldsymbol{A}\boldsymbol{s} - f(\boldsymbol{s}_\star)\left(\boldsymbol{s}^H \boldsymbol{B}\boldsymbol{s}\right) = \boldsymbol{s}^H \underbrace{\left(\boldsymbol{A} - f(\boldsymbol{s}_\star)\boldsymbol{B}\right)}_{\triangleq \tilde{\boldsymbol{\chi}}} \boldsymbol{s}$$

Now, in order to ensure that $\tilde{\boldsymbol{\chi}}$ is positive definite, one can perform a diagonal loading procedure by defining $\boldsymbol{\chi} \triangleq \tilde{\boldsymbol{\chi}} + \lambda \boldsymbol{I}_N$, where $\lambda \geq \max\{0, -\lambda_{\min}(\tilde{\boldsymbol{\chi}})\}$. Next, the optimization problem of (Equation 6.5) can be cast as the following UQP [214]:

$$\max_{\boldsymbol{s}} \quad \boldsymbol{s}^H \boldsymbol{\chi} \boldsymbol{s}, \quad \text{s.t.} \quad |s_k| = 1, \quad k \in \{1, \ldots, N\}. \tag{6.9}$$

In order to efficiently tackle (Equation 6.9), a set of *power method-like iterations* (PMLI) were introduced in [214, 215] that can be used to monotonically increase the objective value in (Equation 6.9) using the following nearest-vector problem:

$$\min_{\boldsymbol{s}^{(n+1)}} \left\| \boldsymbol{s}^{(n+1)} - \boldsymbol{\chi}\boldsymbol{s}^{(n)} \right\|_2, \quad \text{s.t.} \quad \left| s_k^{(n+1)} \right| = 1, \, \forall \, k. \tag{6.10}$$

The solution to (Equation 6.10) can be computed analytically and is given as follows [214, 215]:

$$\boldsymbol{s}^{(n+1)} = e^{j \arg(\boldsymbol{\chi}\boldsymbol{s}^{(n)})}. \tag{6.11}$$

where n denotes the internal iteration number, and $\boldsymbol{s}^{(0)}$ is the current value of $\boldsymbol{s}$. One can continue updating $\boldsymbol{s}$ until convergence in the objective of (Equation 6.5), or for a fixed number

of steps, say L. These iterations are already shown to provide a monotonic behavior of the quadratic objective (no matter what the signal constraints are), and subsume the well-known power method as a special case. Such a general approach to computationally efficient quadratic programming that can handle various signal constraints (many of which cause the problems to become NP-hard) opens new avenues in signal processing in low-cost scenarios.

Note that, in many practical scenarios, one might not have access to the *a priori* information about environmental parameters. In the following, we aim to devise a hybrid data-driven and model-based approach that allows us to jointly design adaptive transmit code sequences while learning these parameters given the fact that the environmental information are in fact embedded into the observed received signal $\boldsymbol{y}$. Namely, we propose a novel neural network structure for waveform design, **D**eep **E**volutionary **Co**gnitive **R**adar (DECoR), by considering the above power method-like iterations as a baseline algorithm for the design of a model-based deep neural network. In particular, we consider an over-parametrization of the power method-like iterations and unfold them onto the layers of a deep neural network. Each layer of the resulting network is designed such that it imitates one iteration of the form (Equation 6.11). Consequently, the resulting deep architecture is model-aware, uses the same non-linear operations as those in the power method, and hence, is interpretable (as opposed to general deep learning models). The structure yet allow us to utilize data-driven approaches to optimize the parameters of the network in an online learning manner—making the resulting network a great candidate for reliable adaptive waveform design in automotive radar applications.

6.3 The DECoR Architecture for Signal Design

Consider the dynamics of a general fully connected deep neural network. Let $\tilde{g}_{\phi_i}$ be defined as

$$\tilde{g}_{\phi_i}(\boldsymbol{z}) = a(\boldsymbol{u}), \text{ where } \boldsymbol{u} = \boldsymbol{W}_i \boldsymbol{z}, \tag{6.12}$$

where $\phi_i = \{\boldsymbol{W}_i\}$ denotes the set of parameters of the function g_{ϕ_i}, and $a(\cdot)$ denotes a non-linear activation function. Then, given an input $\boldsymbol{x}_0$, the dynamics of a fully connected neural network with L layers can be expressed as follows:

$$\boldsymbol{x}_L = \mathcal{F}\left(\boldsymbol{x}_0; \boldsymbol{\Upsilon}\right) = \tilde{g}_{\phi_{L-1}} \circ \tilde{g}_{\phi_{L-2}} \circ \cdots \circ \tilde{g}_{\phi_0}(\boldsymbol{x}_0), \tag{6.13}$$

where, for a general DNN, $\boldsymbol{\Upsilon} = \{\phi_i\}_{i=0}^{L-1}$ denotes the set of weight matrices $\boldsymbol{W}_i$ for each layer. Now, consider the power method-like iterations of the form (Equation 6.11). The connection between the two becomes clear by paying attention to the fact that a fully connected DNN with an activation function defined as $a(\boldsymbol{x}) = e^{j \arg(\boldsymbol{x})}$, and parameterized on a matrix $\boldsymbol{W}$ (that is tied along the layers), boils down to performing L iterations of the PMLI. Therefore, one can immediately see that a fully connected DNN with the specific choice of non-linear activation function given by the projection operator $\mathcal{S}(\boldsymbol{x}) \triangleq e^{j \arg(\boldsymbol{x})}$ is an optimal architecture for waveform design with respect to the power method-like iterations extensively used in waveform design in various applications [227, 228]. Hence, power method-like iterations are perfect candidates for

unfolding into DNNs since they can be characterized by a linear step, followed by a possibly non-linear operation.

6.3.1 The Deep Evolutionary Cognitive Radar Architecture

The derivation begins by considering that in the vanilla PMLI algorithm, the matrix $\boldsymbol{\chi}$ is tied along all iterations. Hence, we enrich the PML iterations by introducing a weight matrix $\boldsymbol{\chi}_i$ *per iteration* i. Note that in the original PMLI algorithm, the matrix $\boldsymbol{\chi}$ changes from one outer iteration to another. Hence, such an over-parameterization of the iterations results in a deep architecture that is faithful to the original model-based signal design method. Such an over-parametrization yields the following computation model for our proposed deep architecture (DECoR). Let us define g_{ϕ_i} as

$$g_{\phi_i}(\boldsymbol{z}) = \mathcal{S}(\boldsymbol{u}), \quad \text{where} \quad \boldsymbol{u} = \boldsymbol{\chi}_i \boldsymbol{z}, \tag{6.14}$$

where $\phi_i = \{\boldsymbol{\chi}_i\}$ denotes the set of parameters of the function g_{ϕ_i}, and recall that the non-linear activation function is defined as $\mathcal{S}(\boldsymbol{x}) = e^{j \arg(\boldsymbol{x})}$ applied element-wise on the vector argument. Then, the dynamics of the proposed DECoR architecture with L layers can be expressed as:

$$\boldsymbol{s}_L = \mathcal{G}\left(\boldsymbol{s}_0; \boldsymbol{\Omega}\right) = g_{\phi_{L-1}} \circ g_{\phi_{L-2}} \circ \cdots \circ g_{\phi_0}(\boldsymbol{s}_0), \tag{6.15}$$

where $\boldsymbol{s}_0$ denotes some initial unimodular vector, and $\boldsymbol{\Omega} = \{\boldsymbol{\chi}_0, \ldots, \boldsymbol{\chi}_{L-1}\}$ denotes the set of trainable parameters of the network. The block diagram of the proposed architecture is depicted in Figure 27.

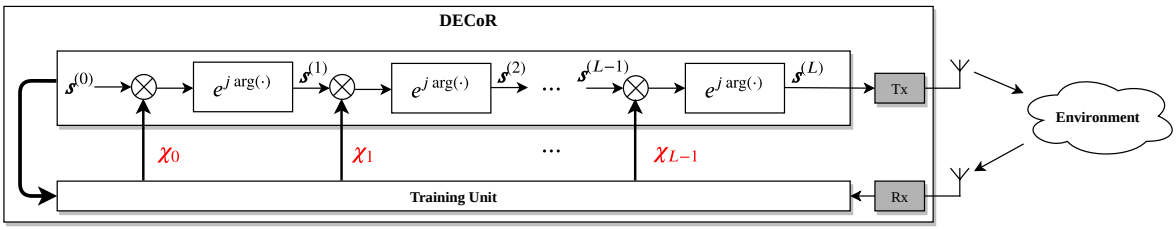


Figure 27. The proposed DECoR architecture for adaptive radar waveform design.

Our goal is to optimize the set of parameters $\mathbf{\Omega}$ of the proposed DECoR architecture using an online learning strategy that allows for fast adaptation to different environment. Intuitively, given the nature of the PML iterations, learning the parameters $\mathbf{\Omega} = \{\boldsymbol{W}_l\}_{l=0}^{L-1}$ corresponds to learning the information corresponding to the signal dependent interference and environmental noise profile.

6.3.2 The Proposed Online Learning Strategy

In an automotive radar application, the environment might undergo drastic changes along different coherent processing intervals, and the noise and interference statistics might vary as a result. Hence, it is natural to consider an online learning strategy for training the proposed DECoR architecture.

Let $\mathbf{\Omega}^{(t)}$ denote the set of parameters at time t. Then, the resulting code sequence given the set of parameters $\mathbf{\Omega}^{(t)}$ is simply given by the output of the last layer of the proposed DECoR architecture, i.e. $s_L^{(t)} = \mathcal{G}\left(\boldsymbol{s}_0; \mathbf{\Omega}^{(t)}\right)$. We define the goal of our online training procedure as learn-

ing the set of parameters $\mathbf{\Omega}^{(t+1)}$ such that the resulted code sequence $\boldsymbol{s}^{(t+1)} = \mathcal{G}\left(\boldsymbol{s}_0; \mathbf{\Omega}^{(t+1)}\right)$ satisfies the following criterion:

$$f(\boldsymbol{s}^{(t+1)}) \geq f(\boldsymbol{s}^{(t)}). \tag{6.16}$$

Accordingly, we propose the following *random walk-based* training strategy for optimizing the parameters of the proposed DECoR architecture in an online manner:

• ***Step 0*** (Initialization): Choose an arbitrary unimodular transmit sequence $\boldsymbol{s}_0 \in \mathbb{C}^N$, and set the training counter to $t = 0$. Initialize the radius σ of the search region to some positive constant c, and choose $\delta \in (0, 1]$. Further initialize the set of weight matrices $\boldsymbol{\Omega}^{(0)} = \{\boldsymbol{\chi}_i^{(0)}\}_{i=0}^{L-1}$ such that $\boldsymbol{\chi}_i^{(0)} \succ 0$, for $i \in \{0, \ldots, L-1\}$.

• ***Step 1*** (Random walk- generation): For $l \in \{0, \ldots, L-1\}$, generate B random lower triangular matrices $\boldsymbol{L}_l^0, \ldots, \boldsymbol{L}_l^{B-1} \sim \mathcal{CN}(0, \sigma \boldsymbol{I})$, and form the set of Hermitian positive-definite search direction matrices $\boldsymbol{D}_l^i = \boldsymbol{L}_l^i \boldsymbol{L}_l^{iH}$, for each layer l and for $i \in \{0, \ldots, B-1\}$, where $\boldsymbol{D}_l^i \in \mathbb{C}^{N \times N}$.

• ***Step 2*** (Random walk- perturbation): For $i \in \{0, \ldots, B-1\}$, form the set of possible candidate updates for the current parameter space $\boldsymbol{\Omega}^{(t)}$ as $\boldsymbol{\Omega}_i^{(t)} = \{\boldsymbol{\chi}_0^{(t)} + \boldsymbol{D}_0^i, \ldots, \boldsymbol{\chi}_{L-1}^{(t)} + \boldsymbol{D}_{L-1}^i\}$. Compute the corresponding B unimodular codes $\boldsymbol{s}_{L,i}^{(t)} = \mathcal{G}(\boldsymbol{s}_0; \boldsymbol{\Omega}_i^{(t)})$ for $i \in \{0, \ldots, B\}$ and form the set of training transmission codes as $\boldsymbol{S}^{(t)} = \{\boldsymbol{s}_{L,0}^{(t)}, \ldots, \boldsymbol{s}_{L,B-1}^{(t)}\}$.

• ***Step 3*** (Collecting information): Transmit the unimodular codes in the set $\boldsymbol{S}^{(t)}$ and obtain the corresponding set of received signals $\boldsymbol{Y} = \{\boldsymbol{y}_i^{(t)}\}_{i=0}^{B-1}$. Compute the function $f(\boldsymbol{s})$ for each transmit/receive pair $(\boldsymbol{s}_{L,i}^{(t)}, \boldsymbol{y}_i^{(t)})$ and construct the set of objective values as $\mathcal{F} = \{f(\boldsymbol{s}_{L,i}^{(t)})\}_{i=0}^{B-1}$.

• ***Step 4*** (Optimizing the DECoR architecture): Choose the current optimal parameter space using

$$i_\star = \underset{i \in [B]}{\arg\max} \; f(\boldsymbol{s}_{L,i}^{(t)}).$$

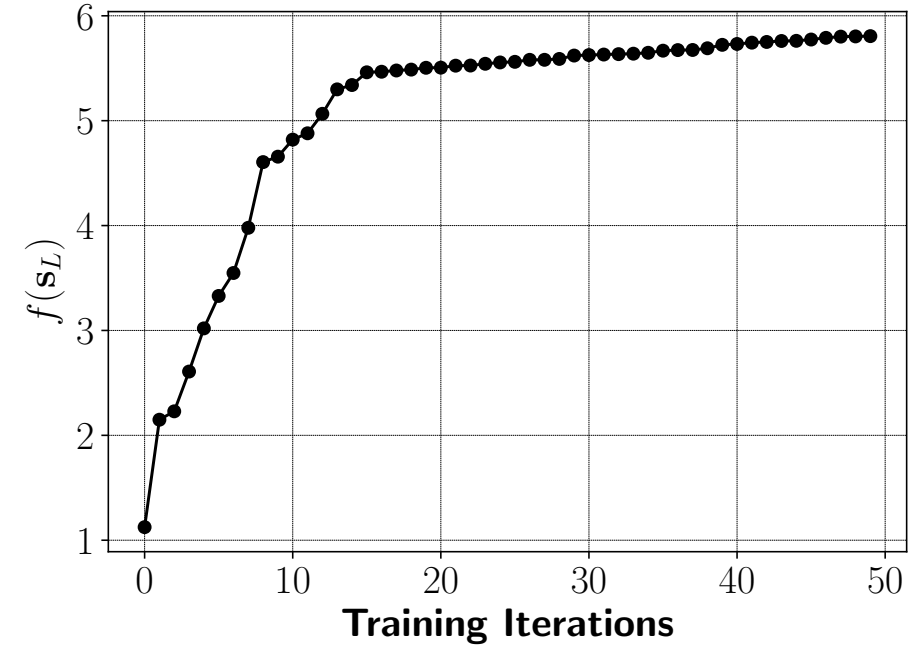
Update the network parameters if $f(\boldsymbol{s}_{L,i_\star}^{(t)}) \geq f(\boldsymbol{s}_L^{(t-1)})$ and set the search radius as $\sigma \leftarrow c$. Otherwise, only update the search radius as $\sigma \leftarrow \delta\sigma$. Continue the online learning by going to Step 1.

The above proposed online learning strategy for the proposed DECoR architecture is an amalgamation of natural evolutionary optimization techniques and policy optimization in reinforcement learning. In particular, the increase in the objective function $f(\boldsymbol{s})$ can be seen as a task for an agent that is interacting with an unknown environment over the action space of $\boldsymbol{\Omega}$ and the corresponding unimodular code $\boldsymbol{s}_L = \mathcal{G}(\boldsymbol{s}_0, \boldsymbol{\Omega})$. Note that the power method-like iterations and the model of the system impose a positive definite constraint on the weight matrices $\{\boldsymbol{\chi}_i\}_{i=0}^{L-1}$. In order to impose such a constraint in incrementally learning the parameters $\boldsymbol{\Omega}$, we initialize each $\boldsymbol{\chi}_i^{(0)}$ with some positive-definite matrix. We then perform a random walk in the cone of positive definite matrices by forming positive definite search direction matrices $\boldsymbol{D}_l^i = \boldsymbol{L}_l^i \boldsymbol{L}_l^{iH}$. Such a training strategy results in a fast adaptation to the ever changing environment. Hence, the radar agent can continually perform the training on the fly.

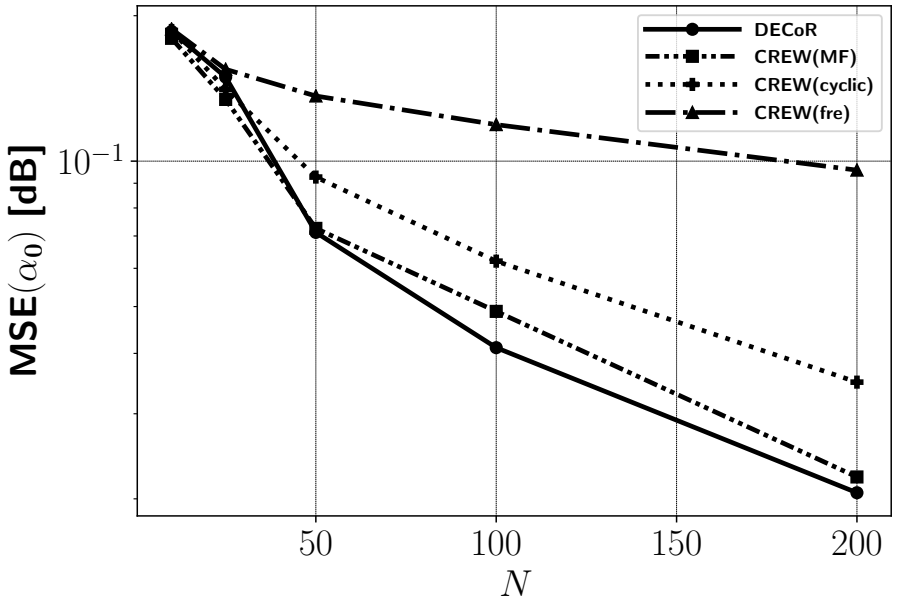
6.4 Results and Concluding Remarks

We begin by evaluating the performance and effectiveness of the proposed online learning strategy for optimizing the parameters of the DECoR architecture. For this experiment, we fix the total number of layers of the proposed DECoR architecture as $L = 30$. Throughout the simulations, we assume an environment with dynamics described in (Equation 6.1), and with more details in [193], with clutter power $\beta = 1$, and a noise covariance of $\boldsymbol{\Gamma} = \boldsymbol{I}$. These information were not made available to the DECoR architecture and we only use them for data generation purposes.

28(a) demonstrates the objective value $f(\boldsymbol{s}_L)$ in (Equation 6.4) vs. training iterations, for a code length of $N = 10$. It can be clearly seen that the proposed learning strategy and



(a)



(b)

Figure 28. Illustration of (a) the objective value $f(\boldsymbol{s}_L)$ of the DECoR vs. training iterations for a code length of $N = 10$, and (b) MSE values obtained by the different design algorithms for code lengths $N \in \{10, 25, 50, 100, 200\}$.

the corresponding DECoR architecture results in a *monotonically* increasing objective value $f(\boldsymbol{s}_L)$. Furthermore, note that the proposed learning algorithm optimizes the parameters of the proposed DECoR architecture very quickly. Next, we evaluate the performance of the presented hybrid model-based and data-driven architecture in terms of recovering the target coefficient α_0. In particular, we compare the performance of our method (DECoR) in designing unimodular codes with two state-of-the-art model-based algorithms: (a) CREW(cyclic) [215], a cyclic optimization of the transmit sequence and the receive filter, (b) CREW(MF) [215], a version of CREW(cyclic) that uses a matched filter as the receive filter, and (c) CREW(fre) [229], a frequency domain algorithm to jointly design transmit sequence and the receive filter. 28(b) illustrates the MSE of the estimated α_0 vs. code lengths $N \in \{10, 25, 50, 100, 200\}$. For each N, we perform the optimization of DECoR architecture by allowing the radar agent to interact with the environment for 50 training epochs. After the training is completed, we use the optimized architecture to generate the unimodular code sequence $\boldsymbol{s}_L$ and use a MF to estimate α_0. We let the aforementioned algorithms to perform the code design until convergence, while the presented DECoR architecture has been only afforded $L = 30$ layers (equivalent of L iterations).

It is evident that the proposed method significantly outperforms other state-of-the-art approaches. Although the DECoR framework does not have access to the statistics of the environmental parameters (as opposed to the other algorithms), it is able to learn them by exploiting the observed data from interaction with the environment.

I-E

Model-Based Deep Learning for Image Processing

CHAPTER 7

DEEP-URL: A MODEL-AWARE APPROACH TO BLIND DECONVOLUTION BASED ON DEEP UNFOLDED RICHARDSON-LUCY NETWORK

Overview: The lack of interpretability in current deep learning models causes serious concerns as they are extensively used for various life-critical applications. Hence, it is of paramount importance to develop interpretable deep learning models. In this paper, we consider the problem of blind deconvolution and propose a novel model-aware deep architecture that allows for the recovery of both the blur kernel and the sharp image from the blurred image. In particular, we propose the Deep Unfolded Richardson-Lucy (Deep-URL) framework — an interpretable deep-learning architecture that can be seen as an amalgamation of classical estimation technique and deep neural network, and consequently leads to improved performance. Our numerical investigations demonstrate significant improvement compared to state-of-the-art algorithms.

Keywords: Blind deconvolution, model-aware deep learning, machine learning, deep unfolding, non-convex optimization

7.1 Introduction

In digital photography, motion blur is a common and longstanding problem where the blurring is induced by the relative motion of the camera or the subject with respect to the other [231].

In classical image processing, such a motion blur is generally regarded as a motion kernel being applied on the original sharp image through a linear operation, *e.g.*, convolution. Often in practice, however, neither the blur kernel nor the original image is known *a priori*, and thus the task becomes to estimate both from the blurry input image. In image processing, the term *blind deconvolution* is often used to represent the task of image restoration without any explicit knowledge of the impulse response function, also known as the point-spread function (PSF) and the original sharp image [1, 231]. The blurred image $\boldsymbol{y}$ is typically formulated as:

$$\boldsymbol{y} = \boldsymbol{H} \circledast \boldsymbol{x} + \boldsymbol{n}, \tag{7.1}$$

where $\boldsymbol{x}$ and $\boldsymbol{H}$ are the unknown original clean image and the blur kernel, respectively, $\boldsymbol{n}$ is the additive measurement noise generally modeled as white Gaussian noise (AWGN) with variance σ^2, and $\circledast$ represents the 2D convolution operator. Hence, the task of blind deconvolution is to estimate a sharp $\boldsymbol{x}$ and the corresponding $\boldsymbol{H}$ from an infinite set of pairs $(\boldsymbol{x}, \boldsymbol{H})$ using the blurry image $\boldsymbol{y}$, making it an ill-posed and very challenging problem.

A judicious approach to such problems is to utilize some prior knowledge about the statistics of the natural image and/or motion kernels. There exists a multitude of algorithms to efficiently

estimate the image $\boldsymbol{x}$ and kernel $\boldsymbol{H}$ using prior knowledge of the model [232–234]. A majority of them are based on maximum-a-posterior (MAP) framework,

$$\begin{aligned}(\hat{\boldsymbol{x}}, \hat{\boldsymbol{H}}) &= \arg\max_{\boldsymbol{x},\boldsymbol{H}} \Pr\{\boldsymbol{x}, \boldsymbol{H} \mid \boldsymbol{y}\}, \\ &= \arg\max_{\boldsymbol{x},\boldsymbol{H}} \Pr\{\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{H}\} \Pr\{\boldsymbol{x}\} \Pr\{\boldsymbol{H}\}, \end{aligned} \tag{7.2}$$

where $\Pr\{\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{H}\}$ is the likelihood of the noisy output $\boldsymbol{y}$ given a certain $(\boldsymbol{x}, \boldsymbol{H})$, that corresponds to the data fidelity term, and $\Pr\{\boldsymbol{x}\}$ and $\Pr\{\boldsymbol{H}\}$ are the priors of the original image and blur kernel, respectively. Note that, Eq. (Equation 7.2) is correct under the assumption that the sharp original image $\boldsymbol{x}$ and the blur kernel $\boldsymbol{H}$ are independent. These MAP-based algorithms are often *iterative* in nature and usually rely on the sparsity-inducing regularizers, either in gradient domain [234–236] or more generally in sparsifying transformation domain [237]. However, the knowledge of the prior is not usually enough, for instance, Levin *et al.* [1] shows that MAP-based methods may lead to a trivial solution of an impulse kernel resulting in the same noisy image as output. By carefully designing the appropriate regularizer and selecting the proper step size and learning rate, one may find a sharper image. These parameters are, however, difficult to determine analytically as they heavily depend on the noisy input image itself, and thus do not admit any generalization.

Data-driven methods, on the other hand, make an attempt to determine a non-linear mapping that deblurs the noisy image by learning the appropriate parameter choices particular to an underlying image dataset using deep neural networks (DNN) [238, 239]. Given the training

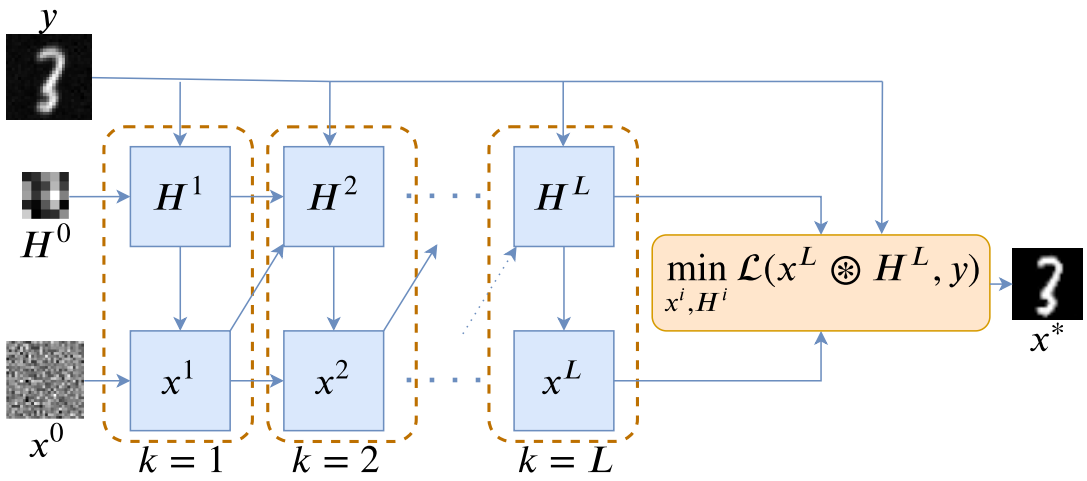


Figure 29. Proposed Deep-URL architecture for model-aware blind deconvolution. Given a blurred image $\boldsymbol{y}$ and initial estimates of the clean image $\boldsymbol{x}^0$ and blurring kernel $\boldsymbol{H}^0$, the model updates $\boldsymbol{x}^k$ and $\boldsymbol{H}^k$ following Algorithm 7.1.

dataset, one can use a DNN either to extract features from the noisy image to estimate the blur kernel [239] or directly learn the mapping to the sharp image [3]. Although these methods achieve substantial performances in certain practical scenarios, they often do not succeed in handling various complex and large-sized blur kernels in blind deconvolution. The structure of the neural networks is usually empirically determined and thus they often lack inherent interpretability. Recent works generate attribution based maps to explain the networks decision, however, they disregard the untapped potential of the model knowledge [240].

In order to enjoy the advantages of both model-based iterative algorithms and data-driven learning strategies, one may exploit the idea of *deep unfolding* [75, 82]. In recent years, deep unfolding networks have gained a significant amount of attention in various branches of signal

processing [36,75,164,241,242]. However, in the context of blind image deconvolution, the extent of deep unfolding capabilities remains largely unexplored. Recently Li *et al.* [2], performed motion deblurring by means of unfolding an iterative algorithm that relies on total-variation (TV) regularization prior in the image gradient domain [243]. Although this approach performs better than the state-of-the-art model-based and data-driven blind deconvolution counterparts, the strict requirement of training a network for a certain dataset makes the algorithm impractical for real-time usage: the algorithm requires a ground truth dataset, to begin with. Additionally, in practice, motion kernels are not pre-deterministic (*e.g.*, in drone image processing), and hence acquiring a labeled dataset is not possible for a supervised learning scenario.

In this paper, we propose a novel technique to unfold an iterative algorithm that estimates the latent clean image and corresponding blur kernel *on the fly* — a zero-shot self-supervised algorithm. In particular, we use the classical Richardson-Lucy blind deconvolution algorithm [244] to construct the network structure and iteratively estimate the clean image and the kernel. We experimentally verify the performance of our algorithm and compare it with [2] and other iterative algorithms and recent neural network approaches.

7.2 Problem Formulation

In this section, we lay the groundwork for our proposed model-aware deep architecture for the problem of blind deconvolution. To this end, we consider an extension of the Richardson-Lucy (RL) algorithm as a baseline to design a deep neural network such that each layer imitates the behavior of one iteration of the RL algorithm.

Generally, the problem of blind deconvolution can be cast as the following optimization problem:

$$\min_{\boldsymbol{x},\boldsymbol{H}} \ \|\boldsymbol{y} - \boldsymbol{H} \circledast \boldsymbol{x}\|_2^2 + \lambda \mathrm{TV}(\boldsymbol{x}), \tag{7.3}$$

where the first term represents the data fidelity term and λ is the regularization coefficient for the total variation (TV) regularization operated on the image $\boldsymbol{x}$. The RL algorithm seeks to recover the sharp image $\boldsymbol{x}$ and the blur kernel $\boldsymbol{H}$ in an iterative manner as described in [244]. Starting from an initial guess for the sharp image and the kernel $(\boldsymbol{x}^0, \boldsymbol{H}^0)$, the update steps for the image and the kernel at the k-th iteration is given by,

$$\boldsymbol{H}^{k+1} = \left(\left[\frac{\boldsymbol{y}}{\boldsymbol{x}^k \circledast \boldsymbol{H}^k} \right] \circledast \boldsymbol{x}^{k\dagger} \right) \odot \boldsymbol{H}^k, \tag{7.4a}$$

$$\boldsymbol{x}^{k+1} = \left(\left[\frac{\boldsymbol{y}}{\boldsymbol{x}^k \circledast \boldsymbol{H}^{k+1}} \right] \circledast \boldsymbol{H}^{k+1\dagger} \right) \odot \boldsymbol{x}^k, \tag{7.4b}$$

where $\odot$ represents the Hadamard product and $(\cdot)^\dagger$ denotes the flipped version of the vector/matrix argument.

7.3 Blind Deconvolution via Deep-URL

In order to obtain a model-aware deep architecture we slightly over parameterize the iterations of RL algorithm (See Eq. (Equation 7.4a)-(Equation 7.4b)) and unfold them onto the layers of a deep neural network. In particular, each layer corresponds to one iteration of the baseline iterative algorithm. Namely, we fix the total computational complexity of the RL algorithm by fixing the total number of iterations as a DNN with L layers. Thus, by substituting the

$\boldsymbol{x}^k$ and $\boldsymbol{H}^k$ in Eq. (Equation 7.4a)-(Equation 7.4b) with trainable parameters, we reformulate each subsequent iterative operation as:

$$\boldsymbol{H}^{k+1} = \sigma\left(\mathrm{ReLU}\left(\left[\frac{\boldsymbol{y}}{\mathrm{ReLU}(\boldsymbol{x}^k \circledast \boldsymbol{W}_{\boldsymbol{H}}^k)}\right] \circledast \boldsymbol{x}^{k\,\dagger}\right) \odot \boldsymbol{W}_{\boldsymbol{H}}^k\right), \tag{7.5a}$$

$$\boldsymbol{x}^{k+1} = \sigma\left(\mathrm{ReLU}\left(\left[\frac{\boldsymbol{y}}{\mathrm{ReLU}(\boldsymbol{W}_{\boldsymbol{x}}^k \circledast \boldsymbol{H}^{k+1})}\right] \circledast \boldsymbol{H}^{k+1\,\dagger}\right) \odot \boldsymbol{W}_{\boldsymbol{x}}^k\right), \tag{7.5b}$$

where $\boldsymbol{W}_{\boldsymbol{x}}^k$ and $\boldsymbol{W}_{\boldsymbol{H}}^k$ are the weights for k-th layer. Furthermore, $\sigma(\cdot)$ represents the *Sigmoid* activation function and ReLU denotes the Rectifier Linear Unit. Note that there exist two implicit constraints on the recovered sharp image and the kernel: (a) both $\boldsymbol{x}$ and $\boldsymbol{H}$ are non-negative and (b) each element of $\boldsymbol{x}$ and $\boldsymbol{H}$ must meet a range constraint.Hence, in order to ensure constraint (a), each convolution operation is activated by a ReLU function, and in addition we use the *Sigmoid* activation after each update step to satisfy constraint (b).

Let $\boldsymbol{\Upsilon}^k = \{\boldsymbol{W}_{\boldsymbol{x}}^k, \boldsymbol{W}_{\boldsymbol{H}}^k\}$ denote the set of trainable parameters of layer k, and $\boldsymbol{\Upsilon} = \boldsymbol{\Upsilon}^1 \cup \boldsymbol{\Upsilon}^2 \cup \cdots \cup \boldsymbol{\Upsilon}^L$. Using the iterative updates from Eq. (Equation 7.5a)-(Equation 7.5b), we formulate the training of our proposed model-aware deep network: *Deep Unfolded Richardson Lucy* (Deep-URL) architecture as follows,

$$\min_{\boldsymbol{\Upsilon}} \mathcal{L}(\boldsymbol{x}^L \circledast \boldsymbol{H}^L, \boldsymbol{y}) + \lambda \mathrm{TV}(\boldsymbol{x}^L) \tag{7.6}$$

where the loss function $\mathcal{L}(\cdot)$ is the negative of the structural similarity index (SSIM) [245] between the true blurred image $\boldsymbol{y}$ and the reconstructed blurred image $\hat{\boldsymbol{y}} = \boldsymbol{x}^L \circledast \boldsymbol{H}^L$.

It is worth mentioning that the proposed deep architecture in conjunction with the proposed learning method manifests itself as a *self-supervised* learning process where the degraded image $\boldsymbol{y}$ is the only information used for estimation of the sharp image $\boldsymbol{x}^*$ and the blurred kernel $\boldsymbol{H}^*$. Figure 29 illustrates the proposed Deep-URL architecture and the training process. Finally, Algorithm 7.1 summarizes the joint optimization process for updating $\boldsymbol{\Upsilon}$. Note that, *once the self-supervised model is optimized for a given blur kernel, the learned weights can be directly used for deblurring any image blurred with the same kernel.*

Algorithm 7.1 DEEP-URL

Input: $\boldsymbol{y}$: blurred image, L: number of layers, N: number of epochs
Output: $\boldsymbol{H}^*$: estimated kernel, $\boldsymbol{x}^*$: sharp image
Initialize: $\boldsymbol{H}^0 \leftarrow \mathcal{U}(0,1)$; $\boldsymbol{x}^0 \leftarrow \mathcal{U}(0,1)$

for $i = 1$ **to** N **do**
  for $k = 0$ **to** $L-1$ **do**
    Compute $\boldsymbol{H}^{k+1}$ using Eq. (Equation 7.5a).
    Compute $\boldsymbol{x}^{k+1}$ using Eq. (Equation 7.5b).
  end for
  Compute the gradients of Eq. (Equation 7.6) *w.r.t.* $\boldsymbol{\Upsilon}$.
  Update $\boldsymbol{\Upsilon}$.
  $\boldsymbol{H}^0 \leftarrow \boldsymbol{H}^L; \boldsymbol{x}^0 \leftarrow \boldsymbol{x}^L$
end for
$\boldsymbol{H}^* = \boldsymbol{H}^L; \boldsymbol{x}^* = \boldsymbol{x}^L$

7.4 Experiments

In this section, we investigate the performance of the proposed Deep-URL framework and compare it with several other state-of-the-art methods in the context of blind deconvolution. First, we compare the performance of Deep-URL with the baseline RL algorithm using the

standard MNIST handwritten digit dataset [246]. Second, we use Levin dataset [1] to compare Deep-URL with existing iterative and deep learning-based blind deconvolution methods proposed in [2, 3, 239].

Optimization setup. The training of Deep-URL (Eq. (Equation 7.6)) is carried out using the RMSprop optimizer for 5000 epochs by employing an adaptive learning rate scheme with an initial learning rate of 0.1 and a decaying factor of 0.1 when reaching 40% and 60% of the total number of epochs. In addition, the TV regularization coefficient λ was set to 0.1 for all experiments. All trainable parameters were initialized using a uniform distribution. We performed a batch-wise optimization, with a batch size of 4, on images blurred using the same kernel for enhancing the performance of Deep-URL.

Evaluation metrics. Inspired by [2], we use the following metrics to evaluate the performance of our proposed method: (1) Structural Similarity Index (SSIM), (2) Peak Signal-to-Noise-Ratio (PSNR), (3) Improvement in Signal-to-Noise-Ratio (ISNR) for the quality of the reconstructed image $\boldsymbol{x}^*$, and (4) Root-Mean-Square Error (RMSE) for comparing the recovered blur kernel $\boldsymbol{H}^*$ with the original $\boldsymbol{H}$. In the sequel, we use the term PSF and blur kernel interchangeably.

MNIST dataset results. For this experiment, we consider the well-known MNIST dataset. We randomly draw 1000 sample images from the MNIST training dataset and use the same motion kernels provided by [1]. Particularly, for each image, we convolve the original image with a randomly chosen aforementioned blur kernel to generate the degraded image. Table Table II demonstrates the performances of the proposed Deep-URL framework with $L \in \{2, 5\}$ layers and the original RL algorithm with the same number of iterations. It is evident from

TABLE II

EVALUATION METRIC SCORES AVERAGED OVER 1000 MNIST IMAGES. ACROSS ALL IMAGE QUALITY METRICS, DEEP-URL (D-URL) OUTPERFORMS THE RL ALGORITHM.

	$L = 2$		$L = 5$	
Metrics	RL	D-URL	RL	D-URL
PSNR(dB)	10.3919	**18.2821**	10.4742	**19.7075**
ISNR (dB)	0.0651	**7.9554**	0.0764	**9.3096**
SSIM	0.4453	**0.7669**	0.4484	**0.8206**
RMSE(×1e-3)	38.54	**4.396**	38.07	**4.399**

Table Table II that the proposed method significantly outperforms the baseline RL algorithm across all evaluation metrics. Interestingly, Deep-URL achieves better performance in terms of both recovering the original image and the PSF even with only $L = 2$ layers—this is presumably due to the hybrid model-based and data-driven nature of the proposed method. Moreover, Deep-URL with $L = 5$ layers attains a very high average ISNR value for the recovered image, which is 121× higher than that of the original RL algorithm. Note that, the RMSE between the original and the reconstructed PSF using the proposed method assumes a 8.55× smaller value than that of the RL algorithm. By comparing the evaluation performance of Deep-URL for $L \in \{2, 5\}$, it is evident that increasing the number of layers result in a much higher increase of scores across all evaluation metrics as compared to the baseline RL algorithm. Finally, from Figure 30, we found that the classical RL algorithm is sensitive to the number of iteration and the performance fluctuates on a random set of 100 MNIST images. However, the performance of Deep-URL always increases as we increase the num-

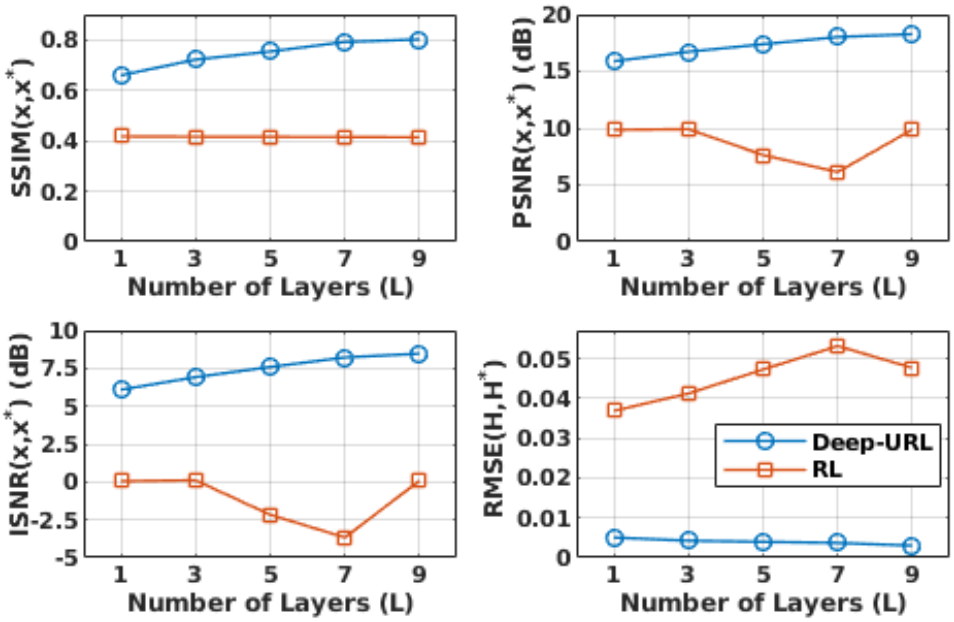
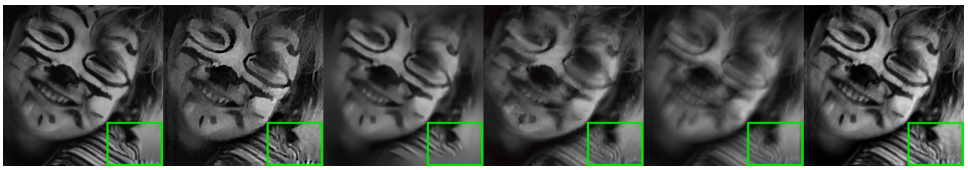


Figure 30. The evaluation metric scores across all image and kernel for different number of layers (L) show that the performance of Deep-URL increases on increasing L as compared to the baseline RL algorithm.

ber of iterations *i.e.*, the number of layers.par **Levin dataset results.** For this experiment, we use the dataset provided by [1] – a widely used benchmark dataset in several deblurring works [2, 234, 235]. It comprises of 4 grayscale images and 8 motion blur kernels: a total of 32 motion blurred images. Table Table III summarizes the performance of Deep-URL in comparison with the baseline RL as well as the methodologies proposed in [239], [3] and [2] on the same dataset. It can be observed from Table Table III that Deep-URL significantly outperforms the baseline RL algorithm across all image and kernel evaluation metrics. In contrast to other methods that include *a priori* learning using training images, Deep-URL is a self-deblurring framework and performs at par (PSNR) or better (ISNR and SSIM) on the image

(a) (b) (c) (d) (e) (f)

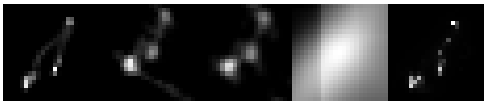


SSIM: 1.0 0.6809 [2] 0.7584 [239] 0.6082 [3] 0.5315 (RL) 0.9189 (Deep-URL)

Figure 31. Qualitative results for a sample image from Levin dataset [1] taken from [2]. The SSIM score between the ground truth image (row 1, a) and the reconstructed images using different iterative and deep learning based blind deconvolution methods (row 1, b-e) shows the superior performance of Deep-URL (f). Comparing the inset images (green boxes in row 1), shows the effectiveness of Deep-URL in retaining fine details of the image. Interestingly, the SSIM of [2] (row 1, b) is low since the image is slightly shifted as it fails to reconstruct the blurring kernel correctly (Figure 32, b)

quality evaluation metrics. Interestingly, an 1.8× increase can be observed in ISNR using Deep-URL with just $L = 5$ when compared to [2]. In regards to the reconstructed blur kernel, it was found that most pixels did not converge to absolute zero and hence a higher RMSE score was obtained in reconstructing the motion kernel blindly. From Figure 31, we observe Deep-URL reconstructs smoother images with lesser artifacts as compared to other state-of-the-art methods.

(a) (b) (c) (d) (e)



Ground Truth [2] [239] RL Deep-URL

Figure 32. Ground truth blur kernel (a) used to blur the image in Figure 31a. Deep-URL reconstructs the kernel with minimum shifts as compared to other iterative and deep learning methods (b-d). Classical RL algorithm completely fails and generates a noisy kernel (d). Note that [3] does not predict the motion blur kernel and hence is not included in the figure above.

TABLE III

DEEP-URL (D-URL) OUTPERFORMS RL ALGORITHM (RAN TILL 5 ITERATIONS) ACROSS ALL IMAGE QUALITY AND RMSE METRICS. IN CONTRAST TO EXISTING DEBLURRING METHODS WHICH LEARN FROM TRAINING IMAGES, DEEP-URL PERFORMS ON PAR (PSNR) AND BETTER (ISNR AND SSIM) IN RECONSTRUCTING THE CLEAN IMAGE.

Metrics	[2]	[3]	[239]	RL	D-URL ($L = 2$)	D-URL ($L = 5$)
PSNR(dB)	27.15	24.51	23.18	19.42	24.85	27.12
ISNR (dB)	3.79	1.35	0.02	-2.98	5.36	**6.95**
SSIM	0.88	0.81	0.81	0.53	0.89	**0.91**
RMSE(×1e-3)	**3.87**	-	-	10.10	8.08	7.10

7.5 Conclusion

In this work, we considered the problem of blind deconvolution and proposed the Deep-URL framework—a model-aware deep blind deconvolution architecture—by unfolding the Richardson-Lucy algorithm (Sec. 7.3). Quantitative and qualitative evaluations (Sec. 7.4) show Deep-URL achieves superior performance than both its baseline RL algorithm and several existing blind deconvolution techniques. In contrast to other MAP-based frameworks, Deep-URL does not show convergence to the trivial solution of an impulse like kernel.

Part II

Performance Guarantees

CHAPTER 8

GUARANTEED DEEP LEARNING FOR RELIABLE RADAR SIGNAL PROCESSING

Overview: R ecently, there has been a significant level of attention paid to the application of deep learning in radar signal processing. Despite its flexibility, deep learning imposes new challenges in guaranteeing the performance of signal processing systems and in establishing trust with regard to their outcome. This represents a critical bottleneck in the application of deep learning in radar signal processing, where having trust in radar inferences is crucial. In this work, we present a novel implicit deep learning model for learning fast and highly-scalable solvers for a general family of optimization problems commonly encountered in the radar signal processing area, encompassing applications in waveform design and target parameter estimation. Specifically, the proposed methodology is based on generalizing the feed-forward neural networks by introducing implicit layers derived from the dynamics of a fixed-point geometric series. Unlike its black-box data-driven counterparts, the implicit and model-based nature of the proposed neural solver significantly reduces the parameters of the network and implementation cost, and most importantly, makes the network amenable to robustness analysis, as well as deriving performance guarantees/bounds on the output of the model. This presents considerable

---

potential for the adoption of our proposed models in radar applications, be it in the classical settings or emerging applications in autonomous vehicles and automotive radar where a large number of reliable radars are projected to be on the road in the near future.

Keywords: Implicit Deep Learning, Inference Guarantees, Model-Based Deep Learning, Radar Signal Processing, Robustness and Reliability.

8.1 Introduction

The significant success of deep learning models in areas such as natural language processing (NLP) [19], life sciences [20], computer vision (CV) [21], and collaborative learning [22], among many others, have surged a significant interest in employing deep learning models for radar signal processing applications. To account for difficulties in the underlying signal processing tasks, most existing deep learning approaches however resort to very large networks whose number of parameters are in the order of millions and billions [23]—making such models data and computing power hungry. Furthermore, these bulky deep learning models further introduce non-ignorable latency during inference which hinders in-time decision making by autonomous agents. More importantly, with all their repertoire of success, the existing data-driven tools typically lack the interpretability and trustability that comes with model-based signal processing. They are particularly prone to be questioned further, or at least not fully trusted by the users, especially in critical applications such as autonomous vehicles. Last but not least, the deterministic deep architectures are generic and it is unclear how to incorporate the existing domain knowledge on the problem in the processing stage. In contrast, many signal processing algorithms (e.g., in the fields of information theory, wireless communications, and radar signal processing) are

backed by decades of theoretical development and research resulting in accurate, meaningful and reliable models. Owing to their theoretical foundations, the model-based signal processing algorithms thus usually come with performance guarantees and bounds allowing for interpreting the output of the model and certifying the achievable performance required for the underlying task.

Despite the above mentioned drawbacks of the generic deep learning models, there has been a few attempts in adopting and re-purposing such generic deep learning models for applications in radar signal processing, showing good performance. To name a few, the authors in [248] consider developing a data-driven methodology for the problem of joint design of transmitted waveform and detector in a radar system, while the authors in [249] have considered the problem of automatic waveform classification in the context of cognitive radar using generic convolutional auto-encoders models. For a detailed treatment of the recent deep learning models for radar signal processing applications, we refer the reader to consult [250], and the references therein.

It has been long apparent that a mere adoption or modification of generic deep neural networks designed for applications such as NLP and CV, and ignoring years of theoretical developments mentioned earlier will result in inefficient networks. This is even more pronounced in the long-standing radar problems with a rich literature. Hence, it is to our belief that one needs to re-think the architectural design of deep neural models than re-purposing them for adoption in critical fields such as radar signal processing for autonomous vehicles.

In this work, we lay the groundwork for our vision in developing interpretable, trustable, and model-driven neural networks, starting from its theoretical foundations, to advance the

state-of-the-art in radar signal processing, particularly for autonomous vehicles. In contrast to generic deep neural networks, which cannot provide performance guarantees due to their black-box nature, our proposed network allows for performing a mathematical analysis of the worst-case performance bound of the model not only during the training of the network but also once the network is trained and is to be used for inference purposes. Last but not least, due to the incorporation of domain-knowledge in the design of the network, the total number of parameters of the network are in the order of the signal dimension and the training can be performed very quickly with far less data samples—allowing for on-the-fly training for real-time applications.

8.2 Data Model and Problem Formulation

As a central task in radar processing, we begin by looking at the problem of receive filter design for a given probing signal. Specifically, our goal is to design a filter to minimize the recovery mean-square-error (MSE) of the scattering coefficient of the target in presence of clutter; see below for more details.

We assume that the complex-valued probing sequence $\boldsymbol{s} \in \mathbb{C}^n$ of length n, that modulates the train of sub-pulses [251], is given by the vector $\boldsymbol{s} = [s_1, s_2, \cdots, s_n]^T$, and that the total energy of the probing signal is fixed and constrained to $\|\boldsymbol{s}\|_2^2 = n$. We note that such power-constraints on $\boldsymbol{s}$ must be always imposed due to practical considerations [252]. Specifically, we focus our attention on the widely used uni-modular probing signals, i.e., $s_i = e^{j\phi_i}$, with $|s_i| = 1$ for $i \in \{1, \cdots, n\}$.

The received digital discrete-time base-band data $\boldsymbol{y} = [y_1, \cdots, y_n]^T$, after pre-processing and range alignment to the range bin of interest is given by [253,254], $y_i = \sum_{0<|k|<n-1} \alpha_k s_{n-k+i} + n_i,\ i \in \{1, \cdots, n\}$, where $s_k = 0$ for $k \notin [1, n]$, the vector of noise elements is assumed to be white, complex-valued, and distributed according to $\boldsymbol{n} \sim \mathcal{CN}(\boldsymbol{0}, \boldsymbol{C} = \sigma^2 \boldsymbol{I})$, $\alpha_0 \in \mathbb{C}$ is the scattering coefficient of the current range bin of interest, and $\{\alpha_i \in \mathbb{C}\}_{i \neq 0}$ represent the scattering coefficients of the adjacent range cells contributing to clutter, as observed at the the current range cell. Furthermore, we assume that the clutter scattering coefficients $\{\alpha_i\}_{i \neq 0}$ are independent of each other and that their power is fixed, i.e., $\mathbb{E}\{|\alpha_i|^2\} = \gamma$, for $i \neq 0$. We note that the scattering coefficients are directly proportional to the radar cross section of the range bins illuminated by the radar system.

Under the considered sytem model, a principal task and challenging problem for a radar signal processing unit is to obtain an estimation of the scattering coefficient α_0 given the acquired samples $\boldsymbol{y}$ in presence of clutter, to find the significantly contributing RCS. A useful methodology in obtaining an estimation of the scattering coefficient of interest α_0 is to use the so-called mismatched filter (MMF) in the receiver side [254]. The deployment of mismatched filter in pulse compression has a great impact in clutter rejection and can be viewed as a linear estimator in which the estimate of α_0 is given by the MMF model $\hat{\alpha}_0 = \boldsymbol{w}^H \boldsymbol{y} / \boldsymbol{w}^H \boldsymbol{s}$, where $\boldsymbol{w} \in \mathbb{C}^n$ is the MMF vector. Under the aforementioned assumptions on the signal model and noise

statistics, the optimal MMF filter $\boldsymbol{w}^\star$ can be formulated as the minimizer of the following objective function:

$$f(\boldsymbol{w};\boldsymbol{s}) = \mathrm{MSE}(\hat{\alpha}_0;\boldsymbol{w},\boldsymbol{s}) = \mathbb{E}\left\{\left|\alpha_0 - \frac{\boldsymbol{w}^H\boldsymbol{y}}{\boldsymbol{w}^H\boldsymbol{s}}\right|^2\right\} = \frac{\boldsymbol{w}^H\boldsymbol{R}\boldsymbol{w}}{|\boldsymbol{w}^H\boldsymbol{s}|^2}, \tag{8.1}$$

where the interference covariance matrix $\boldsymbol{R}$ is given by:

$$\boldsymbol{R} = \gamma \sum_{0<|k|\leq(n-1)} \boldsymbol{J}_k\boldsymbol{s}\boldsymbol{s}^H\boldsymbol{J}_k^H + \boldsymbol{C}, \tag{8.2}$$

and $\{\boldsymbol{J}_k\}$ are the shift matrices (see, e.g., [254] for more discussion on the formation of this interference covariance matrix). Specifically, the minimizer of the objective function $f(\boldsymbol{w};\boldsymbol{s})$ with respect to the MMF filter, follows the well-known closed-form solution [252]:

$$\boldsymbol{w}^\star = \boldsymbol{R}^{-1}\boldsymbol{s} = \underset{\boldsymbol{w}\in\mathbb{C}^n}{\operatorname{argmin}} f(\boldsymbol{w};\boldsymbol{s}), \tag{8.3}$$

where the lower-bound on the performance of the estimator (Equation 8.3) is given by $\mathrm{MSE}\left(\hat{\alpha}_0;\boldsymbol{w}^\star,\boldsymbol{s}\right) = (\boldsymbol{s}^H\boldsymbol{R}^{-1}\boldsymbol{s})^{-1}$.

Although the optimal MMF vector $\boldsymbol{w}^\star = \boldsymbol{R}^{-1}\boldsymbol{s}$ has a good performance in recovering the scattering coefficient in the presence of clutter, obtaining it requires inverting an $n\times n$ matrix $\boldsymbol{R}$, which may be computationally prohibitive in practice for large n, and present a critical computational bottleneck in real-time radar implementation. The inversion is not only computationally expensive, but also requires large data storage capabilities and is highly prone to numerical

errors for ill-conditioned matrices. In the following, we propose a highly-tailored model-based deep architecture based on the Neumann series inversion lemma, where the resulting network allows for 1) controlling the computational complexity of the inference rule, 2) efficiently and quickly finding the optimal MMF vector, and 3) deriving performance bounds on the error of the estimator, upon training the network.

8.3 The Proposed Deep Architecture

In this section, we present our proposed **D**eep **N**eural **M**atrix **I**nversion technique, abbreviated as **DNMI**.

At the heart of our illustrative derivations in the following lies the Neumann power series expansion for matrix inversion [255]. As indicated earlier, in many signal detection and estimation tasks, including our radar problem, one usually encounters matrix inversion, whose computation may be prohibitive in large-scale settings. A similar problem arises, more broadly, in mathematical optimization techniques where a Hessian matrix is to be inverted. In such scenarios, truncated Neumann series (NS) expansions provide a low-cost alternative to approximate the inverted matrices. In the following, we first give a brief introduction to Neumann series for matrix inversion, upon which we derive the architecture of our proposed deep neural network.

Theorem 1. (Neumann Series Theorem) [256]: Denote by $\{\lambda_1, \lambda_2, \cdots, \lambda_n\}$ *the set of eigenvalues of the augmented square matrix* $\bar{\boldsymbol{R}} = (\boldsymbol{I} - \boldsymbol{R})$*. If* $\rho\left(\bar{\boldsymbol{R}}\right) \triangleq \max_i |\lambda_i| < 1$*, the power series* $\bar{\boldsymbol{R}}^0 + \bar{\boldsymbol{R}}^1 + \bar{\boldsymbol{R}}^2 + \cdots$ *then converges to* $\boldsymbol{R}^{-1}$*, i.e., we have* $\boldsymbol{R}^{-1} = \sum_{l=0}^{\infty} (\boldsymbol{I} - \boldsymbol{R})^l$.

The above Neumann series theorem provides a powerful alternative for the exact matrix inversion by considering a truncation of the inversion series. Specifically, in practice, one may

truncate the above expansion to only K-term and use it as an approximation of the exact inversion. However, one can only rely on such an inversion technique if the underlying matrix satisfies the condition $\rho(\boldsymbol{I}-\boldsymbol{R})<1$. In many practical applications, the underlying matrix does not satisfy such a condition, which in turn renders the K-term Neumann series approximation inapplicable. To alleviate this problem, we further propose to augment the NS technique with a preconditioning matrix $\boldsymbol{W}\in\mathbb{C}^{n\times n}$, to obtain the following modified NS for matrix inversion:

$$\boldsymbol{R}^{-1}=(\boldsymbol{W}\boldsymbol{R})^{-1}\boldsymbol{W}=\left(\sum_{l=0}^{\infty}(\boldsymbol{I}-\boldsymbol{W}\boldsymbol{R})^{l}\right)\boldsymbol{W}, \tag{8.4}$$

where the convergence is now ensured if $\rho(\boldsymbol{I}-\boldsymbol{W}\boldsymbol{R})<1$. Note that, even with this augmentation, a judicious design of the pre-conditioning matrix is critical to the convergence of the above series. Constructing such matrices is an active area of research and is indeed a very difficult task [257, 258]. To the best of our knowledge, there exist no general methodology for designing the pre-conditioning matrices that ensure convergence, and also result in an accelerated convergence of the underlying truncated NS. In the following, we present our proposed deep learning model which allows not only for tuning the pre-conditioning matrix in a data-driven manner but also an accelerated and accurate matrix inversion.

In light of the above, we propose to interpret the first K-term truncation of the Neumann series in (Equation 8.4) as a K-layer deep neural network, for which the matrix to be inverted $\boldsymbol{R}$ constitutes the input, and the pre-conditioning matrix the set of trainable parameters, given by $\phi=\{\boldsymbol{W}\in\mathbb{C}^{n\times n}\}$. Accordingly, let $\boldsymbol{G}=\boldsymbol{I}-\boldsymbol{W}\boldsymbol{R}$ and $\mathbb{L}=\{1,\cdots,K-2\}$. Then, the

mathematical operations carried out in the layers of the proposed deep architecture are governed by the relations:

$$\begin{aligned}
&\boldsymbol{g}_0(\boldsymbol{R};\phi) = \boldsymbol{u}_0,\ \boldsymbol{u}_0 = \boldsymbol{I},\\
&\boldsymbol{g}_i(\boldsymbol{R};\phi) = \boldsymbol{G}\boldsymbol{u}_{i-1} + \boldsymbol{g}_{i-1}(\boldsymbol{R};\phi),\ \boldsymbol{u}_i = \boldsymbol{G}\boldsymbol{u}_{i-1},\ \ \forall\, i \in \mathbb{L},\\
&\boldsymbol{g}_{K-1}(\boldsymbol{R};\phi) = \boldsymbol{g}_{K-2}(\boldsymbol{R};\phi)\boldsymbol{W}.
\end{aligned} \tag{8.5}$$

The overall mathematical expression of the proposed neural network can be given as

$$\mathcal{G}_\phi(\boldsymbol{R}) = \boldsymbol{g}_{K-1} \circ \cdots \circ \boldsymbol{g}_0(\boldsymbol{R};\phi). \tag{8.6}$$

It is not difficult to observe that the proposed neural network in (Equation 8.6) is equivalent to performing a K-term truncated version of (Equation 8.4), i.e.,

$$\mathcal{G}_\phi(\boldsymbol{R}) = \boldsymbol{R}_K^{-1}(\boldsymbol{W}) = \left(\sum_{l=0}^{K-1} (\boldsymbol{I} - \boldsymbol{W}\boldsymbol{R})^l\right)\boldsymbol{W} \tag{8.7}$$

yielding a matrix inversion operation with controllable computational cost, whose accuracy depends on the choice of the pre-conditioning matrix $\boldsymbol{W}$ and the total number of terms K. In particular, a judicious design of the pre-conditioning matrix $\boldsymbol{W}$ is expected to result in an accelerated Neumann series that provides higher accuracy while utilizing very few terms, as well as ensuring the convergence of the NS by guaranteeing $\rho(\boldsymbol{I} - \boldsymbol{W}\boldsymbol{R}) < 1$. In fact, the proposed deep architecture provides significant flexibility in learning the pre-conditioning matrix $\boldsymbol{W}$: one

can impose a diagonal structure by defining $\boldsymbol{W} = \text{diag}(w_1, \cdots, w_n)$, a tri-diagonal structure, or a rank-constrained structure via parameterization $\boldsymbol{W} = \boldsymbol{A}\boldsymbol{B}$, where $\boldsymbol{A} \in \mathbb{C}^{n\times m}$ and $\boldsymbol{B} \in \mathbb{C}^{m\times n}$; among many other useful structures depending on the application.

Recall that the optimal MMF vector, for a given covariance matrix $\boldsymbol{R}$ and probing signal $\boldsymbol{s}$ can be expressed as $\boldsymbol{w}^\star(\boldsymbol{R}, \boldsymbol{s}) = \boldsymbol{R}^{-1}\boldsymbol{s}$. The application of the proposed DNMI technique is thus immediate. Instead of using (Equation 8.3), we propose the following approximation of the MMF vector using the DNMI architecture:

$$\boldsymbol{w}_K^\star(\boldsymbol{R}, \boldsymbol{s}; \phi) = \mathcal{G}_\phi(\boldsymbol{R})\boldsymbol{s}, \tag{8.8}$$

where $\boldsymbol{w}_K^\star(\boldsymbol{R}, \boldsymbol{s}; \phi)$ denotes the DNMI-based MMF vector. In the following, we briefly discuss the training stage of the DNMI network.

8.3.1 Training Procedure

The training of the proposed network can be carried out by using stochastic gradient descent optimizers commonly used for deep learning. Specifically, we consider the following scenario for training of the network depending on the available data. We assume the existence of a dataset of size B containing training tuples of the form $\{(\boldsymbol{w}^\star(\boldsymbol{R}^i, \boldsymbol{s}^i), \boldsymbol{R}^i)\}_{i=0}^{B-1}$. Such a dataset can be easily generated in an offline manner, via computing the optimal MMF vector through exact matrix inversion (a one-time cost), and upon training the network, one can employ the optimized DNMI

network for inference purposes through (Equation 8.8). The training is thus can be carried out according to:

$$\min_{\phi \in \mathbb{C}^{n \times n}} \frac{1}{B} \sum_{i} \|\boldsymbol{w}^{\star}(\boldsymbol{R}^{i}, \boldsymbol{s}^{i}) - \mathcal{G}_{\phi}(\boldsymbol{R}^{i})\|_{2}^{2}. \tag{8.9}$$

8.3.2 Performance Guarantees

In contrast to generic deep neural networks, which cannot provide performance guarantees due to their black-box nature, one can perform a mathematical analysis of the worst-case performance bound of the expansion series-based deep networks after the training is completed. As a case in point, one can verify that the accuracy of the K-layer DNMI network is bounded by the $(K+1)$-th power of the spectral norm of the matrix $\boldsymbol{I} - \boldsymbol{W}\boldsymbol{R}$. This provides an upper bound of the error that can guide the training in terms of the number of training epochs, training samples, and number of layers, and once the network is trained, provides an upper bound on the error of the network inference or optimization output—see below.

Let $\boldsymbol{G} = (\boldsymbol{I} - \boldsymbol{W}\boldsymbol{R})$. Then, we define the error vector between the true MMF vector $\boldsymbol{w}^{\star}(\boldsymbol{R}, \boldsymbol{s})$, and the output of the DNMI network as follows:

$$\boldsymbol{e} = \boldsymbol{w}^{\star}(\boldsymbol{R}, \boldsymbol{s}) - \mathcal{G}_{\phi}(\boldsymbol{R})\boldsymbol{s} = \left(\boldsymbol{R}^{-1} - \boldsymbol{R}_{K}^{-1}(\boldsymbol{W})\right)\boldsymbol{s}, \tag{8.10}$$

where we have that

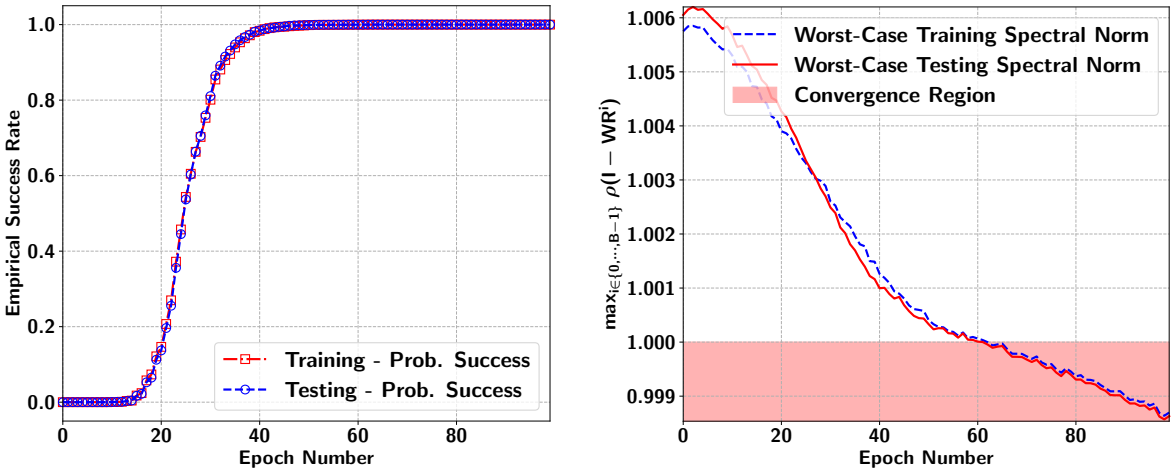
$$\boldsymbol{R}^{-1} - \boldsymbol{R}_K^{-1}(\boldsymbol{W}) = \sum_{l=K}^{\infty} \boldsymbol{G}^l \boldsymbol{W} = \boldsymbol{G}^K \sum_{l=0}^{\infty} \boldsymbol{G}^l \boldsymbol{W} \tag{8.11}$$

$$= \boldsymbol{G}^K \boldsymbol{R}^{-1}. \tag{8.12}$$

Thus, from (Equation 8.10)-(Equation 8.12), we have the following upper-bound on the error:

$$\|\boldsymbol{e}\|_2 = \|\boldsymbol{G}^K \boldsymbol{R}^{-1} \boldsymbol{s}\|_2 \leq \|\boldsymbol{G}^K\| \|\boldsymbol{R}^{-1} \boldsymbol{s}\|_2 \leq \rho^K(\boldsymbol{G}) \|\boldsymbol{R}^{-1} \boldsymbol{s}\|_2, \tag{8.13}$$

where the last inequality is obtained considering that $\|\boldsymbol{G}^K\| \leq \|\boldsymbol{G}\|^K = \rho^K(G)$. Note that such an error bound directly translates to a *measure of closeness to the optimal MSE in the recovery of the target scattering coefficient* α_0. It is clear from (Equation 8.13) that the spectral norm $\rho(\boldsymbol{G})$ provides a certificate for convergence. In addition, once can observe that a judicious design of $\boldsymbol{W}$ can result in the acceleration of the convergence, i.e., having a smaller $\rho(\boldsymbol{G})$. The importance of the above upper-bound is two-fold. First, during the training of the network, one can check the convergence of the network and obtain a measure of success by looking into $\rho(\boldsymbol{G}^i = \boldsymbol{I} - \boldsymbol{W}\boldsymbol{R}^i)$ for training/testing data points. Second, once the network is trained, the obtained $\rho(\boldsymbol{G})$, for a specific covariance matrix, provides an upper-bound on the expected error of the network for the current number of layers or as we introduce more layers. Indeed, once the network is certified for a specific $\boldsymbol{R}$ (meaning $\rho(\boldsymbol{G}) < 1$), one can aim to have $\|\boldsymbol{e}\|_2 \leq \epsilon$, for arbitrary $\epsilon > 0$, by employing more layers, for instance K' layers, of the trained network (without



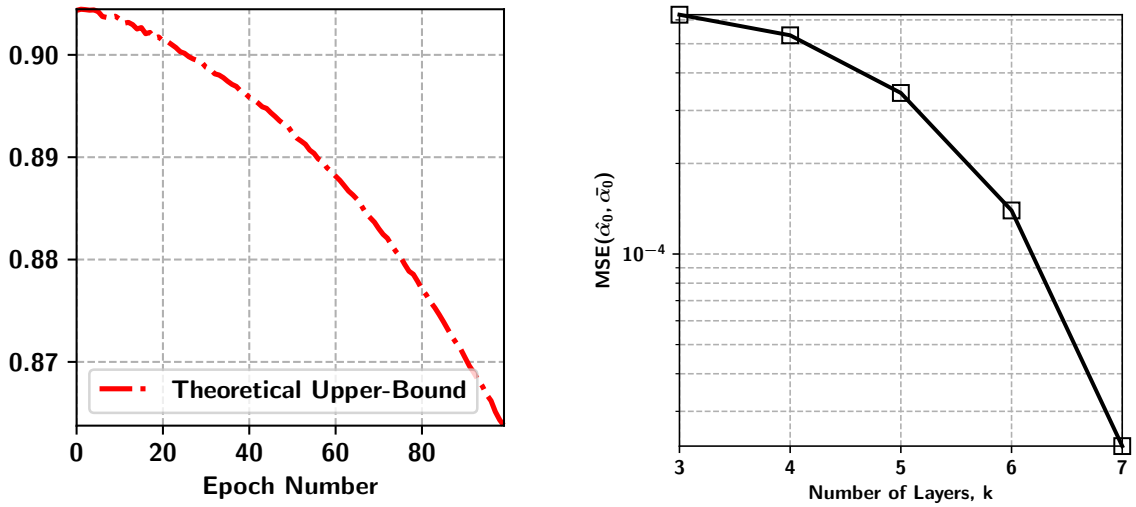
(a) (b)

Figure 33. Examination of training and test success: (a) empirical success rate of training/test data versus training epoch number; (b) worst-Case spectral norm of the training and test data versus epoch number.

re-training) to meet such a bound via simply choosing $K' \geq \left(\ln \epsilon - \ln \left\|\boldsymbol{w}^{\star}(\boldsymbol{R}, \boldsymbol{s})\right\|_2\right) / \ln \rho(\boldsymbol{G})$.

8.4 Numerical Studies and Concluding Remarks

In this part, we investigate the performance of the proposed DNMI network through various numerical studies. We set $\beta = 1$, and $\sigma^2 = 1$, and fix the number of layers of the proposed DNMI to $K = 3$, and set the signal length to $n = 25$. We assume that the phases of the uni-modular probing sequence $\boldsymbol{s}$ is independently and uniformly chosen from the range $[0, 2\pi)$. Accordingly, we generate a training dataset of size $B = 5000$, and evaluate the performance of the network over a testing dataset of the same size. We train the network for a total of



(a) (b)

Figure 34. Numerical study of performance bounds: (a) the theoretical upper bound on the performance of the network versus epoch number; (b) the MSE between the estimated scattering coefficient from the exact MMF and the DNMI-based MMF vector, versus number of layers of the DNMI network.

100 epochs. Moreover, we define the probability of success as the percentage of data-points for which we have $\rho(\boldsymbol{I} - \boldsymbol{W}\boldsymbol{R}^i) < 1$.

Fig. 33(a) presents the empirical success rate of both training and testing points versus epoch number. Furthermore, Fig. 33(b) presents the worst-case training and testing spectral norm versus training epoch. We define the worst-case spectral norm as the maximum $\rho(\boldsymbol{G}^i)$ among the points in testing and training datasets. We note that the pre-conditioning matrix is initialized as $\boldsymbol{W} = \boldsymbol{I}$, and thus at the very first epoch the network boils down to a regular NS operator. On the other hand, it can be deduced from Fig. 33(a) that merely using conventional NS for matrix inversion is not possible (the series diverges) in that none of the data points

satisfy the convergence criterion, as $\rho(\boldsymbol{G}^i) < 1$. This shows the importance of employing an optimized pre-conditioning matrix. However, as the training of the proposed DNMI continues, one can deduce from Fig. 33(b) that for epochs ≥ 60, the proposed methodology can successfully achieve $\rho(\boldsymbol{G}^i) < 1$ for all data-points in both training and testing dataset. Furthermore, the training and testing curves in Fig. 1 closely following each other indicates the highly significant generalization performance of the proposed methodology. This is in contrast to the conventional black-box data-driven methodologies for which the generalization gap is typically large.

Fig. 34(a) demonstrates the theoretical upper-bound on the error obtained in (Equation 8.13) for the proposed DNMI network versus the training epoch. Interestingly, one can observe that the network not only implicitly learns to reduce the theoretical upper-bound (which is a function of the network parameter $\boldsymbol{W}$), but also keeps reducing it even after epochs ≥ 60 where the probability of success and worst-case spectral norm enter the convergence area. This implies that the learned pre-conditioning matrix is indeed resulting in an acceleration of the underlying NS (as the network keeps reducing the upper-bound). This phenomenon is also in accordance with what we observe in Fig. 33(b).

Fig. 34(b) demonstrates the MSE between the estimated scattering coefficient obtained via employing the exact MMF vector and the one obtained using the proposed DNMI network, versus the number of layers of the DNMI network. First, we note the MSE between the two methods is indeed very small, and one can obtain an accurate estimation even with K as low as 3. Second, we observe that as the number of layers increases, the accuracy of the DNMI network increases.

To facilitate fully reliable deep learning for critical applications, we have proposed the model-based DNMI network which is based on the Neumann series expansions—something that has not been considered in the existing literature for advanced radar signal processing algorithms. In contrast to the existing deep learning models, a key advantage of the proposed series expansion-based DNMI network is its inherent ability to provide worst-case performance guarantees.

CHAPTER 9

CONCLUSION

Despite its flexibility, deep learning imposes new challenges in guaranteeing the performance of signal processing systems and in establishing trust in safety and critical decision-making scenarios. we lay the groundwork for our vision in developing interpretable, trustable, and model-driven neural networks, starting from its theoretical foundations, to advance the state-of-the-art in statistical signal processing for various applications including but not limited to communication networks, compressive sensing, phase retrieval, computational imaging, radar signal processing, and autonomous vehicles. In contrast to generic deep neural networks, which cannot provide performance guarantees due to their black-box nature, our proposed methodology allows for performing a mathematical analysis of the worst-case performance bound of the model not only during the training of the network but also once the network is trained and is to be used for inference purposes. Last but not least, due to the incorporation of domain-knowledge in the design of the network, the total number of parameters of the network are in the order of the signal dimension and the training can be performed very quickly with far less data samples—allowing for on-the-fly training for real-time applications, making the proposed methodologies a great candidate for real-time machine learning and signal processing. Namely, we significantly advanced the state of the theory of machine learning and signal processing algorithms through a foundational study of model-based deep learning. Specifically, we established the theoretical guarantees and unveiled the potentials of model-based deep learning for advanced

signal processing schemes. This work is expected to have a significant impact on the theory and practice of advanced hybrid signal processing, computing and machine learning. The impact of the proposed reliable model-based machine learning frameworks is remarkable and can be used in the emerging hardware and software solutions in communication networks, signal processing, autonomous vehicles, and numerous other fields.

9.0.1 Concluding Remarks and Future Directions

In this thesis, we significantly advanced the state of the theory of machine learning and signal processing algorithms through a foundational study of model-based deep learning. Specifically, in Chapter 2 and Chapter 3 we considered the development of a model-based deep architecture that not on is capable of compensating for unknown system parameters in a data-driven manner, but also is able to accelerate the convergence of the underlying inference algorithm while showing significantly better performance than that of its fully model-based and data-driven counterparts. In Chapter 4 and Chapter 5, we visited the problems of one-bit compressive sensing and phase retrieval and developed several model-based deep architectures specifically tailored for the problem at hand. We further showed the potential of the proposed model-based architecture in allowing for learning not only the parameters of the underlying inference rule, but also learning the sensing matrix in a data-driven manner for a specific task at hand. Our results have shown that the resulting architecture outperforms the model-based techniques designed for such problem. Moreover, in Chapter 6, we considered the development of a model-based deep architecture for the field of automotive radar sensing and proposed a novel online-learning strategy of the proposed approach. We further showed that the proposed approach equipped with

the considered online training methodology, is able to quickly adapt to an ever changing environment while outperforming the state-of-the-art algorithms in the respective field. In Chapter 7, we considered the problem of blind deconvolution and developed a sophisticated model-driven deep neural network outperforming the existing methodologies in the field in terms of accuracy, performance, and speed of convergence. Finally, we devoted the second part of this thesis to provide mathematical analysis and performance guarantees on the proposed series expansion-based deep architecture.

In the following, we suggest a few interesting and unexplored future directions:

- The majority of deep unfolding architectures are currently developed based on first-order methods. A fundamental study on the best-case performance of deep unfolded networks using the available mathematical analysis in the field of optimization theory can be of significance interest.
- As explored in Chapter 2, the proposed methodologies and ideas in this thesis can be used to bridge the gap between relaxed optimization processes and their original counterparts. A fundamental study of the advantages and applications of learning surrogate models for bridging such optimality gaps can significantly advance the field.
- As shown in this work, the proposed methodologies in this thesis can be used to not only learn the optimization/inference solutions but also the underlying problem in a structured manner. Hence, a fundamental study of learning parametric optimization models for a given task can significantly advance the state of statistical inference, control theory, economics, and game theory, among others.

APPENDIX

COPYRIGHT PERMISSIONS

In this appendix, we present the copyright permissions for the articles, whose contents were used in this thesis.

LoRD-Net: Unfolded Deep Detection Network With Low-Resolution Receivers

Author: Shahin Khobahi

Publication: IEEE Transactions on Signal Processing

Publisher: IEEE

Date: 2021

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK CLOSE WINDOW

Model-Inspired Deep Detection with Low-Resolution Receivers

Conference Proceedings: 2021 IEEE International Symposium on Information Theory (ISIT)

Author: Shahin Khobahi

Publisher: IEEE

Date: 12 July 2021

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK CLOSE WINDOW

Deep Signal Recovery with One-bit Quantization

Conference Proceedings:
ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)

Author: Shahin Khobahi; Naveed Naimipour; Mojtaba Soltanalian; Yonina C. Eldar

Publisher: IEEE

Date: 12-17 May 2019

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

Deep One-Bit Compressive Autoencoding

Conference Proceedings: 2021 IEEE Statistical Signal Processing Workshop (SSP)

Author: Shahin Khobahi

Publisher: IEEE

Date: 11 July 2021

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK CLOSE WINDOW

UPR: A Model-Driven Architecture for Deep Phase Retrieval

Conference Proceedings: 2020 54th Asilomar Conference on Signals, Systems, and Computers

Author: Naveed Naimipour

Publisher: IEEE

Date: 01 November 2020

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK CLOSE WINDOW

Model-Based Deep Learning for One-Bit Compressive Sensing

Author: Shahin Khobahi; Mojtaba Soltanalian

Publication: IEEE Transactions on Signal Processing

Publisher: IEEE

Date: 2020

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK CLOSE WINDOW

Deep Radar Waveform Design for Efficient Automotive Radar Sensing

Conference Proceedings:
2020 IEEE 11th Sensor Array and Multichannel Signal Processing Workshop (SAM)

Author: Shahin Khobahi; Arindam Bose; Mojtaba Soltanalian

Publisher: IEEE

Date: 8-11 June 2020

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK CLOSE WINDOW

DEEP-URL: A Model-Aware Approach to Blind Deconvolution Based on Deep Unfolded Richardson-Lucy Network

Conference Proceedings: 2020 IEEE International Conference on Image Processing (ICIP)

Author: Chirag Agarwal

Publisher: IEEE

Date: Oct. 2020

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK CLOSE WINDOW

CITED LITERATURE

1. Levin, A., Weiss, Y., Durand, F., and Freeman, W. T.: Understanding and evaluating blind deconvolution algorithms. In 2009 IEEE Conference on Computer Vision and Pattern Recognition, pages 1964–1971, June 2009.

2. Li, Y., Tofighi, M., Monga, V., and Eldar, Y. C.: An algorithm unrolling approach to deep image deblurring. In ICASSP 2019-2019 Proc. IEEE ICASSP, pages 7675–7679. IEEE, 2019.

3. Nah, S., H. Kim, T., and M. Lee, K.: Deep multi-scale convolutional neural network for dynamic scene deblurring. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 3883–3891, 2017.

4. Bishop, C. M.: Pattern recognition and machine learning. springer, 2006.

5. Friedman, J., Hastie, T., Tibshirani, R., et al.: The elements of statistical learning, volume 1. Springer series in statistics New York, 2001.

6. Wasserman, L.: All of statistics: a concise course in statistical inference. Springer Science & Business Media, 2013.

7. LeCun, Y., Bengio, Y., and Hinton, G.: Deep learning. Nature, 521(7553):436, 2015.

8. Shalev-Shwartz, S. and Ben-David, S.: Understanding machine learning: From theory to algorithms. Cambridge university press, 2014.

9. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A.: Automatic differentiation in pytorch. 2017.

10. Baydin, A. G., Pearlmutter, B. A., Radul, A. A., and Siskind, J. M.: Automatic differentiation in machine learning: a survey. Journal of machine learning research, 18, 2018.

11. Bertsekas, D. P.: Nonlinear programming. Journal of the Operational Research Society, 48(3):334–334, 1997.

12. Nocedal, J. and Wright, S.: Numerical optimization. Springer Science & Business Media, 2006.

13. Boyd, S., Boyd, S. P., and Vandenberghe, L.: Convex optimization. Cambridge university press, 2004.

14. Vandenberghe, L. and Boyd, S.: Semidefinite programming. SIAM review, 38(1):49–95, 1996.

15. Parikh, N. and Boyd, S. P.: Proximal algorithms. Foundations and Trends in optimization, 1(3):127–239, 2014.

16. Boyd, S., El Ghaoui, L., Feron, E., and Balakrishnan, V.: Linear matrix inequalities in system and control theory. SIAM, 1994.

17. Lessard, L., Recht, B., and Packard, A.: Analysis and design of optimization algorithms via integral quadratic constraints. SIAM Journal on Optimization, 26(1):57–95, 2016.

18. Sastry, S. and Bodson, M.: Adaptive control: stability, convergence and robustness. Courier Corporation, 2011.

19. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I.: Attention is all you need. Advances in neural information processing systems, 30, 2017.

20. Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., et al.: Highly accurate protein structure prediction with alphafold. Nature, 596(7873):583–589, 2021.

21. Hassaballah, M. and Awad, A. I.: Deep learning in computer vision: principles and applications. CRC Press, 2020.

22. Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., et al.: Mastering chess and shogi by self-play with a general reinforcement learning algorithm. arXiv preprint arXiv:1712.01815, 2017.

23. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language models are few-shot learners. Advances in neural information processing systems, 33:1877–1901, 2020.

24. Khobahi, S., Shlezinger, N., Soltanalian, M., and Eldar, Y. C.: Model-inspired deep detection with low-resolution receivers. In 2021 IEEE International Symposium on Information Theory (ISIT), pages 3349–3354, 2021.

25. Khobahi, S., Shlezinger, N., Soltanalian, M., and Eldar, Y. C.: Lord-net: Unfolded deep detection network with low-resolution receivers. IEEE Transactions on Signal Processing, 69:5651–5664, 2021.

26. Eldar, Y. C.: Sampling theory: Beyond bandlimited systems. Cambridge University Press, 2015.

27. Walden, R. H.: Analog-to-digital converter survey and analysis. 17(4):539–550, 1999.

28. Andrews, J. G., Buzzi, S., Choi, W., Hanly, S. V., Lozano, A., Soong, A. C., and Zhang, J. C.: What will 5G be? 32(6):1065–1082, June 2014.

29. Jeon, Y.-S., Lee, N., Hong, S.-N., and Heath, R. W.: One-bit sphere decoding for uplink massive MIMO systems with one-bit ADCs. 17(7):4509–4521, 2018.

30. Rao, S., Seco-Granados, G., Pirzadeh, H., and Swindlehurst, A. L.: Massive MIMO channel estimation with low-resolution spatial sigma-delta ADCs. arXiv preprint arXiv:2005.07752, 2020.

31. Khobahi, S., Naimipour, N., Soltanalian, M., and Eldar, Y. C.: Deep signal recovery with one-bit quantization. In ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 2987–2991, 2019.

32. Ameri, A., Bose, A., Li, J., and Soltanalian, M.: One-bit radar processing with time-varying sampling thresholds. 67(20):5297–5308, 2019.

33. Jin, B., Zhu, J., Wu, Q., Zhang, Y., and Xu, Z.: One-bit LFMCW radar: spectrum analysis and target detection. 2020.

34. Xi, F., Shlezinger, N., and Eldar, Y. C.: BiLiMO: Bit-limited MIMO radar via task-based quantization. arXiv preprint arXiv:2010.00195, 2020.

35. Xiao, P., Liao, B., and Li, J.: One-bit compressive sensing via Schur-concave function minimization. 67(16):4139–4151, 2019.

36. Khobahi, S. and Soltanalian, M.: Model-based deep learning for one-bit compressive sensing. 68:5292–5307, 2020.

37. Shlezinger, N., Eldar, Y. C., and Rodrigues, M. R.: Hardware-limited task-based quantization. 67(20):5223–5238, 2019.

38. Salamatian, S., Shlezinger, N., Eldar, Y. C., and Médard, M.: Task-based quantization for recovering quadratic functions using principal inertia components. In Proc. IEEE ISIT, 2019.

39. Shlezinger, N. and Eldar, Y. C.: Deep task-based quantization. Entropy, 23(1), 2021.

40. Shlezinger, N., van Sloun, R. J. G., Hujiben, I. A. M., Tsintsadze, G., and Eldar, Y. C.: Learning task-based analog-to-digital conversion for MIMO receivers. In Proc. IEEE ICASSP, 2020.

41. Neuhaus, P., Shlezinger, N., Dörpinghaus, M., Eldar, Y. C., and Fettweis, G.: Task-based analog-to-digital converters. arXiv preprint arXiv:2009.14088, 2020.

42. Gong, T., Shlezinger, N., Ioushua, S. S., Namer, M., Yang, Z., and Eldar, Y. C.: RF chain reduction for MIMO systems: A hardware prototype. 2020.

43. Ioushua, S. S. and Eldar, Y. C.: A family of hybrid analog–digital beamforming methods for massive MIMO systems. 67(12):3243–3257, 2019.

44. Wang, H., Shlezinger, N., Eldar, Y. C., Jin, S., Imani, M. F., Yoo, I., and Smith, D. R.: Dynamic metasurface antennas for MIMO-OFDM receivers with bit-limited ADCs. 2020.

45. Shlezinger, N., Alexandropoulos, G. C., Imani, M. F., Eldar, Y. C., and Smith, D. R.: Dynamic metasurface antennas for 6G extreme massive MIMO communications. 2021.

46. Liu, J., Luo, Z., and Xiong, X.: Low-resolution ADCs for wireless communication: A comprehensive survey. IEEE Access, 7:91291–91324, 2019.

47. Zhang, Y., Alrabeiah, M., and Alkhateeb, A.: Deep learning for massive MIMO with 1-bit ADCs: When more antennas need fewer pilots. 2020.

48. Klautau, A., González-Prelcic, N., Mezghani, A., and Heath, R. W.: Detection and channel equalization with deep learning for low resolution MIMO systems. In 52nd Asilomar Conference on Signals, Systems, and Computers, pages 1836–1840. IEEE, 2018.

49. Balevi, E. and Andrews, J. G.: One-bit OFDM receivers via deep learning. 67(6):4326–4336, 2019.

50. Balevi, E. and Andrews, J. G.: Two-stage learning for uplink channel estimation in one-bit massive MIMO. arXiv preprint arXiv:1911.12461, 2019.

51. Balevi, E. and Andrews, J. G.: Autoencoder-based error correction coding for one-bit quantization. 2020.

52. Kim, D. and Lee, N.: Machine learning based detections for mmwave two-hop MIMO systems using one-bit transceivers. In Proc. IEEE SPAWC, 2019.

53. Nguyen, L. V., Swindlehurst, A. L., and Nguyen, D. H.: SVM-based channel estimation and data detection for one-bit massive MIMO systems. arXiv preprint arXiv:2003.10678, 2020.

54. Nguyen, L. V., Swindlehurst, A. L., and Nguyen, D. H.: Linear and deep neural network-based receivers for massive MIMO systems with one-bit ADCs. arXiv preprint arXiv:2008.03757, 2020.

55. Choi, J., Mo, J., and Heath, R. W.: Near maximum-likelihood detector and channel estimator for uplink multiuser massive mimo systems with one-bit adcs. 64(5):2005–2018, 2016.

56. Risi, C., Persson, D., and Larsson, E. G.: Massive MIMO with 1-bit ADC. arXiv preprint arXiv:1404.7736, 2014.

57. Jacobsson, S., Durisi, G., Coldrey, M., Gustavsson, U., and Studer, C.: One-bit massive MIMO: Channel estimation and high-order modulations. In Proc. IEEE ICCW, pages 1304–1309, 2015.

58. Ivrlac, M. T. and Nossek, J. A.: On MIMO channel estimation with single-bit signal-quantization. In ITG smart antenna workshop, 2007.

59. Mo, J., Schniter, P., Prelcic, N. G., and Heath, R. W.: Channel estimation in millimeter wave mimo systems with one-bit quantization. In 2014 48th Asilomar Conference on Signals, Systems and Computers, pages 957–961. IEEE, 2014.

60. Mezghani, A. and Swindlehurst, A. L.: Blind estimation of sparse broadband massive MIMO channels with ideal and one-bit ADCs. 66(11):2972–2983, 2018.

61. Li, Y., Tao, C., Seco-Granados, G., Mezghani, A., Swindlehurst, A. L., and Liu, L.: Channel estimation and performance analysis of one-bit massive MIMO systems. IEEE Trans. Signal Process, 65(15):4075–4089, 2017.

62. Jacobsson, S., Durisi, G., Coldrey, M., Gustavsson, U., and Studer, C.: Throughput analysis of massive MIMO uplink with low-resolution ADCs. 16(6):4038–4051, 2017.

63. Mo, J., Schniter, P., and Heath, R. W.: Channel estimation in broadband millimeter wave MIMO systems with few-bit ADCs. 66(5):1141–1154, 2017.

64. Farsad, N. and Goldsmith, A.: Detection algorithms for communication systems using deep learning. arXiv preprint arXiv:1705.08044, 2017.

65. Corlay, V., Boutros, J. J., Ciblat, P., and Brunel, L.: Multilevel MIMO detection with deep learning. In 2018 52nd Asilomar Conference on Signals, Systems, and Computers, pages 1805–1809. IEEE, 2018.

66. Liao, Y., Farsad, N., Shlezinger, N., Eldar, Y. C., and Goldsmith, A. J.: Deep neural network symbol detection for millimeter wave communications. arXiv preprint arXiv:1907.11294, 2019.

67. Shlezinger, N., Farsad, N., Eldar, Y. C., and Goldsmith, A. J.: ViterbiNet: A deep learning based Viterbi algorithm for symbol detection. 19(5):3319–3331, 2020.

68. Shlezinger, N., Fu, R., and Eldar, Y. C.: DeepSIC: Deep soft interference cancellation for multiuser MIMO detection. 2020.

69. Shlezinger, N., Farsad, N., Eldar, Y. C., and Goldsmith, A. J.: Data-driven factor graphs for deep symbol detection. In Proc. IEEE ISIT, 2020.

70. He, H., Wen, C.-K., Jin, S., and Li, G. Y.: Model-driven deep learning for joint MIMO channel estimation and signal detection. arXiv preprint arXiv:1907.09439, 2019.

71. Samuel, N., Diskin, T., and Wiesel, A.: Learning to detect. 67(10):2554–2564, 2019.

72. Takabe, S., Imanishi, M., Wadayama, T., Hayakawa, R., and Hayashi, K.: Trainable projected gradient detector for massive overloaded mimo channels: Data-driven tuning approach. IEEE Access, 7:93326–93338, 2019.

73. Balatsoukas-Stimming, A. and Studer, C.: Deep unfolding for communications systems: A survey and some new directions. arXiv preprint arXiv:1906.05774, 2019.

74. Farsad, N., Shlezinger, N., Goldsmith, A. J., and Eldar, Y. C.: Data-driven symbol detection via model-based machine learning. arXiv preprint arXiv:2002.07806, 2020.

75. Hershey, J. R., Roux, J. L., and Weninger, F.: Deep unfolding: Model-based inspiration of novel deep architectures. arXiv preprint arXiv:1409.2574, 2014.

76. Monga, V., Li, Y., and Eldar, Y. C.: Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing. 2020.

77. Naimipour, N., Khobahi, S., and Soltanalian, M.: Unfolded algorithms for deep phase retrieval. arXiv preprint arXiv:2012.11102, 2020.

78. Agarwal, C., Khobahi, S., Bose, A., Soltanalian, M., and Schonfeld, D.: Deep-url: A model-aware approach to blind deconvolution based on deep unfolded richardson-lucy network. In 2020 IEEE International Conference on Image Processing (ICIP), pages 3299–3303. IEEE, 2020.

79. Khobahi, S., Bose, A., and Soltanalian, M.: Deep radar waveform design for efficient automotive radar sensing. In 2020 IEEE 11th Sensor Array and Multichannel Signal Processing Workshop (SAM), pages 1–5. IEEE, 2020.

80. Naimipour, N., Khobahi, S., and Soltanalian, M.: Upr: A model-driven architecture for deep phase retrieval. arXiv preprint arXiv:2003.04396, 2020.

81. Shlezinger, N., Whang, J., Eldar, Y. C., and Dimakis, A. G.: Model-based deep learning. arXiv preprint arXiv:2012.08405, 2020.

82. Gregor, K. and LeCun, Y.: Learning fast approximations of sparse coding. In Proceedings of the 27th International Conference on Machine Learning, pages 399–406. Omnipress, 2010.

83. Flordelis, J., Li, X., Edfors, O., and Tufvesson, F.: Massive MIMO extensions to the COST 2100 channel model: Modeling and validation. 19(1):380–394, 2019.

84. Kingma, D. P. and Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

85. Le, B., Rondeau, T. W., Reed, J. H., and Bostian, C. W.: Analog-to-digital converters. 22(6):69–77, 2005.

86. Kipnis, A., Eldar, Y. C., and Goldsmith, A. J.: Fundamental distortion limits of analog-to-digital compression. 64(9):6013–6033, 2018.

87. Gianelli, C., Xu, L., Li, J., and Stoica, P.: One-bit compressive sampling with time-varying thresholds: Maximum likelihood and the cramér-rao bound. In 2016 50th Asilomar Conference on Signals, Systems and Computers, pages 399–403. IEEE, 2016.

88. Naimipour, N. and Soltanalian, M.: Graph clustering using one-bit comparison data. In 2018 IEEE Asilomar Conference on Signals, Systems and Computers. IEEE, 2018.

89. Khobahi, S. and Soltanalian, M.: Signal recovery from 1-bit quantized noisy samples via adaptive thresholding. In 2018 IEEE Asilomar Conference on Signals, Systems and Computers. IEEE, 2018.

90. Liu, F., Zhu, H., Li, J., Wang, P., and Orlik, P. V.: Massive MIMO channel estimation using signed measurements with antenna-varying thresholds. In 2018 IEEE Statistical Signal Processing Workshop (SSP), pages 188–192. IEEE, 2018.

91. He, K., Zhang, X., Ren, S., and Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In Proceedings of the IEEE international conference on computer vision, pages 1026–1034, 2015.

92. Deng, L., Yu, D., et al.: Deep learning: methods and applications. Foundations and Trends® in Signal Processing, 7(3–4):197–387, 2014.

93. He, H., Wen, C.-K., Jin, S., and Li, G. Y.: A model-driven deep learning network for MIMO detection. arXiv preprint arXiv:1809.09336, 2018.

94. Samuel, N., Diskin, T., and Wiesel, A.: Learning to detect. arXiv preprint arXiv:1805.07631, 2018.

95. Wisdom, S., Hershey, J., Le Roux, J., and Watanabe, S.: Deep unfolding for multichannel source separation. In 2016 Proc. IEEE ICASSP, pages 121–125. IEEE, 2016.

96. Borgerding, M. and Schniter, P.: Onsager-corrected deep learning for sparse linear inverse problems. In 2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP), pages 227–231. IEEE, 2016.

97. Mezghani, A., Antreich, F., and Nossek, J. A.: Multiple parameter estimation with quantized channel output. In 2010 International ITG Workshop on Smart Antennas (WSA), pages 143–150. IEEE, 2010.

98. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al.: Tensorflow: a system for large-scale machine learning. In OSDI, volume 16, pages 265–283, 2016.

99. Khobahi, S. and Soltanalian, M.: Model-based deep learning for one-bit compressive sensing. IEEE Transactions on Signal Processing, 68:5292–5307, 2020.

100. Khobahi, S., Bose, A., and Soltanalian, M.: Deep one-bit compressive autoencoding. In 2021 IEEE Statistical Signal Processing Workshop (SSP), pages 371–375, 2021.

101. Eldar, Y. C. and Kutyniok, G.: Compressed sensing: theory and applications. Cambridge university press, 2012.

102. Boufounos, P. T. and Baraniuk, R. G.: 1-bit compressive sensing. In 2008 42nd Annual Conference on Information Sciences and Systems, pages 16–21, March 2008.

103. Laska, J. N., Wen, Z., Yin, W., and Baraniuk, R. G.: Trust, but verify: Fast and accurate signal recovery from 1-bit compressive measurements. 59(11):5289–5301, Nov 2011.

104. Jacques, L., Laska, J. N., Boufounos, P. T., and Baraniuk, R. G.: Robust 1-bit compressive sensing via binary stable embeddings of sparse vectors. 59(4):2082–2102, April 2013.

105. Movahed, A., Panahi, A., and Durisi, G.: A robust RFPI-based 1-bit compressive sensing reconstruction algorithm. In 2012 IEEE Information Theory Workshop, pages 567–571, Sep. 2012.

106. Yan, M., Yang, Y., and Osher, S.: Robust 1-bit compressive sensing using adaptive outlier pursuit. 60(7):3868–3875, July 2012.

107. Bose, A., Ameri, A., and Soltanalian, M.: Waveform design for one-bit radar systems under uncertain interference statistics. In 2019 53rd Asilomar Conference on Signals, Systems, and Computers, pages 1167–1171, 2019.

108. Xiao, P., Liao, B., and Li, J.: One-bit compressive sensing via Schur-concave function minimization. IEEE Transactions on Signal Processing, 67(16):4139–4151, Aug 2019.

109. Knudson, K., Saab, R., and Ward, R.: One-bit compressive sensing with norm estimation. 62(5):2748–2758, 2016.

110. Candès, E. J. and Wakin, M. B.: An introduction to compressive sampling. 25(2):21–30, 2008.

111. Bandeira, A. S., Dobriban, E., Mixon, D. G., and Sawin, W. F.: Certifying the restricted isometry property is hard. IEEE transactions on information theory, 59(6):3448–3450, 2013.

112. Candès, E. J., Romberg, J., and Tao, T.: Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. IEEE Transactions on information theory, 52(2):489–509, 2006.

113. Candès, E. J. and Tao, T.: Decoding by linear programming. 51(12):4203–4215, 2005.

114. Foucart, S. and Rauhut, H.: A mathematical introduction to compressive sensing. Bull. Am. Math, 54:151–165, 2017.

115. Elad, M.: Sparse and redundant representations: from theory to applications in signal and image processing. New York, NY, Springer, 2010.

116. Elad, M.: Optimized projections for compressed sensing. IEEE Transactions on Signal Processing, 55(12):5695–5702, 2007.

117. Strohmer, T. and Heath, R. W.: Grassmannian frames with applications to coding and communication. Applied and Computational Harmonic Analysis, 14(3):257–275, 2003.

118. Eldar, Y. C. and Forney, G. D.: Optimal tight frames and quantum measurement. IEEE Transactions on Information Theory, 48(3):599–610, 2002.

119. Drineas, P., Magdon-Ismail, M., Mahoney, M. W., and Woodruff, D. P.: Fast approximation of matrix coherence and statistical leverage. The Journal of Machine Learning Research, 13(1):3475–3506, 2012.

120. Hu, H., Soltanalian, M., Stoica, P., and Zhu, X.: Locating the few: Sparsity-aware waveform design for active radar. IEEE Transactions on Signal Processing, 65(3):651–662, 2016.

121. Li, Z., Xu, W., Zhang, X., and Lin, J.: A survey on one-bit compressed sensing: theory and applications. Frontiers of Computer Science, 12(2):217–230, Apr 2018.

122. Plan, Y. and Vershynin, R.: Robust 1-bit compressed sensing and sparse logistic regression: A convex programming approach. IEEE Transactions on Information Theory, 59(1):482–494, 2012.

123. Kamilov, U. S., Bourquard, A., Amini, A., and Unser, M.: One-bit measurements with adaptive thresholds. IEEE Signal Processing Letters, 19(10):607–610, 2012.

124. Kulkarni, K., Lohit, S., Turaga, P., Kerviche, R., and Ashok, A.: Reconnet: Non-iterative reconstruction of images from compressively sensed measurements. arXiv preprint arXiv:1601.06892, 2016.

125. Iliadis, M., Spinoulas, L., and Katsaggelos, A. K.: Deep fully-connected networks for video compressive sensing. Digital Signal Processing, 72:9 – 18, 2018.

126. Mousavi, A., Patel, A. B., and Baraniuk, R. G.: A deep learning approach to structured signal recovery. In 2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton), pages 1336–1343, Sep. 2015.

127. Shlezinger, N., Eldar, Y. C., and Rodrigues, M. R.: Hardware-limited task-based quantization. IEEE Transactions on Signal Processing, 67(20):5223–5238, 2019.

128. Wu, Y., Rosca, M., and Lillicrap, T.: Deep compressed sensing. arXiv preprint arXiv:1905.06723, 2019.

129. Chien, J.-T. and Lee, C.-H.: Deep unfolding for topic models. IEEE transactions on pattern analysis and machine intelligence, 40(2):318–331, 2017.

130. Khobahi, S., Naimipour, N., Soltanalian, M., and Eldar, Y. C.: Deep signal recovery with one-bit quantization. In Proc. IEEE ICASSP, pages 2987–2991. IEEE, 2019.

131. Solomon, O., Cohen, R., Zhang, Y., Yang, Y., He, Q., Luo, J., van Sloun, R. J., and Eldar, Y. C.: Deep unfolded robust pca with application to clutter suppression in ultrasound. IEEE transactions on medical imaging, 2019.

132. Plan, Y. and Vershynin, R.: One-bit compressed sensing by linear programming. Communications on Pure and Applied Mathematics, 66(8):1275–1297, 2013.

133. Shen, Y., Fang, J., Li, H., and Chen, Z.: A one-bit reweighted iterative algorithm for sparse signal recovery. In 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, pages 5915–5919, May 2013.

134. Blumensath, T. and Davies, M. E.: Iterative hard thresholding for compressed sensing. Applied and computational harmonic analysis, 27(3):265–274, 2009.

135. Naimipour, N., Khobahi, S., and Soltanalian, M.: Unfolded algorithms for deep phase retrieval. arXiv preprint arXiv:2012.11102, 2020.

136. Naimipour, N., Khobahi, S., and Soltanalian, M.: Upr: A model-driven architecture for deep phase retrieval. In 2020 54th Asilomar Conference on Signals, Systems, and Computers, pages 205–209, 2020.

137. Millane, R. P.: Phase retrieval in crystallography and optics. J. Opt. Soc. Am. A, 7(3):394–411, Mar 1990.

138. Kim, W. and Hayes, M. H.: The phase retrieval problem in x-ray crystallography. In Proceedings of the Acoustics, Speech, and Signal Processing, 1991. ICASSP-91., 1991 International Conference, ICASSP '91, page 1765–1768, USA, 1991. IEEE Computer Society.

139. Fienup, J. R., Marron, J. C., Schulz, T. J., and Seldin, J. H.: Hubble space telescope characterized by using phase-retrieval algorithms. Appl. Opt., 32(10):1747–1767, Apr 1993.

140. Krist, J. E. and Burrows, C. J.: Phase-retrieval analysis of pre- and post-repair hubble space telescope images. Appl. Opt., 34(22):4951–4964, Aug 1995.

141. Sarnik, A. M.: Phase Retrieval: A Practical Application For The Space Telescope. In Inverse Optics II, eds. R. Bates and A. J. Devaney, volume 0558, pages 85 – 94. International Society for Optics and Photonics, SPIE, 1985.

142. Gerchberg, R. W. and Saxton, W. O.: Phase determination for image and diffraction plane pictures in the electron microscope. Optik, 34:275–284, 1971.

143. Gerchberg, R. W. and Saxton, W. O.: A practical algorithm for the determination of phase from image and diffraction plane pictures. Optik, 35:227–246, 1972.

144. Fienup, J. R.: Comments on "the reconstruction of a multidimensional sequence from the phase or magnitude of its Fourier transform". IEEE Transactions on Acoustics, Speech, and Signal Processing, 31(3):738–739, Jun 1983.

145. Fienup, J. R.: Phase retrieval algorithms: a comparison. Appl. Opt., 21(15):2758–2769, Aug 1982.

146. Fienup, J. R.: Reconstruction of an object from the modulus of its Fourier transform. Opt. Lett., 3(1):27–29, Jul 1978.

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

148. Candès, E. and Li, X.: Solving quadratic equations via PhaseLift when there are about as many equations as unknowns. Foundations of Computational Mathematics, 14(5):1017–1026, 2014.

149. Jaganathan, K., Oymak, S., and Hassibi, B.: Sparse phase retrieval: Convex algorithms and limitations. In 2013 IEEE International Symposium on Information Theory, pages 1022–1026. IEEE, 2013.

150. Liang, J., Stoica, P., Jing, Y., and Li, J.: Phase retrieval via the alternating direction method of multipliers. IEEE Signal Processing Letters, 25(1):5–9, 2017.

151. Qiu, T., Babu, P., and Palomar, D. P.: Prime: Phase retrieval via majorization-minimization. IEEE Transactions on Signal Processing, 64(19):5174–5186, 2016.

152. Candès, E. J., Li, X., and Soltanolkotabi, M.: Phase retrieval via Wirtinger flow: Theory and algorithms. IEEE Transactions on Information Theory, 61(4):1985–2007, Apr 2015.

153. Chen, Y. and Candes, E. J.: Solving random quadratic systems of equations is nearly as easy as solving linear systems. In Advances in Neural Information Processing Systems 28, pages 739–747. Curran Associates, Inc., 2015.

154. Kolte, R. and Özgür, A.: Phase retrieval via incremental truncated wirtinger flow. ArXiv, abs/1606.03196, 2016.

155. Zhang, H., Liang, Y., and Chi, Y.: A nonconvex approach for phase retrieval: Reshaped Wirtinger flow and incremental algorithms. Journal of Machine Learning Research, 18(141):1–35, 2017.

156. Jaganathan, K., Eldar, Y., and Hassibi, B.: Phase retrieval: An overview of recent developments. Oct 2015.

157. Wang, G., Giannakis, G. B., Chen, J., and Akçakaya, M.: SPARTA: Sparse phase retrieval via truncated amplitude flow. In 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 3974–3978, 2017.

158. Qian, C., Fu, X., Sidiropoulos, N. D., Huang, L., and Xie, J.: Inexact alternating optimization for phase retrieval in the presence of outliers. IEEE Transactions on Signal Processing, 65(22):6069–6082, 2017.

159. Kim, K. and Chung, S.: Fourier phase retrieval with extended support estimation via deep neural network. IEEE Signal Processing Letters, 26(10):1506–1510, Oct 2019.

160. Işil, c., Oktem, F. S., and Koç, A.: Deep iterative reconstruction for phase retrieval. Appl. Opt., 58(20):5422–5431, Jul 2019.

161. Metzler, C., Schniter, P., Veeraraghavan, A., and Baraniuk, R. G.: prDeep: Robust phase retrieval with a flexible deep network. In Proceedings of the 35th International Conference on Machine Learning, volume 80, pages 3501–3510. PMLR, 2018.

162. Paine, S. W. and Fienup, J. R.: Smart starting guesses from machine learning for phase retrieval. In Space Telescopes and Instrumentation 2018: Optical, Infrared, and Millimeter Wave, volume 10698, pages 1681 – 1687. International Society for Optics and Photonics, SPIE, 2018.

163. LeCun, Y., Bengio, Y., and Hinton, G.: Deep learning. Nature, 521:436–44, May 2015.

164. Bertocchi, C., Chouzenoux, E., Corbineau, M.-C., Pesquet, J.-C., and Prato, M.: Deep unfolding of a proximal interior point method for image restoration. Inverse Problems, 2019.

165. Candes, E. J. and Tao, T.: Near-optimal signal recovery from random projections: Universal encoding strategies? IEEE Transactions on Information Theory, 52(12):5406–5425, 2006.

166. Candes, E. J., Romberg, J., and Tao, T.: Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. IEEE Transactions on Information Theory, 52(2):489–509, 2006.

167. Candès, E. J., Eldar, Y. C., Strohmer, T., and Voroninski, V.: Phase retrieval via matrix completion. SIAM Review, 57(2):225–251, Jan 2015.

168. Candès, E. J. and Li, X.: Solving quadratic equations via phaselift when there are about as many equations as unknowns. Foundations of Computational Mathematics, 14(5):1017–1026, Jun 2013.

169. Li, S. and Ge, G.: Deterministic sensing matrices arising from near orthogonal systems. IEEE Transactions on Information Theory, 60(4):2291–2302, 2014.

170. Frosio, I. and Borghese, N. A.: Statistical based impulsive noise removal in digital radiography. IEEE Transactions on Medical Imaging, 28(1):3–16, 2009.

171. Fienup, J. R.: Phase-retrieval algorithms for a complicated optical system. Applied optics, 32 10:1737–46, 1993.

172. Haupt, J., Nowak, R., and Castro, R.: Adaptive sensing for sparse signal recovery. In 2009 IEEE 13th Digital Signal Processing Workshop and 5th IEEE Signal Processing Education Workshop, pages 702–707, 2009.

173. Blumensath, T. and Davies, M.: Iterative hard thresholding for compressed sensing. Applied and Computational Harmonic Analysis, 27:265–274, Dec 2009.

174. Sharanabasaveshwara, H. B. and Herur, S.: Designing of sensing matrix for compressive sensing and reconstruction. In 2018 Second International Conference on Advances in Electronics, Computers and Communications (ICAECC), pages 1–5, 2018.

175. Dong, C., Loy, C. C., He, K., and Tang, X.: Learning a deep convolutional network for image super-resolution. In Computer Vision – ECCV 2014, eds. D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, pages 184–199, Cham, 2014. Springer International Publishing.

176. Shi, W., Jiang, F., Liu, S., and Zhao, D.: Image compressed sensing using convolutional neural network. IEEE Transactions on Image Processing, 29:375–388, 2020.

177. Palangi, H., Ward, R., and Deng, L.: Distributed compressive sensing: A deep learning approach. IEEE Transactions on Signal Processing, 64(17):4504–4518, 2016.

178. Mousavi, A., Patel, A. B., and Baraniuk, R. G.: A deep learning approach to structured signal recovery. In 2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton), pages 1336–1343, 2015.

179. Shlezinger, N., Dabora, R., and Eldar, Y. C.: Measurement matrix design for phase retrieval based on mutual information. IEEE Transactions on Signal Processing, 66(2):324–339, 2018.

180. Sattar, Y. and Oymak, S.: Quickly finding the best linear model in high dimensions via projected gradient descent. IEEE Transactions on Signal Processing, 68:818–829, 2020.

181. Bahmani, S. and Raj, B.: A unifying analysis of projected gradient descent for l_p-constrained least squares. Applied and Computational Harmonic Analysis, 34(3):366–378, 2013.

182. Soltanolkotabi, M.: Learning ReLUs via gradient descent. In Advances in neural information processing systems (NIPS), pages 2007–2017, 2017.

183. Lohit, S., Liu, D., Mansour, H., and Boufounos, P. T.: Unrolled projected gradient descent for multi-spectral image fusion. In IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 7725–7729. IEEE, 2019.

184. Giryes, R., Eldar, Y. C., Bronstein, A. M., and Sapiro, G.: The learned inexact project gradient descent algorithm. In IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 6767–6771. IEEE, 2018.

185. Chiang, P., Geiping, J., Goldblum, M., Goldstein, T., Ni, R., Reich, S., and Shafahi, A.: WITCHcraft: Efficient PGD attacks with random step size. In IEEE International

Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 3747–3751. IEEE, 2020.

186. Weller, D. S.: Robust phase retrieval with sparsity under nonnegativity constraints. In 50th Asilomar Conference on Signals, Systems and Computers, pages 1043–1047. IEEE, 2016.

187. Mukherjee, S. and Seelamantula, C. S.: Quantization-aware phase retrieval. International Journal of Wavelets, Multiresolution and Information Processing, page 2040006, 2020.

188. Qiao, H. and Pal, P.: Sparse phase retrieval using partial nested fourier samplers. In IEEE Global Conference on Signal and Information Processing (GlobalSIP), pages 522–526. IEEE, 2015.

189. Khobahi, S., Bose, A., and Soltanalian, M.: Deep radar waveform design for efficient automotive radar sensing. In 2020 IEEE 11th Sensor Array and Multichannel Signal Processing Workshop (SAM), pages 1–5, 2020.

190. Rohling, H. and Kronauge, M.: Continuous waveforms for automotive radar systems, pages 173–205. Radar, Sonar & Navigation. Institution of Engineering and Technology, 2012.

191. Naghsh, M. M., Soltanalian, M., and Modarres-Hashemi, M.: Radar code design for detection of moving targets. 50(4):2762–2778, 2014.

192. Gini, F., De Maio, A., and Patton, L.: Waveform design and diversity for advanced radar systems, volume 22. London: Institution of Engineering and Technology, 2012.

193. Stoica, P., Li, J., and Xue, M.: Transmit codes and receive filters for radar. IEEE Signal Processing Magazine, 25(6):94–109, November 2008.

194. Li, J. and Stoica, P.: MIMO Radar Signal Processing. Wiley, first edition, 2008.

195. Skolnik, M.: Radar Handbook. New York, McGraw-Hill, third edition, 2008.

196. Aubry, A., De Maio, A., Piezzo, M., Farina, A., and Wicks, M.: Cognitive design of the receive filter and transmitted phase code in reverberating environment. IET Journal on Radar, Sonar, and Navigation, 6(9):822–833, 2012.

197. Aubry, A., De Maio, A., Farina, A., and Wicks, M.: Knowledge-aided (potentially cognitive) transmit signal and receive filter design in signal-dependent clutter. 49:93–117, January 2013.

198. Naghsh, M. M., Modarres-Hashemi, M., Shahbazpanahi, S., Soltanalian, M., and Stoica, P.: Unified optimization framework for multi-static radar code design using information-theoretic criteria. 61(21):5401–5416, November 2013.

199. Naghsh, M. M., Soltanalian, M., Stoica, P., Modarres-Hashemi, M., De Maio, A., and Aubry, A.: A Doppler robust design of transmit sequence and receive filter in the presence of signal-dependent interference. 62(4):772–785, February 2014.

200. J.Li, Xu, L., Stoica, P., Forsythe, K., and Bliss, D.: Range compression and waveform optimization for MIMO radar: A Cramer-Rao bound based study. 56(1):218–232, 2008.

201. Xu, L. and Li, J.: Iterative generalized-likelihhod ratio test for MIMO radar. 55:2375–2385, 2007.

202. Hu, H., Soltanalian, M., Stoica, P., and Zhu, X.: Locating the few: Sparsity-aware waveform design for active radar. 65(3):651–662, 2017.

203. Bell, M. R.: Information theory and radar waveform design. 39(5):1578–1597, 1993.

204. Friedlander, B.: Waveform design for MIMO radars. 43:1227–1238, July 2007.

205. De Maio, A., Huang, Y., Piezzo, M., Zhang, S., and Farina, A.: Design of optimized radar codes with a peak to average power ratio constraint. 59:2683–2697, June 2011.

206. Blum, R. S. and Yang, Y.: MIMO radar waveform design based on mutual information and minimum mean-square error estimation. 43:330–343, January 2007.

207. Xu, L., Li, J., and Stoica, P.: Target detection and parameter estimation for MIMO radar systems. 44(3):927–939, 2008.

208. Delong, D. F. and Hofstteter, E. M.: On the design of optimum radar waveforms for clutter rejection. 13:454–463, July 1967.

209. Spafford, L. J.: Optimum radar signal processing in clutter. 14:734–743, September 1968.

210. Soltanalian, M. and Stoica, P.: Computational design of sequences with good correlation properties. 60(5):2180–2193, 2012.

211. Tang, B., Tang, J., and Peng, Y.: MIMO radar waveform design in colored noise based on information theory. 58:4684–4697, September 2010.

212. Song, X., Willett, P., Zhou, S., and Luh, P.: The MIMO radar and jammer games. 60(2):687 –699, February 2012.

213. Kay, S. M.: Waveform design for multistatic radar detection. 45:1153–1165, July 2009.

214. Soltanalian, M. and Stoica, P.: Designing unimodular codes via quadratic optimization. 62(5):1221–1234, March 2014.

215. Soltanalian, M., Tang, B., Li, J., and Stoica, P.: Joint design of the receive filter and transmit sequence for active sensing. IEEE Signal Processing Letters, 20(5):423–426, May 2013.

216. Bose, A. and Soltanalian, M.: Constructing binary sequences with good correlation properties: An efficient analytical-computational interplay. 66(11):2998–3007, 2018.

217. Bose, A., Khobahi, S., and Soltanalian, M.: Joint optimization of waveform covariance matrix and antenna selection for MIMO radar. arXiv preprint arXiv:1910.07591, 2019.

218. Khobahi, S. and Soltanalian, M.: Signal recovery from 1-bit quantized noisy samples via adaptive thresholding. In 2018 52nd Asilomar Conference on Signals, Systems, and Computers, pages 1757–1761, Oct 2018.

219. Bose, A., Ameri, A., and Soltanalian, M.: Waveform design for one-bit radar systems under uncertain interference statistics. arXiv preprint arXiv:1912.00556, 2019.

220. Rohling, H. and Meinecke, M.: Waveform design principles for automotive radar systems. In 2001 CIE International Conference on Radar Proceedings (Cat No.01TH8559), pages 1–4, October 2001.

221. Rohling, H. and Moller, C.: Radar waveform for automotive radar systems and applications. In 2008 IEEE Radar Conference, pages 1–4, May 2008.

222. Kronauge, M. and Rohling, H.: New chirp sequence radar waveform. IEEE Transactions on Aerospace and Electronic Systems, 50(4):2870–2877, October 2014.

223. Mason, E., Yonel, B., and Yazici, B.: Deep learning for radar. In 2017 IEEE Radar Conference (RadarConf), pages 1703–1708, May 2017.

224. Hershey, J. R., Roux, J. L., and Weninger, F.: Deep unfolding: Model-based inspiration of novel deep architectures. ArXiv, abs/1409.2574, 2014.

225. Dinkelbach, W.: On nonlinear fractional programming. Management Science, 13(7):492–498, 1967.

226. Tofallis, C.: Fractional programming: Theory, methods and applications. Journal of the Operational Research Society, 49(8):895–895, 1998.

227. Soltanalian, M., Gharanjik, A., Shankar, M. R. B., and Oftersten, B.: Grab-n-pull: An optimization framework for fairness-achieving networks. In 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 3301–3305, March 2016.

228. Khobahi, S., Soltanalian, M., Jiang, F., and Swindlehurst, A. L.: Optimized transmission for parameter estimation in wireless sensor networks. IEEE Transactions on Signal and Information Processing over Networks, pages 1–1, 2019.

229. Stoica, P., He, H., and Li, J.: New algorithms for designing unimodular sequences with good correlation properties. IEEE Transactions on Signal Processing, 57(4):1415–1425, April 2009.

230. Agarwal, C., Khobahi, S., Bose, A., Soltanalian, M., and Schonfeld, D.: Deep-url: A model-aware approach to blind deconvolution based on deep unfolded richardson-lucy network. In 2020 IEEE International Conference on Image Processing (ICIP), pages 3299–3303, 2020.

231. Lai, W. S., Huang, J. B., Hu, Z., Ahuja, N., and Yang, M. H.: A comparative study for single image blind deblurring. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1701–1709, 2016.

232. Fergus, R., Singh, B., Hertzmann, A., Roweis, S. T., and Freeman, W. T.: Removing camera shake from a single photograph. In ACM SIGGRAPH 2006 Papers, SIG-

GRAPH '06, pages 787–794, New York, NY, USA, 2006. Association for Computing Machinery.

233. Levin, A.: Blind motion deblurring using image statistics. In Proceedings of the 19th International Conference on Neural Information Processing Systems, pages 841–848, Cambridge, MA, USA, 2006. MIT Press.

234. Xu, L. and Jia, J.: Two-phase kernel estimation for robust motion deblurring. In Computer Vision – ECCV 2010, eds. K. Daniilidis, P. Maragos, and N. Paragios, pages 157–170. Springer Berlin Heidelberg, 2010.

235. Krishnan, D., Tay, T., and Fergus, R.: Blind deconvolution using a normalized sparsity measure. In CVPR 2011, pages 233–240, June 2011.

236. Xu, L., Zheng, S., and Jia, J.: Unnatural l_0 sparse representation for natural image deblurring. In 2013 IEEE Conference on Computer Vision and Pattern Recognition, pages 1107–1114, June 2013.

237. Pan, J., Sun, D., Pfister, H., and Yang, M.: Deblurring images via dark channel prior. IEEE Transactions on Pattern Analysis and Machine Intelligence, 40(10):2315–2328, October 2018.

238. Xu, X., Pan, J., Zhang, Y., and Yang, M.: Motion blur kernel estimation via deep learning. IEEE Transactions on Image Processing, 27(1):194–205, January 2018.

239. Chakrabarti, A.: A neural approach to blind motion deblurring. In Computer Vision – ECCV 2016, eds. B. Leibe, J. Matas, N. Sebe, and M. Welling, pages 221–235, Cham, 2016. Springer International Publishing.

240. Agarwal, C., Schonfeld, D., and Nguyen, A.: Removing input features via a generative model to explain their attributions to classifier's decisions. arXiv preprint arXiv:1910.04256, 2019.

241. Khobahi, S., Naimipour, N., Soltanalian, M., and Eldar, Y. C.: Deep signal recovery with one-bit quantization. In ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 2987–2991, May 2019.

242. Khobahi, S., Bose, A., and Soltanalian, M.: Deep radar waveform design for efficient automotive radar sensing. arXiv preprint arXiv:1912.08180, 2019.

243. Perrone, D. and Favaro, P.: A clearer picture of total variation blind deconvolution. IEEE Transactions on Pattern Analysis and Machine Intelligence, 38(6):1041–1055, June 2016.

244. Fish, D. A., Brinicombe, A. M., Pike, E. R., and Walker, J. G.: Blind deconvolution by means of the Richardson–Lucy algorithm. The Journal of the Optical Society of America A, 12(1):58–65, January 1995.

245. Z. Wang, Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P.: Image quality assessment: from error visibility to structural similarity. IEEE Transactions on Image Processing, 13(4):600–612, April 2004.

246. Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278–2324, November 1998.

247. Khobahi, S., Mostajeran, A., Emadi, M., Wang, P., and Soltanalian, M.: Guaranteed deep learning for reliable radar signal processing. In 2022 IEEE Sensor Array and Multichannel Signal Processing Workshop (SAM), 2022.

248. Jiang, W., Haimovich, A. M., and Simeone, O.: Joint design of radar waveform and detector via end-to-end learning with waveform constraints. IEEE Transactions on Aerospace and Electronic Systems, 2021.

249. Dai, A., Zhang, H., and Sun, H.: Automatic modulation classification using stacked sparse auto-encoders. In 2016 IEEE 13th International Conference on Signal Processing (ICSP), pages 248–252, 2016.

250. Geng, Z., Yan, H., Zhang, J., and Zhu, D.: Deep-learning for radar: A survey. IEEE Access, 9:141800–141818, 2021.

251. Levanon, N. and Mozeson, E.: Radar signals. John Wiley & Sons, 2004.

252. Stoica, P., He, H., and Li, J.: Optimization of the receive filter and transmit sequence for active sensing. IEEE Transactions on Signal Processing, 60(4):1730–1740, 2011.

253. Blunt, S. D. and Gerlach, K.: Adaptive pulse compression via mmse estimation. IEEE Transactions on Aerospace and Electronic Systems, 42(2):572–584, 2006.

254. Stoica, P., Li, J., and Xue, M.: Transmit codes and receive filters for radar. 25(6):94–109, 2008.

255. Stewart, G. W.: Matrix algorithms: volume 1: basic decompositions. SIAM, 1998.

256. Horn, R. A. and Johnson, C. R.: Matrix Analysis. Cambridge university press, 2012.

257. Gustafsson, O., Bertilsson, E., Klasson, J., and Ingemarsson, C.: Approximate neumann series or exact matrix inversion for massive mimo? In 2017 IEEE 24th Symposium on Computer Arithmetic (ARITH), pages 62–63. IEEE, 2017.

258. Zhu, D., Li, B., and Liang, P.: On the matrix inversion approximation based on neumann series in massive mimo systems. In 2015 IEEE international conference on communications (ICC), pages 1763–1769. IEEE, 2015.

VITA

Shahin Khobahi

Email: skhoba2@uic.edu
Alt Email: shahin.khobahi@gmail.com, shahinkhobahi2021@u.northwestern.edu
Phone: +1 (312) 383-9097
Personal Webiste: khobahi.net
Google Scholar: Google Scholar Profile
LinkedIn: https://linkedin.com/in/shahin-khobahi
GitHub: https://github.com/skhobahi

EDUCATION

University of Illinois at Chicago, Chicago, IL — Aug. 2016 - Present
Ph.D. Candidate in Electrical and Computer Engineering
Overall GPA: **4.0/4.0**
Supervisor: Prof. Mojtaba Soltanalian

University of Illinois at Chicago, Chicago, IL — Aug. 2016 - May. 2021
M.Sc. in Electrical and Computer Engineering
Overall GPA: **4.0/4.0**
Supervisor: Prof. Mojtaba Soltanalian

Isfahan University of Technology, Isfahan, Iran — Sept. 2012 - Jul. 2016
B.Sc., Electrical Engineering
Supervisor: Prof. Mohammad Mahdi Naghsh
Thesis: "Design and Implementation of a Secure Wireless Mesh Sensor Network"

RESEARCH EXPERIENCE

- **Research Assistant**, WaveOPT Lab, University of Illinois at Chicago. — Aug. 2016 - Nov. 2021
  · **Signal Processing**
  · **Communication Systems**
  · **Optimization Theory**
  · **Machine Learning**
  · **Statistical Learning Theory**
  · **Model-Driven Deep Learning**
  · **Statistical and Graph Signal Processing**
  · **Theory of Deep Learning**

Students Co-Supervised

MSc Students

- Yiming Zheng — Summer 2020 - Present
  Research Topic: *Model-Based Deep Learning.*
  Future Position: PhD Student at the University of Illinois at Chicago (UIC).

BSc Students

- Alex Domagala — Fall 2021 - Present
  Research Topic: *Model-Based Deep Learning for LDPC Codes Detection in Massive MIMO Systems.*

RESEARCH GRANTS

I have been the main contributor to the following federal and external funding:

- My research in developing ubiquitous radar systems for autonomous vehicles have been supported in part by:
  - Mitsubishi Electric Research Laboratories Gift Funding (PI: Dr. Mojtaba Soltanalian), Amount: $\$5,000.00$, Spring 2021.
  - Mitsubishi Electric Research Laboratories Gift Funding (PI: Dr. Mojtaba Soltanalian), Amount: $\$5,000.00$, Spring 2020.
  - Discovery Partners Institute – DPI (PI: Dr. Mojtaba Soltanalian), Project Title: *Ubiquitous Radar Systems for Autonomous Vehicles and Advanced Safety in Urban Environments* – developing novel machine learning algorithms and hardware for low-cost and effective autonomous radar sensing, Amount: $\$100,000.00$, 07/01/2019 – 05/15/2021
- NSF (PI: Dr. Mojtaba Soltanalian), Project Title: "*Waveform Design and Processing for Next-Generation Radar Systems – Adaptivity, Agility, and Reliability*", Amount: $\$323,665.00$, 07/01/2018 – 06/30/2022.
- NSF (PI: Dr. Mojtaba Soltanalian), Project Title: "*Low-Resolution Sampling with Generalized Thresholds*", Amount: $\$399,011.00$, 05/01/2017 – 04/30/2022. (Multi-institution project with a budget of $\$1,198,899$.)

PUBLICATIONS

Preprints

(⋆ Equal Contribution)

- **S. Khobahi** and M. Soltanalian, "Foundations of Model-Based Deep Learning: Applications, Interpretability and Performance Guarantees", to be sumbitted to the *IEEE Transactions on Neural Networks and Learning Systems*.
- **S. Khobahi** and M. Soltanalian, "Guaranteed Deep Learning for Reliable Radar Signal Processing", submitted to 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Sep 2021 - **Invited Paper**.
- **S. Khobahi**, A. Domagala, and M. Soltanalian, "Model-Based Deep Learning for LDPC-coded MIMO Channels", to be submitted to 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).
- Y. Zheng, **S. Khobahi**, and M. Soltanalian, "One-Bit Compressive Sensing: Can We Go Deep and Blind?", to be submitted to the IEEE Signal Processing Letters.
- N. Naimipour⋆, **S. Khobahi**⋆, and M. Soltanalian,"Model-Based Deep Learning for Blind Phase Retrieval", to be sumbitted to *IEEE Signal Processing Letters*, 2021.
- Z. Esmaeilbeigi, **S. Khobahi**, and M. Soltanalian, "Deep-RLS: A Model-Inspired Deep Learning Approach to Nonlinear PCA", submitted to 2021 Statistical Signal Processing Workshop, 2021. Preprint arXiv:2011.07458
- N. Naimipour⋆, **S. Khobahi**⋆, and M. Soltanalian, "Unfolded Algorithms for Deep Phase Retrieval", submitted to *IEEE Transactions on Signal Processing*, 2020. Preprint arXiv:2012.11102
- **S. Khobahi**, A. Bose, A. R. Trivedi, and M. Soltanalian, "Towards Interpretable Deep Learning Frameworks for Non-Convex Optimization and Signal Processing", submitted to *IEEE Signal Processing Magazine*, 2021.

Peer-Reviewed Journal Articles

(⋆ Equal Contribution)

- **S. Khobahi**, N. Shlezinger, M. Soltanalian, and Y. C. Eldar, "LoRD-Net: Low Resolution Detection Network for Deep Receivers", *in IEEE Transactions on Signal Processing, vol. 69, pp. 5651-5664, 2021, doi: 10.1109/TSP.2021.3117503.* Preprint arXiv:2102.02993 IEEE Xplore
- **S. Khobahi** and M. Soltanalian, "Model-Based Deep Learning for One-Bit Compressive Sensing", *IEEE Transactions on Signal Processing, vol. 68, pp. 5292-5307*, 2020. IEEE Xplore
  Appeared on the IEEE TSP Popular Articles list, Oct., Nov., and Dec. 2020
- A. Bose⋆, **S. Khobahi**⋆, and M. Soltanalian, "Efficient Waveform Covariance Matrix Design and Antenna Selection for MIMO Radar", *Signal Processing, Elsevier (2020): 107985.* SIGPRO

- **S. Khobahi**, M. Soltanalian, F. Jiang, and A. Lee Swindlehurst, "Optimized Transmission for Parameter Estimation in Wireless Sensor Networks", *IEEE Transactions on Signal and Information Processing over Networks*, vol. 6, pp. 35-47, 2020, doi: 10.1109/TSIPN.2019.2945631. Preprint arXiv:1908.00600 IEEE Xplore. ***Appeared on the IEEE TSIPN Popular Articles list, Dec. 2019 - Dec. 2020***
- M. M. Naghsh, E. Alian, **S. Khobahi**, and O. Rezaei, "A Majorization–Minimization Approach for Reducing Out-of-Band Radiations in OFDM Systems", *IEEE Communications Letters*, Vol. 21, Issue 8, pp. 1739–1742, Aug. 2017. IEEE Xplore

Peer-Reviewed Conference Articles

(* Equal Contribution)

- **S. Khobahi**, N. Shlezinger, M. Soltanalian, and Y. C. Eldar, "Model-Inspired Deep Detection with Low-Resolution Receivers", 2021 IEEE International Symposium on Information Theory (ISIT), 2021, pp. 3349-3354, doi: 10.1109/ISIT45174.2021.9517812
- **S. Khobahi**, A. Bose, and M. Soltanalian, "Deep One-Bit Compressive Autoencoding", 2021 IEEE Statistical Signal Processing Workshop (SSP), 2021, pp. 371-375, doi: 10.1109/SSP49050.2021.9513806. ***Invited Paper***.
- C. Agarwal*, **S. Khobahi***, D. Schonfeld, and M. Soltanalian "CoroNet: A Deep Network Architecture for Semi-Supervised Task-Based Identification of COVID-19 from Chest X-ray Images", in Medical Imaging 2021: Computer-Aided Diagnosis. Vol. 11597. International Society for Optics and Photonics, 2021. SPIE Library, preprint medRxiv
- N. Naimipour*, **S. Khobahi***, and M. Soltanalian, "UPR: A model-driven architecture for deep phase retrieval", 2020 54th Asilomar Conference on Signals, Systems, and Computers, 2020, pp. 205-209, doi: 10.1109/IEEECONF51394.2020.9443438. Preprint arXiv:2003.04396
- C. Agarwal, **S. Khobahi**, A. Bose, M. Soltanalian, and D. Schonfeld "Deep-URL: A Model-Aware Approach to Blind Deconvolution Based on Deep Unfolded Richardson-Lucy Network", 2020 IEEE International Conference on Image Processing (ICIP), 2020, pp. 3299-3303, doi: 10.1109/ICIP40778.2020.9190825. Preprint arXiv:2002.01053, IEEE Xplore
- **S. Khobahi**, A. Bose, and M. Soltanalian, "Deep Radar Waveform Design for Efficient Automotive Radar Sensing", 2020 IEEE 11th Sensor Array and Multichannel Signal Processing Workshop (SAM), 2020, pp. 1-5, doi: 10.1109/SAM48682.2020.9104367. ***Invited Paper***, preprint arXiv:1912.08180, IEEE Xplore
- **S. Khobahi**, N. Naimipour, M. Soltanalian, and Y. C. Eldar, "Deep Signal Recovery With One-Bit Quantization", 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2019, pp. 2987-2991, doi: 10.1109/ICASSP.2019.8683876. Preprint arXiv:1812.00797, IEEE Xplore
- A. Bose*, **S. Khobahi***, and M. Soltanalian, "Joint Optimization of Waveform Covariance Matrix and Antenna Selection for MIMO Radar", 2019 53rd Asilomar Conference on Signals, Systems, and Computers, 2019, pp. 1534-1538, doi: 10.1109/IEEECONF44664.2019.9048709. Preprint arXiv:1910.07591, IEEE Xplore.
- **S. Khobahi** and M. Soltanalian, "Signal Recovery From 1-Bit Quantized Noisy Samples via Adaptive Thresholding", 2018 52nd Asilomar Conference on Signals, Systems, and Computers, 2018, pp. 1757-1761, doi: 10.1109/ACSSC.2018.8645383. Preprint arXiv:1812.03977, IEEE Xplore
- **S. Khobahi** and M. Soltanalian, "Optimized Transmission For Consensus in Wireless Sensor Networks", 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2018, pp. 3419-3423, doi: 10.1109/ICASSP.2018.8461401. Preprint arXiv:1807.11631, IEEE Xplore

WORK EXPERIENCE

- **Zadar Labs, Inc.** (San Jose, California). Nov. 2021 - Present
  · **Senior Research Scientist**: I am the lead scientist for research and development of novel signal processing and data processing algorithms and AI for Zadar 4D imaging radar sensor solutions. I leverage my expertise in signal processing, data processing algorithms, and AI development for the purpose of developing sensor solutions for the next generation of autonomous systems. Specifically, I am leading a team for developing sophisticated signal processing algorithms and I am responsible for designing perception systems and algorithms for autonomous systems and vehicles and trucking, industrial automation, and intelligent signal processing modules to detect, classify, and track pedestrians and environmental objects, and conduct HD mapping, and to further utilize AI models to enable next generation autonomous systems to learn from the ever changing environment and adapt quickly.

- **Zadar Labs, Inc.** (San Jose, California). May 2021 - Aug. 2021
  · **Research Intern - Signal Processing**: I utilized my knowledge in the field of signal processing, telecommunications, mathematics, statistics, artificial intelligence, and machine learning to perform fundamental research for perception systems for the next-generation intelligent and autonomous systems. In particular, I focused on developing novel state-of-the-art signal processing and beamforming techniques for 4D imaging radars. As a result, I devised a novel beamforming technique known as Zadar FFT which can carry out the beamforming task with an outstanding accuracy outperforming all the existing models for this task.
- **KMB TELEMATICS, Inc.** (Arlington, VA). May 2019 - Aug. 2019
  · **Senior Research Engineer, Intern**: Performed fundamental research in optimization theory, machine learning, and novel signal processing techniques in the field of automotive radar with applications in FMCW radar imaging systems. In particular, we developed an efficient optimizer for the task of 2D radar antenna array design for autonomous vehicles.

SELECTED HONORS AND AWARDS

- Recipient of the **2020-21 College of Engineering Graduate Student Award for Exceptional Research Promise**, University of Illinois at Chicago (this scholarship is awarded annually to only one student from the Electrical and Computer Engineering Department, Summer 2021.
- Recipient of the **Firdawsi Science Fellowship Award**, University of Illinois at Chicago (this scholarship is awarded annually to only one student throughout the University of Illinois campuses), Summer 2020.
- Recipient of the **Wexler Award** ($5000), University of Illinois at Chicago (only 2% of the students), Fall 2016.

TEACHING EXPERIENCE

- **Teaching Assistant.** ECE437: Wireless Communications, University of Illinois at Chicago. Spring 2019
- **Teaching Assistant.** ECE417: Digital Signal Processing II, University of Illinois at Chicago. Fall 2018
- **Teaching Assistant.** ECE317: Digital Signal Processing I, University of Illinois at Chicago. Spring 2018
- **Teaching Assistant.** ECE310: Discrete and Continuous Signals and Systems, University of Illinois at Chicago. Fall 2017

PROFESSIONAL SERVICES

- Main Contributer to the NASA's Open-Source Project: "Toward an Open-Source, Python-Powered, Multi-Doppler Radar Analysis Suite", T. Lang, M. Souto, **S. Khobahi**, and B. Jackson.
- Reviewer (Journal) – **Elsevier Journal of Signal Processing**
- Reviewer (Journal) – **IEEE Transactions On Signal and Information Processing over Networks**.
- Reviewer (Journal) – **IEEE Transactions on Signal Processing**.
- Reviewer (Journal) – **IEEE Transactions on Communications**.
- Reviewer (Journal) – **IEEE Transactions on Internet of Things**.
- Reviewer (Journal) – **IEEE Signal Processing Letters**.
- Reviewer (Journal) – **IEEE Communications Letters**.
- Reviewer (Journal) – **MDPI Journal of Sensors**.
- Reviewer (Conference) – **IEEE Vehicular Technology Conference (VTC2019-Spring)**
- Reviewer (Conference) – **European Signal Processing Conference 2017 (EUSIPCO)**.

COMPUTER SKILLS

- **Programming Languages**:
  - C/C++, Python, MATLAB and Simulink, OCaml, Perl, Bash, X86 Assembly
  - Verilog (VHDL), AVR Assembly
  - PHP, HTML
- **Machine Learning and Statistical Inference Frameworks**:
  - PyTorch, Tensorflow, Keras, OpenAI Gym, Numpy

- **Electrical and Computer Engineering Softwares and Modules**:
  - Orcad, Proteus, Pspice and Schematic, HSpice, Altium Designer
- **Data Management**:
  - MySQL, Microsoft Access, NoSQL, Excel
- **Other**:
  - Microsoft Office, LaTeX, OpenCV Lib, VisualStudio, CodeVision, AVR/ARM Programming