Sparse Channel Estimation for Massive MIMO via Deep Compressive Sensing

by

Pengxia Wu

M.A.Sc., Jilin University, 2016
B.Sc., Jilin University, 2013

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

in

THE COLLEGE OF GRADUATE STUDIES

(Electrical Engineering)

THE UNIVERSITY OF BRITISH COLUMBIA

(Okanagan)

December 2021

The following individuals certify that they have read, and recommend to the College of Graduate Studies for acceptance, a thesis/dissertation entitled:

Sparse Channel Estimation for Massive MIMO via Deep Compressive Sensing

submitted by Pengxia Wu in partial fulfilment of the requirements of the degree of Doctor of Philosophy

Julian Cheng, School of Engineering
Supervisor

Chen Feng, School of Engineering
Supervisory Committee Member

Jahangir Hossain, School of Engineering
Supervisory Committee Member

Yves Lucet, Department of Computer Science, Mathematics, Physics and Statistics
University Examiner

Xiaodai Dong, University of Victoria
External Examiner

Abstract

Sparse channel estimation has great potential to address the challenges caused by the large pilot training overhead and the high dimensionality of the estimating channels when acquiring downlink channel state information (CSI) for massive multiple-input multiple-output (MIMO) systems. In practice, the conventional sparse channel estimation schemes employing compressive sensing techniques are unable to achieve satisfactory performance due to the sub-optimality when designing a random measurement matrix and the inherent inefficiency of iterative reconstruction algorithms. This thesis aims to improve sparse channel estimation by incorporating deep learning techniques into the classical compressive sensing framework to optimize the design of the measurement matrix and sparse channel reconstruction algorithms.

First, novel sparse reconstruction algorithms are proposed. The sparse reconstruction problem is reformulated as a least-squares regression penalized by a trimmed ℓ_1-norm regularizer, which removes penalties on several large-magnitude elements and thus can lead to more accurate solutions compared to the widely-used Lasso regularizer. The difference of convex functions programming framework and gradient projection descent are applied to develop sparse reconstructions algorithms. The proposed algorithms have improved accuracy and require a shorter runtime compared to existing reconstruction algorithms.

After developing the novel sparse reconstruction algorithms, we improve the measurement matrix design by using a data-driven method. The measurement matrix is closely associated with the pilot design and is crucial to channel estimation performance. Several model-based autoencoders are proposed to acquire data-driven measurement matrices. When compared with

the existing random matrices, the acquired data-driven measurement matrices can achieve more accurate reconstructions and consume fewer pilots, thereby allowing higher achievable rates in massive MIMO systems.

After optimizing the measurement matrix and the reconstruction individually, we propose a data-driven scheme to optimize jointly the pilot design and sparse reconstruction. The trimmed ridge regression, i.e., a least-squares regression penalized by a trimmed ℓ_2-norm regularizer, is proposed for sparse reconstructions. The gradient projection descent algorithms of trimmed ridge regression are derived and then unfolded into deep networks to construct model-based autoencoders for joint data-driven pilot acquisition and channel reconstructions. Compared with other deep learning methods, the proposed autoencoders can achieve faster channel reconstruction with higher accuracy using fewer pilots.

Lay Summary

Wireless communication requires to estimate the channel properties, i.e., the combined effects on the transmit signal of the propagation environment from transmitter to receiver. The massive MIMO system, which is often equipped with hundreds of antennas, suffers high spectral cost to estimate the channels since each pair of receiver antenna needs to send sufficient signals as the pilots to probe channels. Fortunately, the massive MIMO channel can be represented in sparse matrices having many zero-valued elements and then can be reconstructed accurately from a few observations. In sparse channel estimation, it is important to design the pilots and reconstruction algorithms properly so that channels can be reconstructed accurately using less pilots. In this thesis, several schemes, including novel sparse reconstruction algorithms, data-driven pilot design and a joint data-driven pilot and sparse reconstruction optimization scheme are proposed to achieve faster and more accurate channel reconstructions while consuming less pilots.

Preface

A list of my publications related to this thesis at The University of British Columbia is provided in the following.

Refereed Journal Publications

J1. **Pengxia Wu** and Julian Cheng, "Nonconvex Regularized Gradient Projection Sparse Reconstruction for Massive MIMO Channel Estimation," *IEEE Transactions on Communications*, vol. 69, no. 11, pp. 7722-7735, November 2021.

J2. **Pengxia Wu** and Julian Cheng, "Deep Unfolding Basis Pursuit: Improving Sparse Channel Reconstruction via Data-Driven Measurement Matrices," Submitted on December 2020 to *IEEE Transactions on Wireless Communications*. (Under Review)

J3. **Pengxia Wu** and Julian Cheng, "Joint Pilot Design and Channel Estimation for Massive MIMO via Learning Trimmed Ridge Regression" (In preparation).

Refereed Conference Publications

C1. **Pengxia Wu**, "Data-Driven Measurement Matrix Design for Sparse Channel Reconstruction in Massive MIMO Systems", *Biennial Symposium on Communications 2021 (BSC 2021)*, Saskatoon, Saskatchewan, Canada, June 28-30, 2021.

C2. **Pengxia Wu**, Hui Ma and Julian Cheng, "Sparse Channel Reconstruction With Nonconvex Regularizer via DC Programming for Mas-

sive MIMO Systems", *2020 IEEE Globecom*, Taipei, Taiwan, December 7-12, 2020.

Table of Contents

Abstract . . . . . **iii**

Lay Summary . . . . . **v**

Preface . . . . . **vi**

Table of Contents . . . . . **viii**

List of Figures . . . . . **xii**

List of Acronyms . . . . . **xvi**

List of Symbols . . . . . **xix**

Acknowledgements . . . . . **xxi**

Chapter 1: Introduction **1**

1.1 Background and Motivation 1

1.2 Literature Review 3

1.3 Thesis Outline and Contributions 9

Chapter 2: Background on Sparse Channel Estimation in Massive MIMO Systems **12**

2.1 Massive MIMO Channel Model 12

2.1.1 Multipath Propagation Channel Model 13

2.1.2 Virtual Sparse Channel Representation 14

2.2 Compressive Sensing for Channel Estimation 18

2.2.1 Model-Driven Channel Estimation 19
2.2.2 A Brief Review on Compressive Sensing Theory 20
2.2.3 Compressive Sensing-Based Sparse Channel Estimation for Narrowband Massive MIMO Systems 28
2.3 Deep Learning for Sparse Channel Estimation 30
2.3.1 Data-Driven Channel Estimation 31
2.3.2 Generic Deep Learning Networks 32
2.3.3 Deep Unfolding for Model-Based Deep Learning . . . 34

Chapter 3: Nonconvex Regularized Gradient Projection Algorithms for Sparse Channel Reconstruction . . . 37
3.1 DC Representation for ℓ_0-norm Constraint 37
3.2 Double-Loop DC Gradient Projection Descent for Sparse Reconstruction 40
3.2.1 DC Programming Framework 40
3.2.2 A Double-Loop DC-GPSR Algorithm 41
3.3 Single-Loop DC Gradient Projection Descent for Sparse Reconstruction 44
3.3.1 A Basic Single-Loop DC-GPSR Algorithm 45
3.3.2 An Extension Algorithm for Single-loop DC-GPSR Using Monotonic BB Step Size 48
3.4 Complexity Analysis 50
3.5 Numerical Results 51
3.5.1 Experiment Setup and Performance Metrics 51
3.5.2 Illustrations of Beamspace Channel Reconstructions and Algorithm Convergence Property 53
3.5.3 Performance Comparisons With Other Algorithms . . 56
3.6 Summary 61

Chapter 4: Improving Sparse Channel Reconstruction via Data-Driven Measurement Matrices 62
4.1 Real-valued Measurement Matrix Design 62
4.2 Learning Measurement Matrices 64

4.2.1 Framework of Deep Unfolding *BP-AE* 64
4.2.2 Specific Structures of Deep Unfolding *BP-AE* 67
4.2.3 Extension Models for Learning the Measurement Matrices Having Double Columns 70
4.3 Numerical Results 73
4.3.1 Experiment Setup and Autoencoder Training 74
4.3.2 Reconstructions by Linear Programming Recovery . . 76
4.3.3 Reconstructions Using the DC-GPSR Algorithm . . . 80
4.3.4 Reconstructions Using the GPSR Algorithm 82
4.3.5 Complexity-Accuracy Tradeoffs of the Learned Matrices for Practical Applications 84
4.4 Summary 86

Chapter 5: Joint Pilot Design and Channel Estimation via Deep Compressive Sensing 88
5.1 Trimmed Ridge Regression Algorithms for Sparse Reconstructions 88
5.1.1 Trimmed Ridge Regression for Sparse Reconstruction 89
5.1.2 Solving Trimmed Ridge Regression Under the DC Programming Framework 90
5.1.3 Practical Algorithms for Trimmed Ridge Regression . 92
5.2 Trimmed Ridge Regression Autoencoder for Joint Data-Driven Pilot Design and Channel Reconstruction 94
5.2.1 The Trimmed Ridge Regression Autoencoder 95
5.2.2 Theoretical Insight for Joint Learning of Measurement Matrix and Sparse Reconstruction 99
5.3 Numerical Results 101
5.3.1 Experiment Setup 102
5.3.2 Performance Comparisons With Other Deep Learning Methods 103
5.3.3 Performance Comparisons With Previous Proposed Sparse Channel Reconstruction Schemes 107
5.4 Summary 110

Chapter 6: Conclusions 111
6.1 Summary of Contributions 111
6.2 Future Works 113

Bibliography 114

Appendices 126
Appendix A: Proof of Theorem 3.1 126
Appendix B: Proof of Theorem 3.2 127
Appendix C: A Toy Experiment With a Manually Structured Dataset 129

List of Figures

Figure 2.1 Categories of pilot training channel estimation methods 19

Figure 2.2 Representation of model-based pilot training channel estimation 20

Figure 2.3 Compressive sensing framework 21

Figure 2.4 Data-driven pilot training channel estimation pipeline 32

Figure 2.5 Unfolding an iterative algorithm into a deep network [1] 34

Figure 2.6 LISTA: deep unfolding of the ISTA [1] 36

Figure 3.1 Channel magnitudes of a true channel sample $\mathbf{x}_{\text{opt}}$ and the reconstruction $\hat{\mathbf{x}}$ by the conventional ℓ_1-regularized GPSR algorithm, the proposed algorithms DlDC-GPSR and SlDC-GPSR-BB. The normalized squared ℓ_2-error is defined as $\|\mathbf{x}_{\text{opt}} - \hat{\mathbf{x}}\|_2^2 / \|\mathbf{x}_{\text{opt}}\|_2^2$ 53

Figure 3.2 Objective values versus iterations for reconstructing a sample of beamspace channel vector. The evaluating objectives are $\frac{1}{2}\|\mathbf{y} - \mathbf{\Phi x}\|_2^2 + \rho(\|\mathbf{x}\|_1 - \|\mathbf{x}\|_{K,1})$ for DlDC-GPSR and SlDC-GPSR-BB, and is $\frac{1}{2}\|\mathbf{y} - \mathbf{\Phi x}\|_2^2 + \rho\|\mathbf{x}\|_1$ for ℓ_1-regularized GPSR 54

Figure 3.3 Objective values ($\frac{1}{2}\|\mathbf{y} - \mathbf{\Phi x}\|_2^2 + \rho\|\mathbf{x}\|_1$) versus iterations for the reconstruction of a sample of beamspace channel vector 55

Figure 3.4 Reconstruction error ($\|\hat{\mathbf{x}} - \mathbf{x}_{\text{opt}}\|_2^2 / \|\mathbf{x}_{\text{opt}}\|_2^2$) versus iterations for the reconstruction of a sample of beamspace channel vector 56

Figure 3.5 Channel reconstruction NMSE versus SNR (dB) with training pilot length $M = 128$ 57
Figure 3.6 Achievable spectral efficiency R/B versus training pilot length at SNR = 25 dB 58
Figure 3.7 Achievable spectral efficiency R/B versus training pilot length at SNR = 40 dB 58
Figure 3.8 NMSE comparison for the training pilot length $M = 96$. The SCAMPI algorithm under uniform distribution is configured to have a 16×16 lens antenna array. The SD algorithm and the proposed SlDC-GPSR-BB and DlDC-GPSR algorithms are configured to have a ULA with 256 antennas. 60
Figure 3.9 Comparison of the achievable spectral efficiency R/B for the training pilot length $M = 96$. The SCAMPI algorithm under uniform distribution is configured to have a 16×16 lens antenna array. The SD algorithm and the proposed SlDC-GPSR-BB and DlDC-GPSR algorithms are configured to have a ULA with 256 antennas. 60

Figure 4.1 Framework of deep unfolding $BP\text{-}AE$ 65
Figure 4.2 Diagram showing the structure of deep $BP\text{-}AE$ . . . 67
Figure 4.3 Computation graph of a $BP\text{-}sAE$ decoder; the module BN represents batch normalization 69
Figure 4.4 Computation graph of a $BP\text{-}gAE$ decoder; the module BN represents batch normalization 69
Figure 4.5 Diagram that shows the $BP\text{-}sAEcat$ extension model; the module BN represents batch normalization and ReLU stands for a rectified linear unit 70
Figure 4.6 Diagram that shows the $BP\text{-}gAEcat$ extension model; the module BN represents batch normalization and ReLU stands for a rectified linear unit 71
Figure 4.7 Diagram of the performed experiment 73

Figure 4.8 Effective achievable rates versus compressed dimension M for different measurement matrices 79
Figure 4.9 Effective achievable rates versus compressed dimension M for different measurement matrices 80
Figure 4.10 Sparse channel reconstruction illustrations produced using the DC-GPSR algorithm with measurement matrices of the size 32×256 81
Figure 4.11 Sparse channel reconstruction illustrations produced using the DC-GPSR algorithm with measurement matrices of the size 48×512 82
Figure 4.12 Reconstruction NMSE versus SNR produced using the GPSR algorithm at compressed dimensions $M = 32$ for the learned matrices $\mathbf{\Phi}_{sae} \in \mathbb{R}^{M\times 256}$, $\mathbf{\Phi}_{gae} \in \mathbb{R}^{M\times 256}$, $\mathbf{\Phi}_{saec} \in \mathbb{R}^{M\times 512}$, $\mathbf{\Phi}_{gaec} \in \mathbb{R}^{M\times 512}$ and random matrices $\mathbf{G} \in \mathbb{R}^{M\times 256}$, $\mathbf{B} \in \mathbb{R}^{M\times 256}$, $\mathbf{G}_c \in \mathbb{R}^{M\times 512}$, $\mathbf{B}_c \in \mathbb{R}^{M\times 512}$ 83
Figure 4.13 Reconstruction NMSE versus SNR produced using the GPSR algorithm at compressed dimensions $M = 40$ for the learned matrices $\mathbf{\Phi}_{sae} \in \mathbb{R}^{M\times 256}$, $\mathbf{\Phi}_{gae} \in \mathbb{R}^{M\times 256}$, $\mathbf{\Phi}_{saec} \in \mathbb{R}^{M\times 512}$, $\mathbf{\Phi}_{gaec} \in \mathbb{R}^{M\times 512}$ and random matrices $\mathbf{G} \in \mathbb{R}^{M\times 256}$, $\mathbf{B} \in \mathbb{R}^{M\times 256}$, $\mathbf{G}_c \in \mathbb{R}^{M\times 512}$, $\mathbf{B}_c \in \mathbb{R}^{M\times 512}$ 84
Figure 4.14 Reconstruction NMSE versus SNR by GPSR algorithm at compressed dimensions $M = 56$ for the learned matrices $\mathbf{\Phi}_{sae} \in \mathbb{R}^{M\times 256}$, $\mathbf{\Phi}_{gae} \in \mathbb{R}^{M\times 256}$, $\mathbf{\Phi}_{saec} \in \mathbb{R}^{M\times 512}$, $\mathbf{\Phi}_{gaec} \in \mathbb{R}^{M\times 512}$ and random matrices $\mathbf{G} \in \mathbb{R}^{M\times 256}$, $\mathbf{B} \in \mathbb{R}^{M\times 256}$, $\mathbf{G}_c \in \mathbb{R}^{M\times 512}$, $\mathbf{B}_c \in \mathbb{R}^{M\times 512}$ . . 85
Figure 4.15 Reconstruction NMSE versus SNR by GPSR algorithm at compressed dimensions $M = 72$ for the learned matrices $\mathbf{\Phi}_{sae} \in \mathbb{R}^{M\times 256}$, $\mathbf{\Phi}_{gae} \in \mathbb{R}^{M\times 256}$, $\mathbf{\Phi}_{saec} \in \mathbb{R}^{M\times 512}$, $\mathbf{\Phi}_{gaec} \in \mathbb{R}^{M\times 512}$ and random matrices $\mathbf{G} \in \mathbb{R}^{M\times 256}$, $\mathbf{B} \in \mathbb{R}^{M\times 256}$, $\mathbf{G}_c \in \mathbb{R}^{M\times 512}$, $\mathbf{B}_c \in \mathbb{R}^{M\times 512}$ . . 86

Figure 5.1 The framework of the TrrAE 96
Figure 5.2 A single-layer structure of the unfolding Trr-MoGPD 97
Figure 5.3 A single-layer structure of the unfolding Trr-NaGPD 97
Figure 5.4 Processing pipeline that shows the offline training process and the online estimation process for designing data-driven pilots and estimating the beamspace channels 98
Figure 5.5 The NMSE versus pilot length for noiseless scenarios 103
Figure 5.6 The NMSE versus pilot length when the standard deviation of noise is $\sigma = 0.001$ 104
Figure 5.7 The NMSE versus pilot length when the standard deviation of noise is $\sigma = 0.01$ 104
Figure 5.8 The NMSE versus pilot length when the standard deviation of noise is $\sigma = 0.1$ 105
Figure 5.9 The NMSE at different noise strengths when the pilot length is $M = 64$ (symbols) 106
Figure 5.10 The NMSE at different noise strengths when the pilot length is $M = 128$ (symbols) 106
Figure 5.11 The efficient achievable rate versus pilot length when the standard deviation of noise is $\sigma = 0.001$ 107
Figure 5.12 Reconstruction errors of our proposed schemes when the pilot length is $M = 48$ 108
Figure 5.13 Reconstruction errors of our proposed schemes when the pilot length is $M = 64$ 108
Figure 5.14 Reconstruction errors of our proposed schemes when the pilot length is $M = 80$ 109

List of Acronyms

Acronyms	Definitions
AD	Analog-and-Digital
AMP	Approximate Message Passing
AoA	Angle of Arrival
AoD	Angle of Departure
AWGN	Additive White Gaussian Noise
BB	Barzilai-Borwein
BCQP	Bound Constrained Quadratic Problem
BP-AE	Basis Pursuit Autoencoders
BS	Base Station
CNN	Convolutional Neural Network
CSI	Channel State Information
DC	Difference of Convex functions
DFT	Digital Fourier Transform
DlDC-GPSR	Double-Loop DC-GPSR
DNN	Deep Neural Network
FDD	Frequency Division Duplex
FISTA	Fast Iterative Shrinkage-Thresholding Algorithm
GPSR	Gradient Projection Sparse Recovery
i.i.d.	Independent Identical Distributed

ISTA	Iterative Shrinkage-Thresholding Algorithm
LARS	Least Angle Regression
LDAMP	Learned Denoising-Based Approximate Message Passing
LAMP	Learned Approximate Message Passing
LISTA	Learned Iterative Shrinkage-Thresholding Algorithm
LMMSE	Linear Minimum Mean Square Error Estimation
LP	Linear Programming
LS	Least Squares
MAP	Maximum A Posteriori
MIMO	Multiple-Input Multiple-Output
MMV	Multiple Measurement Vector
MmWave	Millimeter-Wave
MSE	Mean Square ℓ_2-error
MoGPD	Monotone Gradient Projection Descent
NaGPD	Nesterov's Accelerated Gradient Projection Descent
NMSE	Normalized Mean Squared Error
NP	Non-deterministic Polynomial-time
OFDM	Orthogonal Frequency-Division Multiplexing
OMP	Orthogonal Matching Pursuit
PMF	Probability Mass Function
ReLU	Rectified Linear Unit
RIP	Restricted Isometry Property
RNN	Recurrent Neural Network
SGD	Stochastic Gradient Descent

SINR	Signal-to-Interference-plus-Noise Ratio
SlDC-GPSR	Single-Loop DC-GPSR
SNR	Signal-to-Noise Ratio
SpaRSA	Sparse Reconstruction by Separable Approximation
SMV	Single Measurement Vector
TDD	Time Division Duplex
TNN	Two-Layer Neural Network
Trr	Trimmed Ridge Regression
TrrAE	Trimmed Ridge Regression Autoencoders
UE	User Equipment
ULA	Uniform Linear Array

List of Symbols

Symbols	Definitions
$\mathbf{x}$	Vector
$\mathbf{X}$	Matrix
$x_{(i)}$	The ith element of vector $\mathbf{x}$
$\mathbf{I}$	Identity matrix
$\mathbf{0}$	A zero matrix/vector or all-zeros matrix/vector
$\mathbf{1}$	A matrix/vector of ones or all-ones matrix/vector
$\mathbf{A} \succeq \mathbf{B}$	$\mathbf{A} - \mathbf{B}$ is positive semidefinite
$\mathbb{R}$	Real numbers
$\mathbb{C}$	Complex numbers
$\Re$	The real part of a complex number
$\Im$	The complex part of a complex number
$\arg\max\{\cdot\}$	Points of the domain of the function at which the function values are maximized
$\max\{\cdot\}$	Maximum value of the function
$\min\{\cdot\}$	Minimum value of the function
s.t.	Subject to
$\log_2(\cdot)$	Log function with base 2
$\log(\cdot)$	Log function with base 10

$\mathbb{E}[\cdot]$	Statistical expectation
$\lvert\cdot\rvert$	Absolute value of the argument
$(\cdot)^T$	Transpose operation
$(\cdot)^*$	Conjugate operation
$(\cdot)^H$	Conjugate transpose operation
$(\cdot)^{-1}$	Inverse operation
$(\cdot)^\dagger$	Moore-Penrose pseudoinverse
$\lvert\cdot\rvert$	Absolute value or modulus
$\lVert\cdot\rVert_1$	ℓ_1-norm of a vector
$\lVert\cdot\rVert_2$	ℓ_2-norm of a vector
$\lVert\cdot\rVert_F$	Frobenius norm of a matrix
$\lVert\lVert\cdot\rVert\rVert_2$	Spectral norm of a matrix
$\lVert\cdot\rVert_{K,1}$	Top-$(K,1)$ norm of a vector
$\lVert\cdot\rVert_{K,2}$	Top-$(K,2)$ norm of a vector
$\lceil\cdot\rceil$	Ceiling function
$\partial f(\mathbf{x})$	Gradient of function $f(\cdot)$
$:=$	Define
$O(\cdot)$	Order of magnitude of complexity
$\mathrm{SNR}_{\mathrm{eff}}$	Efficient SNR
$\lambda_{max}(\cdot)$	The largest eigenvalue of a matrix

Acknowledgements

I would like to express my sincere gratitude to my supervisor Dr. Julian Cheng, for his constant support, encouragement, and guidance for my research and professional development. It has been a long journey of this PhD study, and Dr. Julian Cheng has helped me a lot. He developed my academic knowledge, research and writing skills. He granted me great flexibility and freedom in my research work. His enthusiasm towards research, persistence, and integrity have inspired me significantly and will continuously influence me. It is my honor to study and do research under his supervision.

I would like to thank my committee members, Dr. Jahangir Hossain and Dr. Chen Feng for giving important advice on my research. I appreciate that Dr. Chen Feng has spent his valuable time discussing with me and offering his insights and guidance that helped me overcome my research difficulties. I would like to extend my thanks to Dr. Xiaodai Dong from University of Victoria who served as my external examiner. I am honored to have her on my committee. I would also like to thank Dr. Yves Lucet for being my University Examiner. I really appreciate their valuable time.

I am appreciative of my friends and colleagues who shared their academic experiences and constructive viewpoints during my PhD study. We built fond memories while studying and doing research in EME2255. I also would like to thank all my dear friends who helped me a lot while I was in Kelowna. I would like to express my sincerest appreciation to Bradley Reinholz for his thoughtful encouragement, caring support and for helping me with both my research accomplishments and future professional development. Lastly, I would like to express my deepest gratitude to my family who have supported me over my lifetime. All my achievements would not have been possible without their unconditional love, support, patience and encouragement.

Chapter 1

Introduction

1.1 Background and Motivation

Massive multiple-input multiple-output (MIMO) is a key technology for the fifth-generation and future communication systems. By deploying a large number of antenna elements at the base station (BS), a massive MIMO system can provide high spectral efficiency through spatial multiplexing over a channel with favorable propagation. A massive MIMO system can either operate in frequency division duplex (FDD) mode or time division duplex (TDD) mode. In both modes, the massive MIMO system rely heavily on accurate knowledge of channel state information (CSI) to obtain potential performance benefits from large antenna arrays. Acquiring CSI is relatively unproblematic in TDD massive MIMO systems. Based on channel reciprocity, one can accurately and efficiently estimate the downlink channels through uplink pilot training in TDD massive MIMO systems; uplink pilot overhead is affordable since it is only proportional to the number of user equipments (UEs) and is independent with the number of BS antennas. On the other hand, the FDD massive MIMO cannot use channel reciprocity because the uplink and downlink channels occupy different spectrum bands. Consequently, downlink channels are often estimated at the UEs and then the UEs send back the estimated CSI to the BS. This downlink channel estimation and CSI feedback process consumes prohibitively high communication resources because both the overheads of downlink pilots and CSI feedback are proportional to the number of BS antennas. Therefore, acquiring CSI in a resource-efficient manner is the main challenge in practical FDD massive MIMO systems. To overcome downlink CSI acquisition challenges for FDD massive MIMO, one can exploit channel sparsity in certain

domains *a priori* and develop sparse channel estimation schemes to simplify CSI acquisition. In such schemes, unknown massive MIMO channels are characterized as high-dimensional channel vectors, which are sparse or can be represented as sparse vectors in a certain basis. Thus, the overhead associated with the downlink pilots and CSI feedback can be substantially reduced because they are no longer proportional to the number of antennas at the BS, but mostly depend on the sparsity level of the channel.

Sparse channel estimation is regarded as a compressive sensing process. Unknown sparse channel vectors are first mapped to a compact subspace to obtain lower-dimensional measurements by a linear transformation associated with a measurement matrix, which is closely related to the pilots for probing channels. Then, high-dimensional sparse channel vectors are reconstructed from the lower-dimensional measurements given the known measurement matrix. Channel reconstruction quality depends on both the designed measurement matrix and the chosen reconstruction algorithms. While the compressive sensing theory provides a well-defined mathematical framework and suggests robust theoretical performance of existing sparse channel estimation schemes, practical performance is often found to be unsatisfactory. The reasons for poor performance can be categorized into two aspects: suboptimal random measurement matrices and inefficient reconstruction algorithms. First, compressive sensing techniques often adopt random measurement matrices [1] to acquire linear measurements. The random measurement matrix performs a random projection, that is a linear transformation to map sparse channel vectors onto a compact subspace, which is the column space of the adopted random measurement matrix. Random projections cannot fully exploit underlying channel structures. Furthermore, practical communication applications require precise CSI to be obtained within the limited channel coherence time. Existing reconstruction algorithms are usually iterative and computationally expensive, which cannot meet the requirements of low-latency and high-accuracy for practical implementations.

The primary motivation of this thesis is to correct the two aforemen-

[1] The random measurement matrix is a matrix having the elements generated by a random variable following a given distribution.

tioned shortcomings inherent in existing compressive-sensing aided CSI acquisition schemes to improve the performance of sparse channel estimation. First, data-driven projections are used to exploit additional structural features beyond simple sparsity of the massive MIMO channels. Next, accurate non-iterative channel reconstruction algorithms are developed. The deep learning techniques show great potential to develop data-driven schemes for pilot design and channel estimation. Among a variety of deep learning schemes, a model-based deep learning technique called deep unfolding is applied since it enables us to develop smaller-sized deep networks that require less training data than conventional generic deep networks. To develop model-based deep learning networks, this thesis proposes to transform the conventional model-based iterative algorithms into data-driven algorithms having customized trainable parameters. More specifically, classical compressing sensing algorithms will be integrated into deep learning architectures to develop a series of deep compressive sensing networks. The proposed deep compressive sensing schemes are expected to have both the model-based architectures and the data-driven properties, such that the proposed deep compressive sensing networks are not only easy to interpret, but also can improve the sparse channel estimation in terms of both accuracy and computational speed.

1.2 Literature Review

Channel sparsity has been exploited for channel estimation for over a decade in many communication applications such as the orthogonal frequency-division multiplexing (OFDM) system [2], ultra-wideband communications [3], pulse-shaping multicarrier systems [4] and underwater acoustic communication [5]. Channels in these applications have sparse impulse responses since a multipath channel has a large delay spread with a small number of nonzero taps [6]. In recent years, the interest in compressive sensing techniques for signal processing has significantly increased [7, 8]. Furthermore, the compressive sensing framework has been widely employed to develop various sparse channel estimations schemes [9]. Given that channels are

sparse or can be sparsely represented in certain basis or dictionary, the compressive sensing theory provides a framework that enables the channel to be robustly estimated from a significantly reduced number of measurements [10–12].

For massive MIMO systems, channels in the angular domain (beamspace) exhibit sparsity [13]. Due to the limited number of scattering clusters and small angular spread of each scattering cluster, the majority of the channel energy is occupied by a limited number of dimensions in beamspace such that the majority of the elements of beamspace channels are either zero or nearly zero. Based on the beamspace sparsity, various sparse channel estimation [14–21] and compressed CSI feedback schemes [22–30] have been proposed using compressive sensing techniques. In compressive sensing-aided beamspace CSI acquisition approaches, the number of required measurements for robust channel reconstructions only depends on the sparsity level of the beamspace channel [11, 31]. Thus, non-orthogonal pilots can be adopted such that the training pilots and CSI feedback overhead are substantially reduced [20, 26]. Furthermore, additional structural features, such as the common sparsity has been used to improve the compressive CSI acquisitions [29]. The common sparsity can arise in the channels at different pilot subcarriers [26] or in the channels of multi-user systems due to the shared local scatterers by users [29]. Besides the widely-investigated multi-path sparsity in beamspace, it was shown that the sparsity of the channel covariance matrix can be exploited to perform the second-order statistic estimation of CSI [32]. More recently, the emerging hybrid analog-and-digital (AD) architecture in millimeter-wave (mmWave) massive MIMO systems causes additional challenges for channel estimation [33]. In such hybrid AD systems, full-dimensional channel vectors have to be estimated from a few number of observations due to the limited number of radio-frequency chains. Since strong sparsity exists in beamspace mmWave channels, compressive sensing-aided beamspace channel estimation and beamforming schemes have provided promising solutions for the mmWave massive MIMO systems with hybrid AD architectures [33–36].

Compressive sensing-based channel estimation schemes involve two main

components affecting the channel reconstruction performance: the measurement matrix design and the sparse reconstruction algorithm. The existing compressive sensing-aided CSI acquisition approaches commonly adopt random matrices as the measurement matrix, because the randomized matrices drawn from Gaussian or Bernoulli distributions have been proven to achieve accurate recovery solutions with high probabilities under the assumptions of sufficient sparsity of the estimating vectors [10]. In practical implementations, the random measurement matrices were found to be suboptimal for all channel realizations and do not lead to satisfactory reconstruction quality, especially when the number of measurements is insufficient [18]. Improving the reconstruction qualities by increasing the number of measurements will improve recovery accuracy, but this approach is undesirable because it increases the required pilots and degrades the spectral efficiency. Therefore, many efforts have been made to optimize the measurement matrix to reduce the number of required measurements for accurate sparse channel reconstructions [7, 17, 18, 37, 38]. An alternative to the random matrix is to design a deterministic matrix [39–41], which should satisfy the restricted isometry property (RIP) to achieve accurate sparse reconstructions. However, it is non-deterministic polynomial-time (NP)-hard to determine explicitly whether a matrix satisfies the RIP [18, 38]. Some deterministic measurement matrices, which approximately satisfy the RIP, were designed for specific applications in an ad hoc manner [42, 43], and do not perform well in reconstructions for different channel realizations. The existing compressed sensing-aided CSI acquisition approaches commonly adopt random matrices as the measurement matrix, because the randomized matrices drawn from Gaussian or Bernoulli distributions have been proven to achieve accurate recovery solutions with high probabilities under the assumptions of sufficient sparsity of the estimating vectors [10]. In practical implementations, the random measurement matrices were found to be suboptimal for all channel realizations and do not lead to satisfactory reconstruction quality, especially when the number of measurements is insufficient [18]. Improving the reconstruction qualities by increasing the number of measurements will improve recovery accuracy, but this approach is undesirable because it

increases the required pilots and degrades the spectral efficiency. Therefore, several efforts have been made to optimize the measurement matrix to reduce the number of required measurements for accurate sparse channel reconstructions [7, 17, 18, 37, 38]. An alternative to the random matrix is to design a deterministic matrix [39–41], which should satisfy the restricted isometry property (RIP) to achieve accurate sparse reconstructions. However, it is non-deterministic polynomial-time (NP)-hard to determine explicitly whether a matrix satisfies the RIP [18, 38]. Some deterministic measurement matrices, which approximately satisfy the RIP, were designed for specific applications in an ad hoc manner [42, 43], and do not perform well in reconstructions for different channel realizations.

Another essential issue is designing the reconstruction algorithms. Existing algorithms can be divided into two main categories: greedy algorithms and convex-relaxation algorithms. The representative greedy algorithms include orthogonal matching pursuit (OMP) [44, 45], CoSaMP [46] and least angle regression (LARS) [47]. Greedy algorithms work well when the estimating vector is sufficiently sparse, but the performance severely degrades when the sparsity is reduced. Several popular convex-relaxation algorithms have been developed from the ℓ_1-relaxation optimization. The sparse recovery problem is essentially a least squares (LS) minimization with a ℓ_0-norm constraint to force sparsity, but it is simpler to obtain approximate solutions by replacing the nonconvex ℓ_0-norm constraint with a convex ℓ_1-norm constraint. Numerous algorithms have been developed to solve the relaxed ℓ_1-norm constrained convex optimizations, such as the iterative shrinkage-thresholding algorithm (ISTA) [48], the fast iterative shrinkage-thresholding algorithm (FISTA) [49], the gradient projection sparse recovery algorithm (GPSR) [50] and the sparse reconstruction by separable approximation (SpaRSA) [51]. These algorithms have theoretical guarantees of convergence within a finite number of iterations due to the convexity of the ℓ_1-relaxation optimization. However, the ℓ_1-norm constraint is a loose approximation of the ℓ_0-norm constraint, and will lead to biased estimates. To improve the reconstruction algorithms, the weighted l_1-minimization was proposed for downlink CSI recovery [21]. Recently, an adaptive reconstruc-

tion algorithm called modified subspace pursuit was proposed for sparse channel estimation. The modified subspace pursuit explores the temporal correlation of massive MIMO channels and adapted to the prior channel support quality parameter [27, 52].

Deep learning techniques have been successfully applied to develop data-driven channel estimations that are distinct from the model-driven channel estimation schemes using compressive sensing techniques [53–62]. A variety of deep learning-based CSI acquisition approaches adopt generic deep network architectures, such as the deep neural network (DNN), convolutional neural network (CNN) and recurrent neural network (RNN), and have empirically shown satisfactory performance [53–55, 57]. However, those early works often treated the deep learning networks as "black-boxes", leading to large-sized deep networks that require a vast amount of training data. The parameter tuning for certain applications often lacks explicit guidelines and requires extensive trial and error. Moreover, the training and prediction results are not easy to verify or interpret by human beings. The challenges of adopting generic deep learning models motivate and inspire the emerging of model-based deep learning techniques. Recently, model-based deep learning methods have attracted significant attention from the wireless communication research community with a focus on solving inverse problems such as the channel estimation and signal detection problems. Model-based deep learning methods are attractive because one can integrate more domain knowledge, including the well-defined system models, signal statistics, estimation and detection algorithms; these domain knowledge allows smaller deep learning networks to be built with improved accuracy and flexibility [63]. The model-based deep learning networks can learn various elements of the model such as the signal representation, iterative parameters including the step size, regularizers, or even the entire inverse function. To develop model-based deep learning networks, deep unfolding is a promising technique that transforms conventional iterative algorithms into learnable algorithms that have trainable parameters and stacked iterative steps [1, 64]. The unfolded deep learning networks often can achieve faster convergence, have better interpretation, and are easy to train even with a small amount of

data. For example, the approximate message passing (AMP) algorithm [65] was unfolded as the learned denoising-based approximate message passing (LDAMP) network [66], and the LDAMP network has been adopted for sparse beamspace channel reconstructions [56].

In a variety of deep learning-based channel estimation schemes, most prior research investigated channel estimation and pilot design separately, and the resulting design is suboptimal. For example, the LDAMP neural network [66] was constructed for sparse beamspace channel reconstruction, but a random measurement matrix was used for implementing the combining network at the receiver [56]. More recently, several works started focusing on developing an end-to-end pilot training and channel reconstruction model and showed that joint design of pilot training and channel reconstruction can improve channel estimation performance. For example, a deep learning model constructed by the two-layer neural networks (TNNs) followed by the DNN was proposed for joint pilot design and channel estimation for multi-user MIMO systems [57], and this method was subsequently extended to massive MIMO systems [67]. Another deep learning model consisting of a fully-connected dimensional reduction encoder followed by a cascaded CNN decoder was proposed for massive MIMO systems [68]. A model consisting of a fully-connected layer followed by a CNN network was proposed for MIMO-OFDM systems to learn frequency-aware pilots and channel estimations [69]. Several learned AMP (LAMP) models were proposed for the channel estimation and feedback in mmWave massive MIMO systems [70].

In summary, existing channel estimation schemes can be categorized into the compressive sensing-based methods and the deep learning-based methods. Compressive sensing-based channel estimation methods commonly use random matrices as the measurement matrix for pilot design and use iterative sparse reconstruction algorithms to reconstruct the channels. On the other hand, the deep learning-based methods commonly adopt generic deep learning models. Several model-based deep learning approaches have been proposed for channel reconstructions but they consider pilot design separately. Therefore, an efficient data-driven measurement matrix design is in demand for improving the general sparse reconstruction algorithms.

Moreover, the model-based deep learning methods are desired to design the non-iterative and interpretable channel reconstruction models that can also jointly optimize the pilots. This thesis fill in these research gaps from three perspectives. We first aim to improve traditional sparse reconstruction algorithms. Then, we propose data-driven measurement matrix design that can further improve the sparse reconstruction algorithms. At last, we propose the model-based deep learning models that can jointly acquire data-driven pilots and accomplish non-iterative channel reconstructions.

1.3 Thesis Outline and Contributions

In this thesis, we investigate the sparse channel estimation that exploits the sparsity feature of virtual channel representations for massive MIMO systems. Both model-driven and data-driven methods are explored to improve massive MIMO sparse channel estimations. For the model-driven method, we focus on the compressive sensing-based method and propose novel algorithms to improve sparse channel reconstructions. On the other hand, we incorporate the data-driven elements into the classical compressive sensing framework to design improved measurement matrices and non-iterative sparse channel estimation algorithms. The developed model-based deep learning schemes are referred to as the deep compressive sensing. In particular, we present the following three research topics:

- We propose novel sparse reconstruction algorithms that can improve the reconstruction ability of compressive sensing-based sparse channel estimation methods.
- We propose data-driven measurement matrix designs for pilot constructions that can improve the sensing ability of compressive sensing-based sparse channel estimation methods.
- We propose end-to-end model-based deep networks that are data-driven alternatives to the classical compressive sensing-based framework and can jointly improve pilot design and sparse channel reconstructions.

Chapter 1 presents the background and challenges on massive MIMO channel estimation and the motivations of our research. In addition, this chapter provides a detailed literature review on various approaches for sparse channel estimations, especially compressive sensing-based approaches and deep learning-based approaches. After that, the thesis outline and main contributions are summarized.

Chapter 2 provides the required technical background for the thesis. First, the massive MIMO channel model and sparse virtual channel representations are presented. Next, compressive sensing-based and deep learning-based sparse channel estimations are introduced. We present the two main classes of approaches: the model-driven approach and the data-driven approach. The technical premise of compressive sensing and deep learning techniques are also reviewed.

Chapter 3 investigates the issue of sparse reconstructions and proposes novel sparse reconstruction algorithms for compressive sensing-based channel estimations. The sparse reconstruction, which is an NP-hard ℓ_0-norm minimization optimization, is formulated as an LS minimization with a novel non-convex regularizer, i.e., trimmed ℓ_1-norm regularizer. The trimmed ℓ_1-norm regularizer is equivalent to (instead of approximates) the original ℓ_0-norm constraint, thus it can lead to more accurate reconstruction results compared to the convex relaxation methods. The resulting nonconvex optimization is solved by applying the DC programming and gradient projection descent, and three novel algorithms are developed.

Chapter 4 explores the issue of measurement matrix design, which is closely related to the pilot design in channel estimations, and proposes data-driven measurement matrices to improve the compressive sensing-based sparse channel estimations. The proposed sparse algorithms in Chapter 3 are shown to be further improved by using the proposed data-driven measurement matrices to replace common random matrices. To acquire data-driven measurement matrices, the deep compressive sensing framework is proposed that uses the deep unfolding technique and incorporates the classical compressive sensing framework into building model-based deep networks. More specifically, the subgradient descent of the ℓ_1-minimization sparse recon-

struction, which is referred to as the basis pursuit algorithm, is unfolded to construct autoencoders. The constructed autoencoders are trained to acquire data-driven measurement matrices that can achieve more accurate reconstructions while requiring less pilots.

Chapter 5 explores join pilot design and channel estimation by an end-to-end data-driven method and is distinct from Chapter 3 and Chapter 4 that separately improve the sparse reconstructions and measurement matrices. Novel sparse reconstruction algorithms are first derived and then unfolded into deep networks. The unfolded deep network functions as a decoder and is connected with a linear encoder to fit in the deep compressive sensing framework. Novel model-driven deep compressive autoencoders are proposed and then trained to acquire a data-driven measurement matrix and jointly perform channel estimation. Simulation results show the proposed deep compressive sensing autoencoders can produce faster and more accurate sparse channel reconstructions while using less pilots than conventional compressive sensing-based methods.

Chapter 6 summarizes the key findings presented in this thesis and summarizes the contributions. In addition, some future works are also suggested.

Chapter 2

Background on Sparse Channel Estimation in Massive MIMO Systems

In this chapter, we will introduce background knowledge on sparse channel estimation for massive MIMO systems. The multipath channel model of massive MIMO will be first introduced and then followed with a discussion on sparse virtual channel representations. After that, two classes of channel estimation approaches will be introduced: the compressive sensing-based and the deep learning-based approaches, which respectively represent the model-driven and data-driven methods for channel estimation. Also, the basic knowledge about compressive sensing and deep learning will be briefly reviewed.

2.1 Massive MIMO Channel Model

Wireless channels of a massive MIMO system refers to all the effects on transmitted signals from transmitter antennas to receiver antennas. The effects on a radio signal are the combination of three physical propagation mechanisms including reflection, diffraction, and scattering from surrounding objects, leading to a phenomenon known as the multipath propagation. This section first introduces the physical characterization of the multipath propagation channel of a massive MIMO system. Next, the virtual sparse representations of massive MIMO channels are discussed.

2.1.1 Multipath Propagation Channel Model

For a massive MIMO system equipped with uniform linear arrays (ULAs) having N_t antennas at the transmitter and N_r antennas at the receiver, the channel in spatial-temporal space can be modelled as a linear, time-varying system $\mathcal{H}(\cdot)$. The signal $\mathbf{s}(t) \in \mathbb{R}^{N_t}$ transmitted through the channel can be represented as [9]

$$\mathcal{H}(\mathbf{s}(t)) = \int_{\mathbb{R}} \mathbf{H}(t, f)\mathbf{S}(f)e^{j2\pi ft}df \tag{2.1}$$

where $\mathcal{H}(\mathbf{s}(t)) \in \mathbb{C}^{N_r}$ is the channel output; $\mathbf{H}(t, f) \in \mathbb{C}^{N_r \times N_t}$ is the frequency response matrix of the channel, which is time-varying due to the motions of the transceiver and the objects in the propagation environment; $\mathbf{S}(f) \in \mathbb{C}^{N_t}$ is the Fourier transform of the signal $\mathbf{s}(t)$.

Due to multipath propagation, the received signal is the superposition of multiple copies of the transmitted signal, i.e., multipath signal components, where each multipath signal component arrives at the receiver with different attenuation, delay, phase shift and frequency shift. The channel response matrix $\mathbf{H}(t, f)$ can be expressed using the multipath model as [9, 71]

$$\mathbf{H}(t, f) = \sqrt{N_r N_t} \sum_{l=1}^{N_p} \beta_l \boldsymbol{\alpha}_r(\theta_{r,l})\boldsymbol{\alpha}_t^H(\theta_{t,l})e^{j2\pi v_l t}e^{-j2\pi\tau_l f} \tag{2.2}$$

where N_p is the number of multiple paths; β_l is the complex path gain for the lth path; $\boldsymbol{\alpha}_r(\theta_{r,l})$ and $\boldsymbol{\alpha}_t(\theta_{t,l})$ are respectively the array steering vectors of receiver and transmitter array for the lth path, and the corresponding angle of arrival (AoA) and the angle of departure (AoD) are $\theta_{r,l}$ and $\theta_{t,l}$; $\tau_l \in [0, \tau_{max}]$ and $v_l \in [-v_{max}/2, v_{max}/2]$ are the delay and Doppler shifts of the lth path, and where τ_{max} and v_{max} are the delay and Doppler spreads of the channel. An array steering vector represents the array phase profile as a function of physical AoA or AoD. For the one-dimensional ULA having N_t transmitter antennas and N_r receiver antennas, the array steering vectors

for a signal having the AoD θ_t and the AoA θ_r are

$$\begin{aligned}\boldsymbol{\alpha}_r(\theta_t) &= \frac{1}{\sqrt{N_t}}[1, e^{-j2\pi\theta_t}, e^{-j4\pi\theta_t}, ..., e^{-j2\pi\theta_t(N-1)}]^T \\ \boldsymbol{\alpha}_r(\theta_r) &= \frac{1}{\sqrt{N_r}}[1, e^{-j2\pi\theta_r}, e^{-j4\pi\theta_r}, ..., e^{-j2\pi\theta_r(N-1)}]^T\end{aligned} \tag{2.3}$$

where $\theta_t, \theta_r \in [-1/2, 1/2]$ is the normalized spatial angle, which is related to the physical angle $\vartheta_r, \vartheta_t \in [-\pi/2, \pi/2]$ by $\theta_t = \frac{d}{\lambda}\sin(\vartheta_t), \theta_r = \frac{d}{\lambda}\sin(\vartheta_r)$, and where $d = \lambda/2$ is the half-wavelength antenna spacing and λ is the wavelength. For the signal propagated through N_p paths, the channel response can then be expressed as

$$\mathcal{H}(\mathbf{s}(t)) = \sqrt{N_r N_t}\sum_{l=1}^{N_p}\beta_l\boldsymbol{\alpha}_r(\theta_{r,l})\boldsymbol{\alpha}_t^H(\theta_{t,l})e^{j2\pi v_l t}\mathbf{s}(t-\tau_l). \tag{2.4}$$

Now, a real-valued signal $\mathbf{s}(t)$ is considered to have a duration T and a two-sided bandwidth B, which results in a temporal-frequent signal space having the dimension of BT. According to the relationship between the signal parameters, including the signal duration T and the signal bandwidth B, and the channel parameters, including the Doppler spread v_{max} and the delay spread τ_{max}, the channel can be classified into four categories, as shown in Table 2.1 [9].

Table 2.1: Channel Types

Channel Type	$B\tau_{max}$	Tv_{max}
Nonselective Channels	$\ll 1$	$\ll 1$
Frequency-Selective Channels	≥ 1	$\ll 1$
Time-Selective Channels	$\ll 1$	≥ 1
Doubly-Selective Channels	≥ 1	≥ 1

2.1.2 Virtual Sparse Channel Representation

The channel model $\mathbf{H}(t, f)$ in (2.2) describes physical characterization of the N_p path propagations using the parameters $\{\beta_l, \theta_{r,l}, \theta_{t,l}, \tau_l, v_l\}$ for

$l \in \{1, ..., N_p\}$. Instead of requiring those propagation parameters for each path explicitly, a communication system usually requires to know the channel response matrix $\mathbf{H}(t, f)$. However, for a massive MIMO system having large number of transmit antennas, to estimate directly the matrix $\mathbf{H}(t, f)$, which is high-dimensional, often requires a large number of pilot symbols and causes high computational complexity. Fortunately, the sparse virtual channel representations can be exploited to simplify the channel estimation.

The channel response matrix $\mathbf{H}(t, f)$ can be approximately represented using the four-dimensional Fourier series as [9]

$$\mathbf{H}(t,f) \approx \sum_{i=1}^{N_r}\sum_{j=1}^{N_t}\sum_{d=1}^{D}\sum_{s=-S}^{S} H_v(i,j,d,s)\boldsymbol{\alpha}_r(\frac{i}{N_r})\boldsymbol{\alpha}_t^H(\frac{j}{N_t})e^{j2\pi\frac{s}{T}t}e^{-j2\pi\frac{d}{B}f} \tag{2.5}$$

where $H_v(i, j, d, s)$ is the virtual channel coefficients; $\boldsymbol{\alpha}_r(\frac{i}{N_r}), \boldsymbol{\alpha}_t^H(\frac{j}{N_t})$ and $\{e^{j2\pi\frac{s}{T}t}\}, \{e^{j2\pi\frac{d}{B}f}\}$ comprise the spatial-spectral-temporal Fourier bases. This four-dimensional Fourier bases defines a fixed uniformly sampling of $\mathbf{H}(t, f)$ in the virtual angular-delay-Doppler domain at the following rate,

$$\Delta\theta_r = 1/N_r, \Delta\theta_t = 1/N_t, \Delta\tau = 1/B, \Delta v = 1/T. \tag{2.6}$$

Eq. (2.5) shows that the channel response matrix $\mathbf{H}(t, f)$ is a linear representation of the virtual channel coefficients $\{H_v(i, j, d, s)\}$ for $i \in [1, ..., N_r], j \in [1, ..., N_t], d \in [1, ..., D], s \in [-S, ..., S]$. The number of virtual channel coefficients is $V = N_r N_t D(2S + 1)$, where $N_r, N_t, D := \lceil B\tau_{max} \rceil + 1$ and $S := \lceil T v_{max}/2 \rceil$ represent the number of resolvable virtual AoAs, AoDs, delays and one-sided Doppler shifts in the angular-delay-Doppler domain respectively.

The virtual channel coefficient $H_v(i, j, d, s)$ can be represented by $\mathbf{H}(t, f)$ as

$$H_v(i,j,d,s) \approx \frac{1}{BT}\int_0^T\int_{-\frac{B}{2}}^{\frac{B}{2}} \boldsymbol{\alpha}_r^H(\frac{i}{N_r})\mathbf{H}(t,f)\boldsymbol{\alpha}_t(\frac{j}{N_t})e^{j2\pi\frac{s}{T}t}e^{-j2\pi\frac{d}{B}f}dtdf. \tag{2.7}$$

After a substitution of (2.2) into (2.7), the virtual channel coefficient can

be expressed as [9]

$$\begin{aligned} H_v(i,j,d,s) &\approx \sum_{l=1}^{N_p} \beta_l f_{N_r}(i/N_r - \theta_{r,l}) f^*_{N_t}(j/N_t - \theta_{t,l}) e^{-j\pi(s-Tv_l)} \\ &\quad \operatorname{sinc}(s - Tv_l) \operatorname{sinc}(d - B\tau_l) \\ &\approx \sum_{l \in S_{r,i} \cap S_{t,j} \cap S_{\tau,d} \cap S_{v,s}} \beta_l \end{aligned} \tag{2.8}$$

where $f_{N_r}(\cdot)$ and $f_{N_t}(\cdot)$ are Dirichlet kernels, and a Dirichlet kernel is defined as $f_N(\theta) := \frac{1}{N}\sum_{i=0}^{N-1} e^{-j2\pi i\theta}$; $\operatorname{sinc}(\cdot)$ is the Sinc function defined as $\operatorname{sinc}(x) := \sin(\pi x)/\pi x$. Sidelobes of Dirichlet kernel and sinc function lead to power leakage, which can be observed by the approximation signs in (2.8). The approximation becomes more accurate when increasing T, B, N_r and N_t. The sets $S_{r,i}, S_{t,j}, S_{\tau,d}$ and $S_{v,s}$ in (2.8) are

$$\begin{aligned} S_{r,i} &:= \{l : \theta_{r,l} \in (i/N_r - 1/2N_r, i/N_r + 1/2N_r]\} \\ S_{t,j} &:= \{l : \theta_{t,l} \in (j/N_t - 1/2N_t, j/N_t + 1/2N_t]\} \\ S_{\tau,d} &:= \{l : \tau_l \in (d/B - 1/2B, d/B + 1/2B]\} \\ S_{v,s} &:= \{l : v_l \in (s/T - 1/2T, s/T + 1/2T]\} \end{aligned} \tag{2.9}$$

where $S_{r,i}$ is the set of all propagation paths having the AoAs within the range of $[i/N_r - 1/2N_r, i/N_r + 1/2N_r]$; $S_{t,j}$ is the set of all propagation paths having the AoDs within the range of $[j/N_t - 1/2N_t, j/N_t + 1/2N_t]$; $S_{\tau,d}$ is the set of propagation paths having the delays within the range of $(d/B - 1/2B, d/B + 1/2B]$; $S_{v,s}$ is the set of propagation paths having the Doppler shifts within the range of $(s/T - 1/2T, s/T + 1/2T]$. Therefore, $S_{r,i} \cap S_{t,j} \cap S_{\tau,d} \cap S_{v,s}$ represents the set of all the propagation paths whose physical AoAs, AoDs, delays and Doppler shifts within the (i,j,d,s)th virtual bin, which is located around the sampling point $(i/N_r, j/N_t, d/B, s/T)$ and has the size of $\Delta\theta_r \times \Delta\theta_t \times \Delta\tau \times \Delta v$ in the four-dimensional angular-delay-Doppler domain, where $\Delta\theta_r, \Delta\theta_t, \Delta\tau$, and Δv are defined in (2.6). Hence, we can interpret that the virtual channel coefficient $H_v(i,j,d,s)$ is essentially the sum of all complex channel gains of the physical paths having the AoAs,

AoDs, delays and Doppler shifts within the (i, j, d, s)th virtual bin.

Equations (2.5) and (2.7) indicate that the virtual channel coefficients $\{H_v(i, j, d, s)\}$ can completely define the channel response matrix $\mathbf{H}(t, f)$. Thus, channel estimation of $\mathbf{H}(t, f)$ can be equivalently completed by estimating all the virtual channel coefficients $\{H_v(i, j, d, s)\}$. Considering that a limited number of scatters exist in the propagation environment, and that each scatter has limited angular, delay and Doppler-shift spreads, only a few virtual bins contain physical paths while most of the virtual channel coefficients are zero or nearly zero. This feature of most virtual channel coefficients being zeros is known as the channel sparsity of massive MIMO channels.

A special case: narrowband virtual sparse channel representation

For narrowband block-fading massive MIMO channels, we can neglect the time varying and frequency dependency in a channel coherence block, i.e., $\mathbf{H}(t, f) = \mathbf{H}_c$. The multipath propagation channel model in (2.2) reduces to

$$\mathbf{H}_c = \sqrt{N_r N_t} \sum_{l=1}^{N_p} \beta_l \boldsymbol{\alpha}_r(\theta_{r,l}) \boldsymbol{\alpha}_t^H(\theta_{t,l}). \tag{2.10}$$

The channel matrix in (2.5) reduces to

$$\begin{aligned} \mathbf{H}_c &\approx \sum_{i=1}^{N_r} \sum_{j=1}^{N_t} H_a(i, j) \boldsymbol{\alpha}_r(\frac{i}{N_r}) \boldsymbol{\alpha}_t^H(\frac{j}{N_t}) \\ &= \mathbf{U}_r \mathbf{H}_a \mathbf{U}_t^H \end{aligned} \tag{2.11}$$

where $\mathbf{U}_t \in \mathbb{C}^{N_t \times N_t}$ and $\mathbf{U}_r \in \mathbb{C}^{N_r \times N_r}$ are unitary digital Fourier transform (DFT) matrices, and they can be expressed using the array steering vectors as

$$\begin{aligned} \mathbf{U}_t &= [\boldsymbol{\alpha}_t(\theta_{t,0}), \boldsymbol{\alpha}_t(\theta_{t,1}), ..., \boldsymbol{\alpha}_t(\theta_{t,N_t-1})]^T \\ \mathbf{U}_r &= [\boldsymbol{\alpha}_r(\theta_{r,0}), \boldsymbol{\alpha}_r(\theta_{r,1}), ..., \boldsymbol{\alpha}_r(\theta_{r,N_r-1})]^T \end{aligned} \tag{2.12}$$

where $\theta_{t,i} = i/N_r$ for $i = \{1, ..., N_r\}$ and $\theta_{r,j} = j/N_t$ for $j = \{1, ..., N_t\}$ are the virtual AoDs and AoAs. The virtual channel coefficient reduces to

$$\begin{aligned} H_a(i,j) &= \boldsymbol{\alpha}_r^H(\frac{i}{N_r})\mathbf{H}_c\boldsymbol{\alpha}_t(\frac{j}{N_t}) \\ &= \sum_{l=1}^{N_p} \beta_l f_{N_r}(i/N_r - \theta_{r,l}) f_{N_t}^*(k/N_t - \theta_{t,l}) \\ &\approx \sum_{l \in S_{r,i} \cap S_{t,j}} \beta_l. \end{aligned} \tag{2.13}$$

Eq. (2.13) indicates that the beamspace channel coefficient $H_a(i,j)$ is the sum of the complex channel gains of all propagation paths whose AoAs and AoDs are within the (i,j)th virtual grid $(i/N_r - 1/2N_r, i/N_r + 1/2N_r) \times (j/N_t - 1/2N_t, j/N_t + 1/2N_t)$. The virtual channel coefficients $H_a(i,j)$ can be represented by a matrix $\mathbf{H}_a$, which is also called the beamspace channel. Due to the limited number of scatters in the propagation environment and the high dimensionality of massive MIMO channels, only few virtual grids can contain at least one physical path. Consequently, the beamspace channel $\mathbf{H}_a$ is sparse, meaning that most of the beamspace channel coefficients $\{H_a(i,j)\}$ are zero or nearly zeros [71].

2.2 Compressive Sensing for Channel Estimation

Channel estimation approaches can be grouped into two categories: the blind channel estimation and the pilot training channel estimation. The blind channel estimation infers the CSI by exploring the statistics of the received data-carrying signals. For the pilot training channel estimation, transmitters first send pilot signals to probe the channels, and then receivers infer the CSI from the received pilots symbols. It is unsuitable to estimate the rapid time-varying channels in real time using blind channel estimations. Also, the blind channel estimation often involves the inversion computations of large-size matrices. On the contrary, the pilot training channel estimation is more prevalent in existing massive MIMO systems because it has relatively lower computational complexity and can acquire

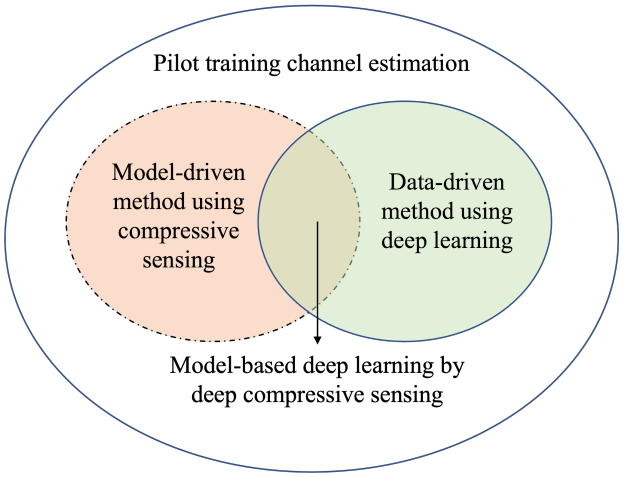


Figure 2.1: Categories of pilot training channel estimation methods

the real-time CSI. The scope of this thesis focuses on pilot training channel estimation. As shown in Fig. 2.1, we classify existing pilot training channel estimation methods into two main classes: the model-driven method that uses compressive sensing and the data-driven method that uses deep learning. The model-driven method often describes the system with explicit models under certain statistical assumptions, while the data-driven method optimizes a trainable deep learning network using a large amount of ground truth data without an explicit model or statistical assumptions. These channel estimation methods will be introduced in this section and the next section.

2.2.1 Model-Driven Channel Estimation

The model-driven method for pilot training channel estimation is illustrated in Fig. 2.2. Usually, the model-based method assumes a linear model. Given the expectations of reducing pilot overhead and computational complexity of massive MIMO systems, the pilot training channel estimation is a task of inferring a large number of unobserved parameters from a limited number of observations. It is generally computationally intractable to infer high-dimensional unobserved parameters from a low-dimensional linear ob-

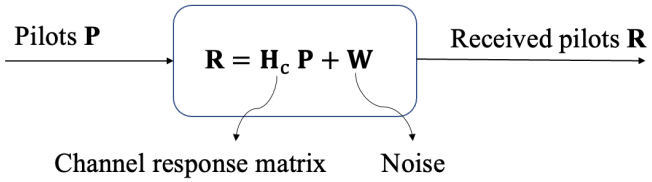


Figure 2.2: Representation of model-based pilot training channel estimation

servations because the underlying problem is under-determined. However, this underdetermined problem is solvable with a certain *a priori*. For example, the problem is solvable when the sparsity is known. In Section 2.1 we have shown that a massive MIMO channel exhibits a sparsity feature when mapped into the angular-delay-Dopper domain using a standard four-dimensional DFT dictionary. By using the channel sparsity as *a priori*, the compressive sensing technique performs well when solving the channel estimation problem with notable advantages including reducing the number of required measurements without degrading the reconstruction fidelity. In the following section, we will first give a brief review on compressive sensing theory and then present how to apply the compressive sensing framework to solve the massive MIMO channel estimation problem.

2.2.2 A Brief Review on Compressive Sensing Theory

Compressive sensing theory provides a paradigm that collects a small set of linear measurements and allows accurate recovery of the high-dimensional sparse data. The compressive sensing framework is illustrated in Fig. 2.3, where $\mathbf{s}$ is an unknown high-dimensional data, $\mathbf{x}$ is the sparse representation of the data $\mathbf{s}$, $\mathbf{\Psi}$ is the dictionary for sparse representation, $\mathbf{A}$ is the sensing matrix, $\mathbf{\Phi}$ is the measurement matrix, $\mathbf{y}$ is the measurement vector containing the compressed measurements, $\hat{\mathbf{x}}$ is the reconstructed sparse vector and $\hat{\mathbf{s}}$ is the recovered data. Two processes are involved in the compressive sensing framework, i.e., sensing and reconstruction of data. The sensing process obtains a limited number of linear samples by combining the conventional

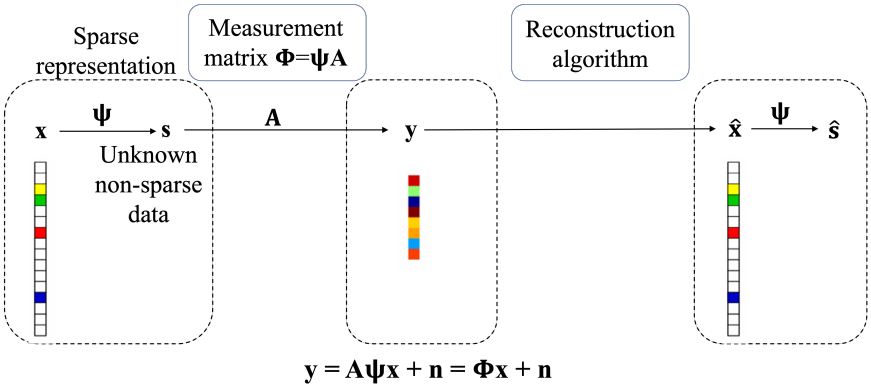


Figure 2.3: Compressive sensing framework

sampling and compression steps. The reconstruction process answers the central question of how to reconstruct the high-dimensional signal from a small number of linear measurements accurately and provide performance guarantees by a variety of sparse recovery algorithms.

2.2.2.1 From Nyquist–Shannon Sampling Theorem to Compressive Sensing

The Nyquist-Shannon sampling theorem states that to rebuild a band limited time-continuous signal, the sampling frequency should be at least two times higher than the highest frequency component of the signal. A discrete version of the sampling theorem states that for a discrete signal $\mathbf{x} \in \mathbb{R}^N$, such as an image with a set of N pixels, the number of required Fourier samples should be equivalent to the dimension of the signal N [72]. However, the required sampling rate can be prohibitively high in many practical applications since processing a large number of samples increases the cost of transmission and storage. Fortunately, many real-world signals have a much lower effective dimensionality than the size of the signal; therefore, one can accurately reconstruct the signal using a sub-Nyquist sampling rate [72]. For example, discrete sparse signals have been shown to achieve perfect reconstructions from partial Fourier coefficients [10, 31, 73]. This phenomenon has inspired the idea of "compressive sensing", which considers

that instead of first sampling a signal at high rate and then compressing the sampled data, we can sense the data directly in a compressive way, i.e., using a lower sampling rate than a standard Nyquist sampling rate. Sampling a time-continuous signal means to obtain measurements by extracting specific points in time, while the sampled measurements of discrete signals are obtained by the inner-product operations between the discrete signal and the given measurement functions [74]. In most compressive sensing contexts, the discrete signal is referred to as a sparse finite-dimensional vector, and the sampling reduces to a matrix-vector product between the measurement matrix and the vector. The compressive sensing has two key factors, namely, the sparsity of the sampled signal and the randomness of sampling [74].

Definition 2.1 (K-sparse signal)**.** For an N-dimensional vector $\mathbf{x}$, the vector $\mathbf{x}$ is called K-sparse when it has at most K nonzero elements, i.e., $\|\mathbf{x}\|_0 \leq K$ for $K \ll N$. The set of nonzero coordinates is defined as the support of $\mathbf{x}$ and denoted as $\text{supp}(\mathbf{x})$, i.e., $\text{supp}(\mathbf{x}) = \{i : x_{(i)} \neq 0\}$ for $\mathbf{x} = [x_{(1)}, ..., x_{(N)}]$, and the $|\text{supp}(\mathbf{x})|$ denotes the cardinality of $\text{supp}(\mathbf{x})$, i.e., $|\text{supp}(\mathbf{x})| = \|\mathbf{x}\|_0$.

In practice, we can obtain the K-sparse approximation of nearly sparse signals by retaining the K elements having the largest absolute values and setting the rest of elements to zeros. Signals are considered as compressible when the signals can be well-approximated by K-sparse signals. A finite-dimensional signal has a sparse or compressible representation if the signal can be represented or well-approximated by a few non-zero coefficients in an appropriate basis or dictionary. Suppose $\mathbf{s} \in \mathbb{C}^N$ is a signal that can be represented by a sparse vector $\mathbf{x}$ in the basis $\mathbf{\Psi}$, i.e., $\mathbf{s} = \mathbf{\Psi x}$. The sensing process can be expressed as $\mathbf{y} = \mathbf{As} = \mathbf{A\Psi x} = \mathbf{\Phi x}$, where $\mathbf{x}$ is the sparse vector containing K nonzero entries and $K \ll N$, $\mathbf{A}$ is called the sensing matrix; $\mathbf{\Psi}$ is called the dictionary where the columns are the basis vectors, $\mathbf{\Phi} = \mathbf{A\Psi}$ is called the measurement matrix and $\mathbf{y} \in \mathbb{C}^M$ is the measurement vector consisting of a set of measurements or samples, where the number of measurements M is much smaller than the dimension of the original signal N, i.e., $M \ll N$.

Compressive sensing theory states that a sparse signal $\mathbf{x}$ with a finite dimension can be recovered from a small set of linear, nonadaptive [2] measurements $\mathbf{y}$ [74]. Here, two main theoretical questions are involved: first, how should we design the measurement matrix $\mathbf{\Phi}$ to ensure that it preserves the information in the signal $\mathbf{x}$; second, how can we recover the sparse signal $\mathbf{x}$ from measurements $\mathbf{y}$ using practical algorithms?

2.2.2.2 Measurement matrix construction

The RIP is a well-known sufficient condition for a desirable measurement matrix to guarantee the successful recovery of the original sparse signal from a few of compressed measurements.

Definition 2.2 (RIP)**.** An $M \times N$ matrix $\mathbf{\Phi}$ satisfies the restricted isometry property of order K if there exists a $\delta_K \in (0, 1)$ such that

$$(1 - \delta_K)\|\mathbf{x}\|_2^2 \leq \|\mathbf{\Phi}\mathbf{x}\|_2^2 \leq (1 + \delta_K)\|\mathbf{x}\|_2^2 \tag{2.14}$$

holds for all N-dimensional sparse vector $\mathbf{x}$ with $|\text{supp}(\mathbf{x})| \leq K$. The smallest δ_K is called the restricted isometry constant of $\mathbf{\Phi}$.

The RIP condition states that a linear transformation by the matrix should be approximately isometric on all K-sparse vectors. In other words, a matrix has restricted isometry if and only if all subset matrices of the size $M \times K$ (any subset of K columns) are almost isometric. It is possible to construct a deterministic matrix of the size $M \times N$ satisfying the RIP of order K, as shown in [39, 40, 75, 76]; however, the required M was often found to be large. For example, in the constructions of [39] and [76], the required number of measurements are $M = O(K^2 \log N)$ and $M = O(KN^\alpha)$, where α is a constant. Fortunately, the random matrices can be shown to have restricted isometries with high probability if the elements are drawn from Gaussian, Bernoulli or the general sub-Gaussian distribution. For example, a matrix with i.i.d. entries drawn from sub-Gaussian distribution satisfies the RIP or order $2K$ with the probability at least $1 - 2\exp(c_1\delta^2 M)$ with

[2] The measurments are non-adaptive meaning that the rows of the sensing matrix are fixed in advance and do not depend on the previously acquired measurements.

$M = O(K\log(N/K)/\delta_{2K}^2)$ for $\delta_{2K} \leq \delta$, where $c_1 > 0$ and $\delta \in (0,1)$ are constants [72, 74]. Other types of random matrices that have been proven to satisfy the RIP with high probabilities include the random Fourier matrix constructed by randomly selecting rows from an $N \times N$ DFT matrix and the random sub-matrices of orthogonal matrices constructed by randomly selecting rows from an $N \times N$ orthonormal matrix, such as the Hadamard matrix [72, 74].

In some applications where the signal $\mathbf{s}$ has a sparse representation $\mathbf{x}$ by a known dictionary $\mathbf{\Psi}$, i.e., $\mathbf{s} = \mathbf{\Psi x}$, we need to design the sensing matrix $\mathbf{A}$ by considering the dictionary $\mathbf{\Psi}$ so that the matrix product $\mathbf{A\Psi}$ satisfies the RIP. Fortunately, the random matrices have a property referred to as *universality* [74]. Given that a sensing matrix $\mathbf{A}$ is drawn from a certain random distribution and the dictionary $\mathbf{\Psi}$ is orthonormal, it is easy to show that the entries in the matrix $\mathbf{A\Psi}$ follow a same random distribution as the matrix $\mathbf{A}$ such that the matrix product $\mathbf{A\Psi}$ also satisfies the RIP with high probability [74]. The universality of random matrices provides significant advantages since the sensing matrix $\mathbf{A}$ can be constructed using random matrices without additional consideration of the matrix $\mathbf{\Psi}$.

2.2.2.3 The ℓ_1-minimization sparse reconstruction

The sparse reconstruction can be formulated as an ℓ_0-norm minimization optimization problem. In a noiseless scenario, the sparse reconstruction can be expressed as

$$\min_{\mathbf{x}} \quad \|\mathbf{x}\|_0 \quad \text{s.t.} \quad \mathbf{y} = \mathbf{\Phi x}. \tag{2.15}$$

For noisy scenarios, the sparse reconstruction is

$$\min_{\mathbf{x}} \quad \|\mathbf{x}\|_0 \quad \text{s.t.} \quad \|\mathbf{y} - \mathbf{\Phi x}\|_2^2 \leq \epsilon \tag{2.16}$$

where ϵ is a small constant related to the noise level. Both the ℓ_0-norm minimization optimization problems in (2.15) and (2.16) are NP-hard [72]. Approximation solutions are often found by two methods: the greedy search and the convex relaxation methods. The greedy search method tries to

approximate the exact solution while the convex relaxation method tries to solve an approximate problem exactly. Common greedy search algorithms include the OMP [44], CoSaMP [46], and the LARS [47]. The most popular convex relaxation method is the ℓ_1-norm relaxation that replaces the ℓ_0-norm constraint using the convex ℓ_1-norm constraint to approximate the original optimization problem.

An important result in the compressive sensing theory reported that a sparse signal with $|\text{supp}(\mathbf{x})| \leq K$ can be recovered exactly from the noiseless measurements $\mathbf{y} = \mathbf{\Phi x}$ when the measurement matrix $\mathbf{\Phi}$ satisfies the RIP with $\delta_{2K} \leq 1$ by solving the following ℓ_1-norm minimization problem [72, 74],

$$\min_{\mathbf{x}} \quad \|\mathbf{x}\|_1 \quad \text{s.t.} \quad \mathbf{y} = \mathbf{\Phi x}. \tag{2.17}$$

The problem (2.17) can be reformulated into a linear programming (LP) problem, which is a class of well-studied optimization problems and can be conveniently solved using various solvers. We first define the intermediate nonnegative variables $\mathbf{u} = (\mathbf{x})_+$ and $\mathbf{v} = (-\mathbf{x})_+$ such that $\mathbf{x} = \mathbf{u} - \mathbf{v}$, where $(\cdot)_+$ denotes the positive-retaining operation that sets the negative elements to zeros. Then we denote the nonnegative vector by $\mathbf{z} = [\mathbf{u}^T, \mathbf{v}^T]^T$, such that the problem (2.17) can be rewritten as

$$\min_{\mathbf{z}} \quad \sum_{i=1}^{2N} z_i \quad \text{s.t.} \quad \mathbf{y} = [\mathbf{\Phi}, -\mathbf{\Phi}]\mathbf{z}, \quad \mathbf{z} \geq \mathbf{0} \tag{2.18}$$

where z_i denotes the ith element of the vector $\mathbf{z}$.

When measurements are contaminated by some noise, the recovery is still proven to be stable given the noise is small and the measurement matrix satisfies the RIP [72, 73]. By the ℓ_1-norm relaxation, the sparse reconstruction for noisy scenarios can be expressed as

$$\min_{\mathbf{x}} \quad \|\mathbf{x}\|_1 \quad \text{s.t.} \quad \|\mathbf{y} - \mathbf{\Phi x}\|_2^2 \leq \epsilon. \tag{2.19}$$

The problem (2.19) can be rewritten as

$$\begin{aligned} \min_{\mathbf{x}} \quad & \|\mathbf{y} - \mathbf{\Phi}\mathbf{x}\|_2^2 \\ \text{s.t.} \quad & \|\mathbf{x}\|_1 \leq t \end{aligned} \tag{2.20}$$

where t is a nonnegative real parameter that controls the sparsity level of the reconstruction vector. Using an appropriate multiplier $\lambda \geq 0$, problem (2.20) can be rewritten in an equivalent form as

$$\min_{\mathbf{x}} \quad \frac{1}{2}\|\mathbf{y} - \mathbf{\Phi}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_1 \tag{2.21}$$

where $\lambda \geq 0$ is a penalty parameter to balance the data-fidelity and regularization; it is important to note that the penalty parameter λ is challenging to tune. Since the ℓ_1-regularizer $\lambda\|\mathbf{x}\|_1$ forces equal penalties on each element of $\mathbf{x}$, a larger λ will force a sparser solution $\hat{\mathbf{x}}$ but can increase the residual error $\|\mathbf{y} - \mathbf{\Phi}\mathbf{x}\|_2^2$, whereas a small λ will lead to a solution that is insufficiently sparse. Cross-validation is a common approach to find a proper value of λ, but it is computationally intensive for large-scale problems having an $\mathbf{x}$ vector with a high dimension. Moreover, it is impractical to select a proper parameter by cross-validation due to a large number of distinct instances. The problem (2.21) is known as the Lasso [3] regression in statistical literature or machine learning while it is referred to as basis pursuit denoising in signal processing. The ℓ_1-norm term in (2.21) is not differentiable at the origin, but a variety of convex optimization techniques can be used to compute the solutions, such as the coordinate descent [77], subgradient method [78], and proximal gradient methods. For example, the well-known ISTA and FISTA algorithms were developed by applying the proximal gradient methods on (2.21). It is worth noting that problem (2.21) can be regarded as the maximum a posteriori (MAP) estimator for the vector $\mathbf{x}$ in the linear model $\mathbf{y} = \mathbf{\Phi}\mathbf{x} + \mathbf{n}$ under the assumption that the measurements follow a Gaussian distribution and the estimating parameters follow a Laplace distribution.

Now, we consider a more general problem setting, i.e. a class of regular-

[3] Lasso refers to least absolute shrinkage and selection operator.

ization problems where the objective function can be described as the sum of a loss function and a regularizer,

$$\min_{\mathbf{x}} L(\mathbf{\Theta}, \mathbf{x}) + \lambda R(\mathbf{x}) \tag{2.22}$$

where $L(\cdot)$ and $R(\cdot)$ denotes the loss function and the regularizer and the $\mathbf{\Theta}$ represents the observed data and the known parameters. For a sparse reconstruction problem where $\mathbf{\Theta} = (\mathbf{\Phi}, \mathbf{y})$, the negative log-likelihood function for the linear Gaussian distributed measurements $\mathbf{y}$ is

$$\begin{aligned} L(\mathbf{y}, \mathbf{\Phi}, \mathbf{x}) &= -\log P(\mathbf{y}|\mathbf{\Phi}\mathbf{x}) - \log P(\mathbf{\Phi}) \\ &= -\log \prod_{i=1}^{M} P(y_i|\boldsymbol{\phi}_i^T\mathbf{x}) - \log P(\mathbf{\Phi}) \\ &= \frac{1}{2}\sum_{i=1}^{M}(y_i - \boldsymbol{\phi}_i^T\mathbf{x})^2 + C \\ &= \frac{1}{2}\|\mathbf{y} - \mathbf{\Phi}\mathbf{x}\|_2^2 + C \end{aligned} \tag{2.23}$$

where $C = \log\sqrt{2\pi} - \log P(\mathbf{\Phi})$ is a constant that does not depend on $\mathbf{x}$. Given that the elements in $\mathbf{x}$ follow a Laplace distribution with the parameter λ, i.e., $p(x_i) = \lambda/2e^{-\lambda|x_i|}$, the regularizer can be obtained by calculating the negative log-likelihood,

$$\begin{aligned} R(\mathbf{x}, \lambda) &= -\log P(\mathbf{x}|\lambda) \\ &= -\log \prod_{i=1}^{N} P(x_i|\lambda) \\ &= \lambda\sum_{i}^{N}|x_i| + C' \\ &= \|\mathbf{x}\|_1 + C' \end{aligned} \tag{2.24}$$

where the constant $C' = -\log(\lambda/2)$ does not depend on $\mathbf{x}$. It is worth mentioning that the loss function and the regularizer terms in (2.22) can be replaced with other functions to accommodate other kinds of assumptions

regarding the model and its estimating parameters. For example, when the regularizer in (2.21) is replaced by the ℓ_2-norm, the Lasso regression problem becomes the Ridge regression problem, i.e., $\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{y}-\mathbf{\Phi x}\|_2^2 + \lambda\|\mathbf{x}\|_2^2$. More generally, with the prior $P(\mathbf{x}|\lambda, q) = C(\lambda, q)e^{-\lambda\|\mathbf{x}\|_q^q}$, we have a general q-norm regularized regression problem $\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{y}-\mathbf{\Phi x}\|_2^2 + \lambda\|\mathbf{x}\|_q^q$. When we extend the standard linear regression model to the generalized linear model[4] with exponential-family noise, the loss function of the maximum-likelihood parameter estimation becomes the Bregman divergence [72].

2.2.3 Compressive Sensing-Based Sparse Channel Estimation for Narrowband Massive MIMO Systems

For a narrowband block-fading massive MIMO channel [5], the beamspace channel can be represented by $\mathbf{H}_a = \mathbf{U}_r^H \mathbf{H}_c \mathbf{U}_t$, as shown in (2.11). Transmitting the known pilots $\mathbf{P} \in \mathbb{C}^{N_t \times M}$ from the BS through the channel, the pilot signal received at UE is

$$\mathbf{R} = \mathbf{H}_c\mathbf{P} + \mathbf{W} \tag{2.25}$$

where $\mathbf{R} \in \mathbb{C}^{N_r \times M}$ is the received pilot observations, and where M is the length of training pilot sequence for each antenna; $\mathbf{H}_c \in \mathbb{C}^{N_r \times N_t}$ is the channel response matrix, $\mathbf{P}$ is the transmitted pilot matrix; $\mathbf{W} \in \mathbb{C}^{N_r \times M}$ is the additive white Gaussian noise matrix whose elements are independent identical distributed (i.i.d.) complex Gaussian random variables having a mean of zero and a variance of σ_n^2. Conventional estimation methods such as the linear minimum mean square error estimation (LMMSE) or the LS

[4]The linear model $y = \boldsymbol{\phi}^T\mathbf{x} + \epsilon$ assumes the mean of the observation y is linear with the model parameters $\mathbf{x}$, i.e., $\mathbb{E}[y] = \phi^T\mathbf{x}$, while the generalized linear model assumes the the mean of the observation y is a function, which is usually nonlinear, with the model parameters $\mathbf{x}$, i.e., $\mathbb{E}[y] = f(\phi^T\mathbf{x})$

[5]In this thesis, the proposed approaches are applied to narrowband block-fading channels of massive MIMO. It is worth mentioning that our proposed approaches can be conveniently applied to other types of sparse channels as long as the channel has sparse representations in certain domains. For example, the proposed approaches can be applied to channel estimation of wideband time-varying channels for massive MIMO systems. The only required modification is to exploit the channel sparsity in a virtual angular-delay-Doppler domain.

estimation require $M \geq N_t$ to obtain accurate estimates, whereas the sparse channel estimation is expected to reconstruct $\hat{\mathbf{H}}_c$ for $M \ll N_t$. From (2.11), we have $\mathbf{H}_c = \mathbf{U}_r \mathbf{H}_a \mathbf{U}_t^H$, which can be substituted in (2.25) to obtain

$$\mathbf{R} = \mathbf{U}_r \mathbf{H}_a \mathbf{U}_t^H \mathbf{P} + \mathbf{W}. \tag{2.26}$$

By taking the transposes and right multiplications with $\mathbf{U}_r^H$ on both sides of (2.26), we have

$$\underbrace{\mathbf{R}^T \mathbf{U}_r^H}_{\tilde{\mathbf{R}}} = \underbrace{\mathbf{P}^T \mathbf{U}_t^*}_{\dot{\mathbf{\Phi}}} \underbrace{\mathbf{H}_a^T \mathbf{U}_r \mathbf{U}_r^H}_{\mathbf{H}} + \underbrace{\mathbf{W}^T \mathbf{U}_r^H}_{\tilde{\mathbf{W}}} \tag{2.27}$$

We simply write (2.27) as

$$\tilde{\mathbf{R}} = \dot{\mathbf{\Phi}} \mathbf{H} + \tilde{\mathbf{W}} \tag{2.28}$$

where $\tilde{\mathbf{R}} = \mathbf{R}^T \mathbf{U}_r^H \in \mathbb{C}^{M \times N_r}$, $\mathbf{H} = \mathbf{H}_a^T \mathbf{U}_r \mathbf{U}_r^H \in \mathbb{C}^{N_t \times N_r}$, $\tilde{\mathbf{W}} \in \mathbb{C}^{M \times N_r}$, and the matrix $\dot{\mathbf{\Phi}} = \mathbf{P}^T \mathbf{U}_t^* \in \mathbb{C}^{M \times N_t}$. The beamspace channel estimation $\hat{\mathbf{H}} = \hat{\mathbf{H}}_a^T$ can be obtained by solving the linear equation (2.28) given the known measurement matrix $\dot{\mathbf{\Phi}}$ and measurements $\tilde{\mathbf{R}}$. To this end, both an efficient sparse recovery algorithm and a proper measurement matrix are essential to the estimation performance. Eq. (2.27) suggests that the pilot matrix $\mathbf{P}$ depends on the measurement matrix $\dot{\mathbf{\Phi}}$. To design the pilot matrix, we can first determine a measurement matrix $\dot{\mathbf{\Phi}}$, and then obtain the pilot matrix by $\mathbf{P} = \dot{\mathbf{\Phi}}^T \mathbf{U}_t$, where $\mathbf{U}_t$ is a DFT matrix. Due to $\|\dot{\mathbf{\Phi}}\|_F^2 = \|\mathbf{P}^T \mathbf{U}_t^*\|_F^2$, the power constraint on a pilot matrix $\|\mathbf{P}\|_F^2 = \mathcal{P}$ can be imposed via scaling the measurement matrix by $\dot{\mathbf{\Phi}} = \sqrt{\mathcal{P}} \frac{\dot{\mathbf{\Phi}}'}{\|\dot{\mathbf{\Phi}}'\|_F}$, where $\dot{\mathbf{\Phi}}'$ is the unscaled measurement matrix, and $\|\cdot\|_F$ represents the Frobenius norm of a matrix.

To solve the multiple measurement vector (MMV) problem (2.28), we propose the column-wise broadcasting vectorization method to vectorize the matrix $\mathbf{H}$. Thus, the MMV problem (2.28) can be transformed into parallel single measurement vector (SMV) problems. Therefore, although the proposed algorithms are developed for an SMV model, they can be extended to

the MMV problem. We express the linear equation (2.28) as

$$[\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_{N_r}] \quad = \dot{\mathbf{\Phi}}[\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_{N_r}] + [\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_{N_r}] \tag{2.29}$$

where $\mathbf{r}_i$ is the ith ($1 \leq i \leq N_r$) column of $\tilde{\mathbf{R}}$; $\mathbf{h}_i$ is the ith ($1 \leq i \leq N_r$) column of $\mathbf{H}$; $\mathbf{w}_i$ is the ith ($1 \leq i \leq N_r$) column of $\tilde{\mathbf{W}}$. For the ith column of $\tilde{\mathbf{R}}, \mathbf{H}$ and $\tilde{\mathbf{W}}$ in (2.28), we have

$$\mathbf{r}_i = \dot{\mathbf{\Phi}}\mathbf{h}_i + \mathbf{w}_i. \tag{2.30}$$

We can further express (2.30) by an equivalent real-valued linear equation

$$\underbrace{\begin{bmatrix} \Re(\mathbf{r}_i) \\ \Im(\mathbf{r}_i) \end{bmatrix}}_{\mathbf{y}} = \underbrace{\begin{bmatrix} \Re(\dot{\mathbf{\Phi}}) & -\Im(\dot{\mathbf{\Phi}}) \\ \Im(\dot{\mathbf{\Phi}}) & \Re(\dot{\mathbf{\Phi}}) \end{bmatrix}}_{\mathbf{\Phi}} \underbrace{\begin{bmatrix} \Re(\mathbf{h}_i) \\ \Im(\mathbf{h}_i) \end{bmatrix}}_{\mathbf{x}} + \underbrace{\begin{bmatrix} \Re(\mathbf{w}_i) \\ \Im(\mathbf{w}_i) \end{bmatrix}}_{\mathbf{n}} \tag{2.31}$$

For the presentational simplicity, we concisely express (2.31) as

$$\mathbf{y} = \mathbf{\Phi}\mathbf{x} + \mathbf{n}. \tag{2.32}$$

Thus, we formulate the sparse channel estimation problem as an SMV problem in (2.32), which can be solved by sparse reconstruction algorithms.

2.3 Deep Learning for Sparse Channel Estimation

In the last section, we have introduced the sparse channel estimation using the compressive sensing technology. The compressive sensing-based channel estimation approach is a model-based method and has well-defined system models and statistical assumptions. For example, the received pilots have a linear relationship with the estimating channels, the massive MIMO channel has virtual sparse representations and the noise is assumed to follow a Gaussian distribution. Under these assumptions, the massive MIMO channel estimation has been formulated as a sparse reconstruction

problem. To solve the sparse reconstruction problem, we have introduced the Lasso regression optimization, where the solution can be interpreted as a MAP estimator for a linear regression assuming the Gaussian noise and Laplace-distributed parameters. In practice, the simple linear model and statistical assumptions cannot describe the real physical process accurately. Therefore, the data-driven approaches based on state-of-the-art deep learning techniques becomes increasingly appealing.

2.3.1 Data-Driven Channel Estimation

In contrast to a model-driven approach using explicit model and statistical assumptions, the data-driven approach based on deep learning can deal with the situations where the underlying physical process is too complicated to formulate mathematically and where the estimating parameters have unknown statistics. The data-driven channel estimation pipeline is shown in Fig. 2.4. The deep learning-based approach uses a large amount of data to train a deep learning network to "learn" the underlying physical process and statistics in the data so that the trained neural network has the ability to make accurate predictions when supplied with new data. We group the data-driven methods into two categories: the generic deep learning method and the model-based deep learning method. The generic deep learning method uses generic deep learning networks such as the common DNN, CNN, and RNN. Generic deep networks have some drawbacks, such as low interpretability and often require a large training data set. Moreover, when there is insufficient training data available, overfitting often occurs, such that the generic deep networks have a low generalization ability. Unlike the generic deep learning method, the model-based deep learning perfectly combines and balances the domain knowledge from the model-based method and data-driven ability from the deep learning. For example, the deep unfolding technique can build small-sized, interpretable model-based deep networks based on the established model-based algorithms. In the following discussions, we will introduce the DNN and learned ISTA (LISTA) networks as examples for the generic deep learning method and the model-based deep

learning method.

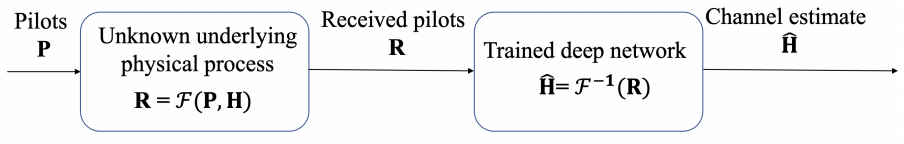


Figure 2.4: Data-driven pilot training channel estimation pipeline

2.3.2 Generic Deep Learning Networks

A DNN is an artificial neural network consisting of multiple differentiable computational layers between the input and output layers. Each intermediate layer has forward-pass and backward-pass computations. The forward-pass computation of a layer takes the input $\mathbf{x}$ and produces an output $\mathbf{y}^{(k)} = q^{(k)}(\mathbf{x})$, where $q^{(k)}(\cdot)$ denotes the computational operations of the kth layer. The backward-pass computations are also called the backpropagation, which is the basic training technique to optimize the neural network parameters. Each layer of the neural network has trainable and non-trainable parameters. The trainable parameters are also called the weights which are optimized to minimize the loss function during training phase by backpropagation, while the non-trainable parameters are called hyper-parameters which need to be defined explicitly and manually.

The forward-pass of an L-layer DNN architecture can be denoted by $\mathbf{y} = \mathcal{A}_{\mathbf{\Theta}}(\mathbf{x})$, where $\mathbf{\Theta}$ denotes the trainable parameters of the network. For a DNN $\mathcal{A}_{\mathbf{\Theta}}(\cdot)$, given the training dataset $\mathcal{D} = \{\mathbf{x}_i, \mathbf{y}_i | i = 0, ..., n-1\}$, where $\mathbf{x}_i$ denotes the ith input samples and $\mathbf{y}_i$ denotes the corresponding target, training the DNN based on the dataset is essentially solving the following optimization problem:

$$\begin{aligned} \min_{\mathbf{\Theta}} \quad & \sum_{i=0}^{n-1} \mathcal{L}(\hat{\mathbf{y}}_i, \mathbf{y}_i) \\ \text{s.t.} \quad & \hat{\mathbf{y}}_i = \mathcal{A}_{\mathbf{\Theta}}(\mathbf{x}_i), i = 0, ..., n-1. \end{aligned} \tag{2.33}$$

The training process usually adopts the first-order mini-batch stochastic gradient descent as the optimization algorithm. Meanwhile, a variety of optimization techniques such as the momentum [79, 80] or the gradient regularization [80, 81] can be incorporated to accelerate the convergence. During training, the gradients of the loss function with respect to the weights are computed using the chain rule and back propagation through the networks. The input of the backward-pass computation for a layer is the gradient of the training loss with respect to the output of a layer. Various loss functions can be used, such as the MSE, cross entropy, and softmax functions. In particular, the MSE loss function is defined as

$$\mathcal{L}_{MSE} = \frac{1}{M}\sum_{m=0}^{M-1}(\hat{y}_m - y_m)^2 \tag{2.34}$$

where y_m denotes the mth element of the M dimensional target $\mathbf{y}$.

The simplest DNN architecture is the feedforward neural network that stacks multiple layers of linear affine transformations with nonlinear activation functions. The forward-pass computation of the kth layer can be expressed as

$$q^{(k)}(\mathbf{x}^{(k-1)}) = \sigma(\mathbf{W}^k\mathbf{x}^{(k-1)} + \mathbf{b}^{(k)}) \tag{2.35}$$

and the input of the network is $q^{(0)} = \mathbf{x}$; the $\sigma(\cdot)$ denotes the nonlinear activation function which performs elementwise on its input. A DNN architecture can be represented as $\mathcal{A}_\Theta(\mathbf{x}) = q^{(L)}(q^{(L-1)}...(q^{(0)}(\mathbf{x})))$, where $\Theta = \{\mathbf{W}_i, \mathbf{b}_i | i = 1, ..., L\}$ and L is the number of network layers. The most popular activation function is the rectified linear unit (ReLU) function defined as

$$\text{ReLU}(x) = \max(0, x). \tag{2.36}$$

Other common activation functions include the sigmoid function (also known as the logistic function) and the hyperbolic tangent function $\tanh(\cdot)$.

2.3.3 Deep Unfolding for Model-Based Deep Learning

Deep unfolding is a popular way to incorporate the domain-knowledge into a deep network architecture and develop partial data-driven approaches. Deep unfolding or deep unrolling, which was proposed by Hershey *et al.* [64], builds the deep networks by implementing the iterative algorithms derived from classical statistical models as the differential computational layers. A high-level overview of an unfolding network is illustrated in Fig. 2.5. The built deep network is essentially a stacking of unfolded iterations. The number of iterations is usually fixed and the trainable weights for each layer can be nominated from the parameters of the algorithms. The training of the unfolded network adopts the backpropagation mechanism and stochastic gradient descent optimization algorithm, which are the same as in generic neural networks.

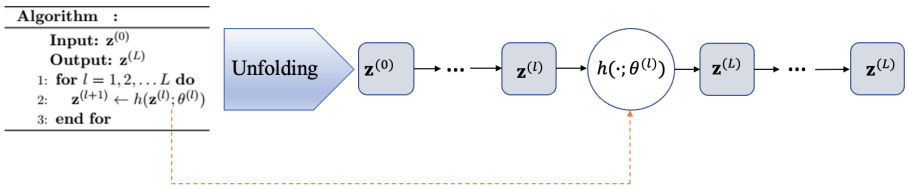


Figure 2.5: Unfolding an iterative algorithm into a deep network [1]

Several sparse recovery algorithms have been unfolded. For example, the ISTA [48] and AMP [65] have been unfolded into the learned algorithms as LISTA [82] and LDAMP [66]. In the following discussion, we introduce the LISTA as an example. The LISTA is the earliest unfolding deep network proposed by Gregor and LeCun [82], and it is the built on the classical sparse recovery algorithm ISTA. The following derivation of ISTA was originally presented in [51] and [82].

The ISTA algorithm can be derived by performing the proximal gradient descent on the sparse reconstruction problem in (2.21). Here, we denote the objective function as $f(\mathbf{x}) = L(\mathbf{x}) + \lambda R(\mathbf{x})$, where $L(\mathbf{x}) = \frac{1}{2}\|\mathbf{y} - \mathbf{\Phi x}\|_2^2$ and $\lambda R(\mathbf{x}) = \lambda\|\mathbf{x}\|_1$. The tth iteration solves the quadratic approximation of

$f(\mathbf{x})$, that is

$$\begin{aligned}\mathbf{x}^{(t+1)} &= \min_{\mathbf{x}} L(\mathbf{x}^{(t)}) + \Delta L(\mathbf{x}^{(t)})^T(\mathbf{x}-\mathbf{x}^{(t)}) + \frac{1}{2\alpha_t}\|\mathbf{x}-\mathbf{x}^{(t)}\|_2^2 + \lambda R(\mathbf{x}) \\ &= \min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x} - (\mathbf{x}^{(t)} - \alpha_t \Delta L(\mathbf{x}^{(t)})^T)\|_2^2 + \alpha_t \lambda R(\mathbf{x}).\end{aligned} \tag{2.37}$$

The solution is the proximal gradient algorithm

$$\mathbf{x}^{(t+1)} = \text{prox}_{\alpha_t, R}\big(\mathbf{x}^{(t)} - \alpha_t \Delta L(\mathbf{x}^{(t)})^T\big) \tag{2.38}$$

where the $\text{prox}_{\alpha_t,R}$ is referred to as the proximal operator with respect to $\alpha_t, R(\cdot)$. Generally, the proximal operator on $\mathbf{z}$ with respect to $R(\cdot)$ is the solution to

$$\text{prox}_R(\mathbf{z}) = \min_{\mathbf{x}} \frac{1}{2}\|\mathbf{x}-\mathbf{z}\|_2^2 + R(\mathbf{x}). \tag{2.39}$$

For $R(\mathbf{x}) = \alpha_t\lambda\|\mathbf{x}\|_1$, the proximal operator $\text{prox}_R(\mathbf{z})$ is the soft threshold $\mathcal{S}_{\alpha_t,\lambda}(\mathbf{z})$, and one applies the following operation to each element z_i of $\mathbf{z}$,

$$z_i \leftarrow \text{sign}(z_i)\max\{0, |z_i| - \alpha_t\lambda\}. \tag{2.40}$$

With $\mathbf{z} = \mathbf{x}^{(t)} - \alpha_t \Delta L(\mathbf{x}^{(t)})^T$, and $\Delta L(\mathbf{x}^{(t)}) = -\mathbf{\Phi}^T(\mathbf{y} - \mathbf{\Phi}\mathbf{x}^{(t)})$, we obtain the ISTA iteration,

$$\begin{aligned}\mathbf{x}^{(t+1)} &= \mathcal{S}_{\alpha_t\lambda}\big(\mathbf{x}^{(t)} + \alpha_t\mathbf{\Phi}^T(\mathbf{y} - \mathbf{\Phi}\mathbf{x}^{(t)})\big) \\ &= \mathcal{S}_{\alpha_t\lambda}\big((\mathbf{I} - \alpha_t\mathbf{\Phi}^T\mathbf{\Phi})\mathbf{x}^{(t)} + \alpha_t\mathbf{\Phi}^T\mathbf{y}\big).\end{aligned} \tag{2.41}$$

Based on the derived ISTA iteration in (2.41), the corresponding ISTA algorithm and the unfolded network structure of LISTA are shown in Fig. 2.6.

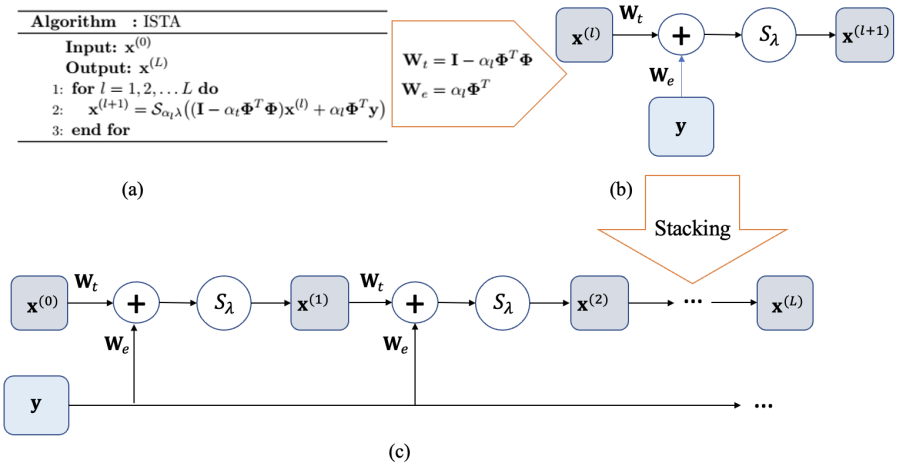


Figure 2.6: LISTA: deep unfolding of the ISTA [1]

Chapter 3

Nonconvex Regularized Gradient Projection Algorithms for Sparse Channel Reconstruction

In this chapter, novel sparse reconstruction algorithms are proposed. We first formulate the sparse reconstruction problem as an optimization to minimize an LS objective having a nonconvex regularizer. This regularizer removes the penalties on a few large-magnitude elements from the conventional ℓ_1-norm regularizer (known as the Lasso regularizer). Then, we perform gradient projection updates within the DC programming framework to develop accurate and fast reconstructions. A double-loop algorithm and a single-loop algorithm are proposed via different DC decompositions, and these two algorithms have distinct computational complexities and convergence rates. Then, an extension algorithm is further proposed by designing new step sizes for the single-loop algorithm. The extension algorithm has a faster convergence rate and can achieve approximately the same level of accuracy as the proposed double-loop algorithm.

3.1 DC Representation for ℓ_0-norm Constraint

In Section 2.2.3, a massive MIMO sparse channel estimation was formulated into an SMV problem to solve the underdetermined linear system $\mathbf{y} = \mathbf{\Phi x} + \mathbf{n}$ in (2.32), where $\mathbf{x} = [\Re(\mathbf{h}_i)^T, \Im(\mathbf{h}_i)^T]^T$ represents an arbitrary

sparse channel vector $\mathbf{h}_i$ for $1 \leq i \leq N_r$. In this section, we introduce the top-$(K,1)$ norm [83] to reformulate the sparse reconstruction optimization problem into an LS minimization with a novel penalty term.

According to (2.32), reconstructing $\mathbf{x}$ from $\mathbf{y}$ and $\mathbf{\Phi}$ using the sparsity as a priori is an NP-hard ℓ_0-minimization problem [72], which is defined as

$$\begin{aligned} \min_{\mathbf{x}} \quad & \|\mathbf{x}\|_0 \\ \text{s.t.} \quad & \|\mathbf{y} - \mathbf{\Phi}\mathbf{x}\|_2^2 \leq \tau \end{aligned} \tag{3.1}$$

where τ is nonnegative and real. Problem (3.1) can be rewritten in an equivalent form as

$$\begin{aligned} \min_{\mathbf{x}} \quad & \|\mathbf{y} - \mathbf{\Phi}\mathbf{x}\|_2^2 \\ \text{s.t.} \quad & \|\mathbf{x}\|_0 \leq K \end{aligned} \tag{3.2}$$

where K is an upper bound for the number of nonzero elements in $\mathbf{x}$, and it is uniquely determined by the parameter τ in (3.1) [72].

Instead of using the common ℓ_1-relaxation, we introduce the top-$(K,1)$ norm to seek an equivalent expression for the constraint $\|\mathbf{x}\|_0 \leq K$ in the original problem (3.2). The top-$(K,1)$ norm $\|\mathbf{x}\|_{K,1}$ is defined as the sum of the K largest elements of the vector $\mathbf{x}$ in terms of the absolute values, namely

$$\|\mathbf{x}\|_{K,1} := |x_{(1)}| + |x_{(2)}| + \cdots + |x_{(K)}| \tag{3.3}$$

where $|x_{(i)}|$ denotes the element whose absolute value is the ith-largest among the N elements of the vector $\mathbf{x}$, i.e., $|x_{(1)}| \geq |x_{(2)}| \geq \cdots \geq |x_{(N)}|$. The constraint $\|\mathbf{x}\|_0 \leq K$ is equivalent to the statement that the $(K+1)$th-largest element of the vector $\mathbf{x}$ is zero, i.e., $\|\mathbf{x}\|_{K+1,1} - \|\mathbf{x}\|_{K,1} = 0$. Thus, we have an equivalent relationship between the following two statements [83]

$$\|\mathbf{x}\|_0 \leq K \Leftrightarrow \|\mathbf{x}\|_1 - \|\mathbf{x}\|_{K,1} = 0. \tag{3.4}$$

Since both $\|\mathbf{x}\|_1$ and $\|\mathbf{x}\|_{K,1}$ are convex, we say the equality $\|\mathbf{x}\|_1 - \|\mathbf{x}\|_{K,1} = 0$ is an exact DC representation for the sparsity constraint. By replacing the sparsity constraint $\|\mathbf{x}\|_0 \leq K$ in (3.2) using the DC constraint $\|\mathbf{x}\|_1 -$

$\|\mathbf{x}\|_{K,1} = 0$, we rewrite the sparse reconstruction problem as

$$\begin{aligned} \min_{\mathbf{x}} \quad & \|\mathbf{y} - \mathbf{\Phi}\mathbf{x}\|_2^2 \\ \text{s.t.} \quad & \|\mathbf{x}\|_1 - \|\mathbf{x}\|_{K,1} = 0. \end{aligned} \tag{3.5}$$

Using an appropriate Lagrange multiplier ρ, from (3.5) we obtain the following unconstrained optimization problem

$$\min_{\mathbf{x}} \quad \frac{1}{2}\|\mathbf{y} - \mathbf{\Phi}\mathbf{x}\|_2^2 + \rho(\|\mathbf{x}\|_1 - \|\mathbf{x}\|_{K,1}) := F(\mathbf{x}) \tag{3.6}$$

where ρ is the regularization parameter that balances the data consistency and the penalty term. Our formulated optimization problem (3.6) differs from the conventional ℓ_1-regularized sparse reconstruction [6] only in terms of the subtracted top-$(K, 1)$ norm $\|\mathbf{x}\|_{K,1}$ in its penalty term. The regularizer $\rho(\|\mathbf{x}\|_1 - \|\mathbf{x}\|_{K,1})$ is better than an ℓ_1-norm regularizer because it removes the penalties on the K largest-magnitude elements [7].

To ensure the equivalence between the unconstrained problem (3.6) and the constrained problem (3.5), the following theorem specifies the range for the penalty parameter.

Theorem 3.1. *Let $\mathbf{x}_{\rho^*}$ be an optimal solution to* (3.6) *with given ρ^*. Suppose there exists a constant $q > 0$ such that $\|\mathbf{x}_{\rho^*}\|_2 \leq q$ for any $\rho^* > 0$. Then $\mathbf{x}_{\rho^*}$ is also optimal to* (3.5) *if*

$$\rho^* \geq \max_i \{q(\|\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{e}_i\|_2 + |(\mathbf{\Phi}^T\mathbf{\Phi})_{ii}|/2) + |(\mathbf{\Phi}^T\mathbf{y})_i|\}$$

where $1 \leq i \leq N$; $\mathbf{e}_i$ represents the unit vector in which the ith element is one while the other elements are zeros; $(\mathbf{\Phi}^T\mathbf{\Phi})_{ii}$ represents the ith diagonal elements of matrix $\mathbf{\Phi}^T\mathbf{\Phi}$; $(\mathbf{\Phi}^T\mathbf{y})_i$ indicates the ith element of the vector $\mathbf{\Phi}^T\mathbf{y}$.

Proof: See Appendix A.

[6] $\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{y} - \mathbf{\Phi}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_1$

[7] In practice, if the sparsity level of reconstructing vector is already known, we set K as the number of nonzero elements of reconstruction vectors; otherwise, we can use cross validation to determine an approximation of K.

Theorem 1 indicates that given a penalty parameter ρ having a suitably large value, the optimal solution to (3.6) is also the optimal solution to (3.5). To calculate the lower threshold ρ^*, we can first estimate a constant q such that $\|\mathbf{x}_{\rho^*}\|_2 \leq q$. In practice, to avoid the computationally intensive inequality in Theorem 1, we can use cross validation to select a suitable value for the penalty parameter ρ.

3.2 Double-Loop DC Gradient Projection Descent for Sparse Reconstruction

We have formulated the sparse reconstructions into a nonconvex optimization problem (3.6). In this section, we use DC programming and gradient projection descent to solve (3.6) and propose a double-loop DC-GPSR algorithm.

3.2.1 DC Programming Framework

For a nonconvex unconstrained optimization problem

$$\min_{\mathbf{x}} f(\mathbf{x}) - g(\mathbf{x}) \tag{3.7}$$

where $f(\mathbf{x})$ and $g(\mathbf{x})$ are two convex functions. The DC programming method solves the following convex subproblem at the tth-iteration,

$$\min_{\mathbf{x}} f(\mathbf{x}) - \mathbf{x}^T \partial g(\mathbf{x}^{t-1}) \tag{3.8}$$

where the second convex function $g(\mathbf{x})$ in (3.7) is linearized by $\mathbf{x}^T \partial g(\mathbf{x}^{t-1})$ in (3.8), and where $\partial g(\mathbf{x}^{t-1})$ represents the gradient (or subgradient) of $g(\mathbf{x}^{t-1})$ with respect to $\mathbf{x}^{t-1}$. The DC algorithm framework can be outlined as follows:

1. **Start:** Given a starting point $\mathbf{x}^0$, and a terminate condition
2. **Repeat:** For $t = 1, 2, ...$
   Compute the gradient (or subgradient) $\partial g(\mathbf{x}^{t-1})$.

Solve the convex subproblem (3.8) to obtain $\mathbf{x}^t$.

3. **End:** Until a terminate condition is satisfied.

The DC programming is an iterative algorithm framework that can ensure global convergence, which means the DC algorithm can converge from an arbitrary initial point [8].

3.2.2 A Double-Loop DC-GPSR Algorithm

Following the DC programming framework, we decompose our objective function in (3.6) as the difference of the two convex functions of $f(\mathbf{x})$ and $g(\mathbf{x})$

$$\min_{\mathbf{x}} \quad \underbrace{\frac{1}{2}\|\mathbf{y}-\mathbf{\Phi}\mathbf{x}\|_2^2+\rho\|\mathbf{x}\|_1}_{f(\mathbf{x})} - \underbrace{\rho\|\mathbf{x}\|_{K,1}}_{g(\mathbf{x})}. \tag{3.9}$$

At the tth-iteration, we solve the following convex subproblem

$$\min_{\mathbf{x}} \quad f(\mathbf{x}) - \rho\mathbf{x}^T\partial\|\mathbf{x}^{t-1}\|_{K,1} \tag{3.10}$$

where $\partial\|\mathbf{x}^{t-1}\|_{K,1}$ denotes the subgradient of top-$(K,1)$ norm of $\mathbf{x}^{t-1}$, and where the superscript $t-1$ indicates the $(t-1)$th update. The subgradient $\partial\|\mathbf{x}\|_{K,1}$ of the top-$(K,1)$ norm of $\mathbf{x}$ is defined as [83]

$$\partial\|\mathbf{x}\|_{K,1} := \underset{\mathbf{w}}{\operatorname{argmax}} \left\{ \sum_{i=1}^{N} x_i w_i \Big| \sum_{i=1}^{N} |w_i| = K, w_i \in [-1,1] \right\}. \tag{3.11}$$

By substituting the $f(\mathbf{x})$ defined in (3.9) into (3.10) and denoting $\mathbf{w}_x^{t-1}$ by a feasible value for the subgradient $\partial\|\mathbf{x}^{t-1}\|_{K,1}$, we write the subproblem (3.10) as

$$\min_{\mathbf{x}} \quad \frac{1}{2}\|\mathbf{y}-\mathbf{\Phi}\mathbf{x}\|_2^2+\rho\|\mathbf{x}\|_1 - \rho\mathbf{x}^T\mathbf{w}_x^{t-1} \tag{3.12}$$

where $\mathbf{w}_x^{t-1} \in \partial\|\mathbf{x}^{t-1}\|_{K,1}$. A feasible subgradient $\mathbf{w}_x^{t-1}$ can be simply obtained by setting the signs of the K largest elements of $|\mathbf{x}^{t-1}|$ to the corre-

[8] The global convergence property has been comprehensively studied for general DC algorithms, hence it is also valid for the proposed algorithms. Further details can be found in the following references: [84, Sec. 3.2], [85, Sec. 2], [86, p. 10], [87, pp. 5-8].

sponding elements of $\mathbf{w}_x^{t-1}$, i.e., $(\mathbf{w}_x)_{(i)}^{t-1} = \text{sign}(x_{(i)}^{t-1})$, where the subscript i indicates the ith element of a vector, and setting the other elements of $\mathbf{w}_x^{t-1}$ to be zeros.

We obtain a convex subproblem (3.12). We will turn (3.12) into a constrained quadratic problem so that we can solve it using the gradient projection descent method. Specifically, we split the positive and negative part of $\mathbf{x}$, and represent $\mathbf{x}$ as the difference of its positive part $\mathbf{u}$ and its negative part $\mathbf{v}$, that is

$$\mathbf{x} = \mathbf{u} - \mathbf{v}, \quad \mathbf{u} \succeq \mathbf{0}, \mathbf{v} \succeq \mathbf{0}. \tag{3.13}$$

where $\mathbf{u} = (\mathbf{x})_+, \mathbf{v} = (-\mathbf{x})_+$, and where $(\cdot)_+$ is the positive-taking operation that retains the positive elements and sets the other elements to be zeros. More precisely, $(\mathbf{x})_+$ represents the operation of $(x)_+ = \max\{0, x\}$ for each element x in vector $\mathbf{x}$; $(-\mathbf{x})_+$ represents the operation of $(-x)_+ = \max\{0, -x\}$ for each element $-x$ in vector $-\mathbf{x}$. Noticing that $\|\mathbf{x}\|_1 = \mathbf{1}^T\mathbf{u} + \mathbf{1}^T\mathbf{v}$, the subproblem (3.12) can be written as a bound constrained quadratic problem (BCQP) problem

$$\begin{aligned} \min_{\mathbf{u},\mathbf{v}} \quad & \frac{1}{2}\|\mathbf{y} - \mathbf{\Phi}(\mathbf{u} - \mathbf{v})\|_2^2 + \rho\mathbf{1}^T\mathbf{u} + \rho\mathbf{1}^T\mathbf{v} \\ & - \rho\mathbf{u}^T\mathbf{w}_u^{t-1} - \rho\mathbf{v}^T\mathbf{w}_v^{t-1} \\ \text{s.t.} \quad & \mathbf{u} \succeq \mathbf{0}, \mathbf{v} \succeq \mathbf{0} \end{aligned} \tag{3.14}$$

where $\mathbf{w}_u^{t-1}$ and $\mathbf{w}_v^{t-1}$, respectively, represent the positive and negative part of $\mathbf{w}_x^{t-1}$, i.e., $\mathbf{w}_u^{t-1} = (\mathbf{w}_x^{t-1})_+$, $\mathbf{w}_v^{t-1} = (-\mathbf{w}_x^{t-1})_+$. Let $\mathbf{z}$ denote the concatenation of $\mathbf{u}$ and $\mathbf{v}$, i.e., $\mathbf{z} = [\mathbf{u}^T, \mathbf{v}^T]^T$, Next, (3.14) can be rewritten into a compact form

$$\begin{aligned} \min_{\mathbf{z}} \quad & \frac{1}{2}\mathbf{z}^T\mathbf{B}\mathbf{z} + \mathbf{c}^T\mathbf{z} := G(\mathbf{z}), \\ \text{s.t.} \quad & \mathbf{z} \succeq \mathbf{0} \end{aligned} \tag{3.15}$$

where

$$\begin{aligned} \mathbf{z} &= \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} \mathbf{\Phi}^T\mathbf{\Phi} & -\mathbf{\Phi}^T\mathbf{\Phi} \\ -\mathbf{\Phi}^T\mathbf{\Phi} & \mathbf{\Phi}^T\mathbf{\Phi} \end{bmatrix}, \\ \mathbf{c} &= \begin{bmatrix} -\mathbf{\Phi}^T\mathbf{y} \\ \mathbf{\Phi}^T\mathbf{y} \end{bmatrix} + \rho\mathbf{1}^T - \rho\mathbf{w}_z^{t-1} \end{aligned} \tag{3.16}$$

and where $\mathbf{1}^T$ represents an all-ones column vector having the same dimension as $\mathbf{z}$, and $\mathbf{w}_z^{t-1} = [(\mathbf{w}_u^{t-1})^T, (\mathbf{w}_v^{t-1})^T]^T$. Note that $\mathbf{w}_z^{t-1}$ is a subgradient of $\|\mathbf{z}^{t-1}\|_{K,1}$, i.e., $\mathbf{w}_z^{t-1} \in \partial\|\mathbf{z}^{t-1}\|_{K,1}$. Since $\mathbf{z}^{t-1} \succeq \mathbf{0}$, a feasible subgradient $\mathbf{w}_z^{t-1}$ can be an indicator vector having either one-valued or zero-valued elements, where the indices for the one-valued elements of $\mathbf{w}_z^{t-1}$ correspond to the indices of the K largest elements of $\mathbf{z}^{t-1}$. Eq. (3.15) is an equivalent problem for the subproblem (3.12). We apply the gradient projection descent method to solve (3.15), thus the kth update is

$$\begin{aligned} \mathbf{z}^{(k+\frac{1}{2})} &= Proj\left(\mathbf{z}^{(k)} - \alpha^k \nabla G(\mathbf{z}^{(k)})\right), \\ \mathbf{z}^{(k+1)} &= \mathbf{z}^{(k)} + \beta^k(\mathbf{z}^{(k+\frac{1}{2})} - \mathbf{z}^{(k)}) \end{aligned} \tag{3.17}$$

where $\alpha^k > 0$ is the step size and it can be determined by the Barzilai-Borwein (BB) step size, which is calculated as $\alpha^t = \frac{\|\mathbf{z}^k - \mathbf{z}^{k-1}\|^2}{(\mathbf{z}^k - \mathbf{z}^{k-1})^T\left(G(\mathbf{z}^k) - G(\mathbf{z}^{k-1})\right)}$; $\beta^k \in (0, 1]$ is another step size to ensure the monotonic-decrease of the objective and it can be calculated in closed-form as $\beta^k = \frac{(\boldsymbol{\delta}^k)^T\nabla G(\mathbf{z}^k)}{(\boldsymbol{\delta}^k)^T\mathbf{B}\boldsymbol{\delta}^k}$, where $\boldsymbol{\delta}^k = \mathbf{z}^{(k+\frac{1}{2})} - \mathbf{z}^{(k)}$ [50]; $Proj(\cdot)$ represents the operation of orthogonal projection that projects the vector onto the nonnegative orthant [9]; $\nabla G(\mathbf{z}^{(k)})$ represents the gradient of $G(\mathbf{z})$ defined in (3.15) with respect to $\mathbf{z}^{(k)}$. We have $\nabla G(\mathbf{z}^{(k)}) = \mathbf{B}\mathbf{z} + \mathbf{c}$, which can be calculated by

$$\begin{aligned} \nabla G(\mathbf{z}^{(k)}) = &\begin{bmatrix} \mathbf{\Phi}^T\mathbf{\Phi}(\mathbf{u}^{(k)} - \mathbf{v}^{(k)}) \\ -\mathbf{\Phi}^T\mathbf{\Phi}(\mathbf{u}^{(k)} - \mathbf{v}^{(k)}) \end{bmatrix} + \begin{bmatrix} -\mathbf{\Phi}^T\mathbf{y} \\ \mathbf{\Phi}^T\mathbf{y} \end{bmatrix} \\ &- \rho\mathbf{w}_z^{t-1} + \rho\mathbf{1}^T. \end{aligned} \tag{3.18}$$

Essentially, the proposed algorithm computes the following two steps

[9] Let $\mathbb{R}_+^N = \{\mathbf{x} = (x_1, x_2, ..., x_N) | x_1 \geq 0, x_2 \geq 0, ..., x_N \geq 0\}$ be the nonnegative orthant of $\mathbb{R}^N$.

iteratively until convergence:

$$\begin{aligned} &\text{(a)} \quad \mathbf{w}_z^{t-1} \in \partial \|\mathbf{z}^{t-1}\|_{K,1} \\ &\text{(b)} \quad \mathbf{z}^t = \underset{\mathbf{z} \succeq 0}{\operatorname{argmin}} \{\frac{1}{2}\mathbf{z}^T\mathbf{B}\mathbf{z} + \mathbf{c}^T\mathbf{z}\} \end{aligned} \tag{3.19}$$

where $\mathbf{z}$, $\mathbf{B}$ and $\mathbf{c}$ are defined in (3.15). The step (b) in (3.19) is calculated by applying the gradient projection descent updates in (3.17). We summarize this double-loop DC-GPSR algorithm in Algorithm 1.

Algorithm 1: Double-loop DC-GPSR (DlDC-GPSR)

Input: measurements $\mathbf{y}$, measurement matrix $\mathbf{\Phi}$ and a small number ϵ

Output: reconstruction $\hat{\mathbf{x}}$

Initialization: $\mathbf{u}^0$, $\mathbf{v}^0$, $\mathbf{z}^0 \leftarrow [(\mathbf{u}^0)^T, (\mathbf{v}^0)^T]^T$

1: **for** $t = 1, 2, \ldots$ **do**
2: Compute a subgradient $\mathbf{w}_z^{t-1} \in \partial \|\mathbf{z}^{t-1}\|_{K,1}$.
3: **for** $k = 1, 2, \ldots$ **do**
4: Compute gradient $\nabla G(\mathbf{z}^{(k)})$ by (3.18).
5: Perform gradient projection descent (3.17) and obtain $\mathbf{z}^{(k+1)}$.
6: Check convergence, set $\mathbf{z}^* \leftarrow \mathbf{z}^{(k+1)}$ and proceed to Step 7 if convergence is satisfied; otherwise return to Step 3.
7: **end for**
8: $\mathbf{z}^t \leftarrow \mathbf{z}^*$
9: Check terminate condition $\|\mathbf{z}^t - \mathbf{z}^{t-1}\|_2 \leq \epsilon$ and return to Step 1 if not satisfied; otherwise, terminate with $\mathbf{z}^t = [(\mathbf{u}^t)^T, (\mathbf{v}^t)^T]^T$, and obtain the reconstruction $\hat{\mathbf{x}} = \mathbf{u}^t - \mathbf{v}^t$.
10: **end for**

3.3 Single-Loop DC Gradient Projection Descent for Sparse Reconstruction

In the proposed DlDC-GPSR algorithm, at each iteration we solve a nonsmooth convex subproblem (3.12) using another iterative algorithm, i.e., the gradient projection descent updates. This double-loop procedure can be

computationally inefficient. In this section, we derive a closed-form solution to the convex subproblem by performing a special DC decomposition to eliminate the inner iterations. Thus, we propose a basic single-loop DC-GPSR algorithm. Interestingly, we observe that the proposed basic single-loop DC-GPSR algorithm can be interpreted as simple gradient projection descent updates with a required step size. Furthermore, we accelerate the single-loop DC-GPSR algorithm using the BB step size and propose an extension algorithm for the single-loop DC-GPSR.

3.3.1 A Basic Single-Loop DC-GPSR Algorithm

We first rewrite the LS term of the objective in (3.6) in an equivalent form as

$$\frac{1}{2}\|\mathbf{y}-\mathbf{\Phi x}\|_2^2 = \frac{l}{2}\|\mathbf{x}\|_2^2 - \left(\frac{l}{2}\|\mathbf{x}\|_2^2 - \frac{1}{2}\|\mathbf{y}-\mathbf{\Phi x}\|_2^2\right) \tag{3.20}$$

where $l \geq 0$ is a Lipschitz constant of the least square objective. By substituting (3.20) into (3.6), we write the unconstrained sparse reconstruction problem as

$$\min_{\mathbf{x}} \quad \frac{l}{2}\|\mathbf{x}\|_2^2 - \left(\frac{l}{2}\|\mathbf{x}\|_2^2 - \frac{1}{2}\|\mathbf{y}-\mathbf{\Phi x}\|_2^2\right) + \rho(\|\mathbf{x}\|_1 - \|\mathbf{x}\|_{K,1}) := F(\mathbf{x}). \tag{3.21}$$

Then, we perform the following DC decomposition on the objective $F(\mathbf{x})$ defined in (3.21), and we have

$$\min_{\mathbf{x}} \underbrace{\frac{l}{2}\|\mathbf{x}\|_2^2 + \rho\|\mathbf{x}\|_1}_{f(\mathbf{x})} - \underbrace{\left(\frac{l}{2}\|\mathbf{x}\|_2^2 - \frac{1}{2}\|\mathbf{y}-\mathbf{\Phi x}\|_2^2 + \rho\|\mathbf{x}\|_{K,1}\right)}_{g(\mathbf{x})} \tag{3.22}$$

where the functions $f(\mathbf{x})$ and $g(\mathbf{x})$ are convex. The convexity of $g(\mathbf{x})$ can be ensured by confirming the convexity of $\frac{l}{2}\|\mathbf{x}\|_2^2 - \frac{1}{2}\|\mathbf{y}-\mathbf{\Phi x}\|_2^2$, which is given in Theorem 2.

Theorem 3.2. *The least squares objective $\frac{1}{2}\|\mathbf{y}-\mathbf{\Phi x}\|_2^2$ is smooth and its gradient function is Lipschitz continuous with the Lipschitz constant $l = \lambda_{max}(\mathbf{\Phi}^T\mathbf{\Phi})$, where $\lambda_{max}(\cdot)$ denotes the maximum eigenvalue of a matrix.*

Thus, the function $h(\mathbf{x}) = \frac{l}{2}\|\mathbf{x}\|_2^2 - \frac{1}{2}\|\mathbf{y} - \mathbf{\Phi}\mathbf{x}\|_2^2$ *for* $l = \lambda_{max}(\mathbf{\Phi}^T\mathbf{\Phi})$ *is convex.*

Proof: See Appendix B.

Writing the ℓ_2-square terms in (3.22) by standard quadratic forms, we have

$$\begin{aligned}\min_{\mathbf{x}} \quad & \underbrace{\frac{l}{2}\mathbf{x}^T\mathbf{x} + \rho\|\mathbf{x}\|_1}_{f(\mathbf{x})} \\ & \underbrace{-\left(\frac{l}{2}\mathbf{x}^T\mathbf{x} - \frac{1}{2}\mathbf{x}^T\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{x} + (\mathbf{\Phi}^T\mathbf{y})^T\mathbf{x} + \rho\|\mathbf{x}\|_{K,1}\right)}_{g(\mathbf{x})}.\end{aligned} \tag{3.23}$$

We split the positive and negative parts of $\mathbf{x}$ by letting $\mathbf{u} = (\mathbf{x})_+, \mathbf{v} = (-\mathbf{x})_+$. By denoting $\mathbf{z} = [\mathbf{u}^T, \mathbf{v}^T]^T$, we express (3.23) as

$$\begin{aligned}\min_{\mathbf{z}} \quad & \underbrace{\frac{l}{2}\mathbf{z}^T\mathbf{z} + \rho\mathbf{1}^T\mathbf{z}}_{f(\mathbf{z})} - \underbrace{\left(\frac{l}{2}\mathbf{z}^T\mathbf{z} - \frac{1}{2}\mathbf{z}^T\mathbf{B}\mathbf{z} + \mathbf{q}^T\mathbf{z} + \rho\mathbf{1}_K^T\mathbf{z}\right)}_{g(\mathbf{z})} \\ \text{s.t.} \quad & \mathbf{z} \succeq \mathbf{0}\end{aligned} \tag{3.24}$$

where

$$\mathbf{z} = \begin{bmatrix}\mathbf{u}\\ \mathbf{v}\end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix}\mathbf{\Phi}^T\mathbf{\Phi} & -\mathbf{\Phi}^T\mathbf{\Phi}\\ -\mathbf{\Phi}^T\mathbf{\Phi} & \mathbf{\Phi}^T\mathbf{\Phi}\end{bmatrix}, \quad \mathbf{q} = \begin{bmatrix}\mathbf{\Phi}^T\mathbf{y}\\ -\mathbf{\Phi}^T\mathbf{y}\end{bmatrix} \tag{3.25}$$

and $\mathbf{1}_K$ is an indicator vector having either one-valued or zero-valued elements, and the one-valued elements of $\mathbf{1}_K$ indicate the K largest elements of $\mathbf{z}$.

Following the DC algorithm framework to solve the problem (3.24), we perform the following two steps repeatedly until convergence:

$$\begin{aligned}&\text{(a)} \quad \partial g(\mathbf{z}^{t-1}) = l\mathbf{z}^{t-1} - \mathbf{B}\mathbf{z}^{t-1} + \mathbf{q} + \rho\mathbf{1}_K^{t-1}\\ &\text{(b)} \quad \mathbf{z}^t = \underset{\mathbf{z}\geq\mathbf{0}}{\operatorname{argmin}}\{\frac{l}{2}\mathbf{z}^T\mathbf{z} + \rho\mathbf{1}^T\mathbf{z} - \mathbf{z}^T\partial g(\mathbf{z}^{t-1})\}\end{aligned} \tag{3.26}$$

where the superscripts $t-1$ and t, respectively, indicate the $(t-1)$th and the tth update. The subproblem (b) of (3.26) has a closed-form optimal solution. To derive it, we rewrite the subproblem (b) of (3.26) as

$$\begin{aligned} \min_{\mathbf{z}} \quad & \frac{l}{2}\mathbf{z}^T\mathbf{z} + \rho\mathbf{z}^T\mathbf{1} - \mathbf{z}^T\partial g(\mathbf{z}^{t-1}) \\ \text{s.t.} \quad & \mathbf{z} \succeq \mathbf{0} \end{aligned} \tag{3.27}$$

where $\partial g(\mathbf{z}^{t-1}) = l\mathbf{z}^{t-1} - \mathbf{B}\mathbf{z}^{t-1} + \mathbf{q} + \rho\mathbf{1}_K^{t-1}$. Next, the problem (5.10) can be expressed as

$$\begin{aligned} \min_{\mathbf{z}} \quad & \frac{l}{2}\|\mathbf{z} - \frac{1}{l}\left(\partial g(\mathbf{z}^{t-1}) - \rho\mathbf{1}\right)\|_2^2 \\ \text{s.t.} \quad & \mathbf{z} \succeq \mathbf{0}. \end{aligned} \tag{3.28}$$

The objective of problem (3.28) is to minimize the $\|\mathbf{z} - \frac{1}{l}\left(\partial g(\mathbf{z}^{t-1}) - \rho\mathbf{1}\right)\|^2$ over $\mathbf{z} \succeq \mathbf{0}$ with respect to variable $\mathbf{z}$. The solution is simply the Euclidean projection of $\frac{1}{l}\left(\partial g(\mathbf{z}^{t-1}) - \rho\mathbf{1}\right)$ onto the nonnegative orthant. Thus, the gradient projection descent update has a closed-form optimal solution

$$\begin{aligned} \mathbf{z}^* &= \mathit{Proj}\left(\frac{1}{l}\left(\partial g(\mathbf{z}^{t-1}) - \rho\mathbf{1}\right)\right) \\ &= \left(\frac{1}{l}\left(\partial g(\mathbf{z}^{t-1}) - \rho\mathbf{1}\right)\right)_+ \end{aligned} \tag{3.29}$$

where the $\mathit{Proj}(\cdot)$ represents the Euclidean projection onto the feasible set $\mathbf{z} \succeq \mathbf{0}$, which is simply the positive-taking operation $(\cdot)_+$. Applying the closed-form solution (3.29) to the subproblem (b) of (3.26), we simplify the DC programming procedure in (3.26) to perform the following two single-step computations repeatedly until convergence:

$$\begin{aligned} &\text{(a)} \quad \partial g(\mathbf{z}^{t-1}) = l\mathbf{z}^{t-1} - \mathbf{B}\mathbf{z}^{t-1} + \mathbf{q} + \rho\mathbf{1}_K^{t-1} \\ &\text{(b)} \quad \mathbf{z}^t = \left(\frac{1}{l}(\partial g(\mathbf{z}^{t-1}) - \rho\mathbf{1})\right)_+. \end{aligned} \tag{3.30}$$

We summarize the updates (3.30) as a single-loop DC gradient projection algorithm in Algorithm 2. By avoiding the inner-loop iterations, the com-

putational complexity of Algorithm 2 for each iteration is largely reduced compared with Algorithm 1. However, we comment that if the parameter l is large, Algorithm 2 can be slow to converge. Therefore, we will further propose an accelerated algorithm in the following section.

Algorithm 2: Single-loop DC-GPSR-Basic (SlDC-GPSR-Basic)

Input: measurements $\mathbf{y}$, measurement matrix $\mathbf{\Phi}$ and a small number ϵ

Output: reconstruction $\hat{\mathbf{x}}$

Initialization: $\mathbf{u}^0$, $\mathbf{v}^0$, $\mathbf{z}^0 \leftarrow [(\mathbf{u}^0)^T, (\mathbf{v}^0)^T]^T$

1: **for** $t = 1, 2, \ldots$ **do**
2: Compute the gradient $\partial g(\mathbf{z}^{t-1}) = l\mathbf{z}^{t-1} - \mathbf{B}\mathbf{z}^{t-1} + \mathbf{q} + \rho \mathbf{1}_K^{t-1}$ as (a) of (3.30).
3: Perform the optimal projection operation $\mathbf{z}^t = \left(\frac{1}{l}(\partial g(\mathbf{z}^{t-1}) - \rho \mathbf{1})\right)_+$ as (b) of (3.30).
4: Check terminate condition $\|\mathbf{z}^t - \mathbf{z}^{t-1}\|_2 \leq \epsilon$, return to Step 1 if not satisfied; otherwise, terminate with $\mathbf{z}^t = [(\mathbf{u}^t)^T, (\mathbf{v}^t)^T]^T$, and return the reconstruction $\hat{\mathbf{x}} = \mathbf{u}^t - \mathbf{v}^t$.
5: **end for**

3.3.2 An Extension Algorithm for Single-loop DC-GPSR Using Monotonic BB Step Size

A crucial observation on SlDC-GPSR-Basic (Algorithm 2) is that we can interpret it as a simple gradient projection descent method to solve a nonconvex optimization problem with a guarantee of global convergence. This observation can be obtained by substituting $\partial g(\mathbf{z}^{t-1})$ in (a) into (b) of (3.30) such that Step 2 and Step 3 in Algorithm 2 can be combined to be the tth update of $\mathbf{z}^t$

$$\begin{aligned} \mathbf{z}^t &= Proj\left(\mathbf{z}^{t-1} - \frac{1}{l}\left(\mathbf{B}\mathbf{z}^{t-1} - \mathbf{q} - \rho \mathbf{1}_K^{t-1} + \rho \mathbf{1}\right)\right) \\ &= \left(\mathbf{z}^{t-1} - \frac{1}{l}\left(\mathbf{B}\mathbf{z}^{t-1} - \mathbf{q} - \rho \mathbf{1}_K^{t-1} + \rho \mathbf{1}\right)\right)_+ . \end{aligned} \tag{3.31}$$

Thus, we can clearly see that the SlDC-GPSR-Basic is just a gradient projection descent method to solve the following nonconvex problem with the required step size $1/l$ for $l = \lambda_{\max}(\mathbf{\Phi}^T\mathbf{\Phi})$

$$\begin{aligned} \min_{\mathbf{z}} \quad & \frac{1}{2}\mathbf{z}^T\mathbf{B}\mathbf{z} - \mathbf{q}^T\mathbf{z} - \rho\mathbf{1}_K^T\mathbf{z} + \rho\mathbf{1}^T\mathbf{z} := F(\mathbf{z}) \\ \text{s.t.} \quad & \mathbf{z} \succeq \mathbf{0} \end{aligned} \tag{3.32}$$

where $\mathbf{z}$, $\mathbf{B}$ and $\mathbf{q}$ have been defined in (3.24). The problem (3.32) is an equivalent form of the sparse reconstruction problem (3.6), but it uses a nonnegative double-sized variable $\mathbf{z} = [(\mathbf{x})_+^T, (-\mathbf{x})_+^T]^T$ to express the sparse vector $\mathbf{x}$. The gradient projection descent is a mature method for convex optimizations. However, its convergence is unguaranteed for solving a nonconvex problem. Thus, in general, the gradient projection descent method cannot be directly applied to a nonconvex problem. Interestingly, as shown, the proposed SlDC-GPSR-Basic algorithm can be viewed as the gradient projection descent method applied to nonconvex optimizations. Since this algorithm is derived from DC programming, it carries the same global convergence property as the other DC programming algorithms [86].

Although the fixed step size $1/l$ ensures a global convergence for the SlDC-GPSR-Basic algorithm, this step size can be too small in practice provided that l is the Lipschitz constant of least squares objective. Thus, we design a variable step size to accelerate the single-loop DC-GPSR algorithm. We denote α^t by a variable step size for the $(t+1)$th update, and extend the update (3.31) to be a gradient projection descent update for the problem (3.32) with step size α^t, i.e.,

$$\begin{aligned} \tilde{\mathbf{z}}^{t+1} &= Proj\left(\mathbf{z}^t - \alpha^t\left(\nabla F(\mathbf{z}^t)\right)\right) \\ &= \left(\mathbf{z}^t - \alpha^t\left(\mathbf{B}\mathbf{z}^t - \mathbf{q} - \rho\mathbf{1}_K^t + \rho\mathbf{1}\right)\right)_+ \end{aligned} \tag{3.33}$$

where α^t can be explicitly calculated by the BB step size method by

$$\alpha^t = \frac{\|\mathbf{z}^t - \mathbf{z}^{t-1}\|^2}{(\mathbf{z}^t - \mathbf{z}^{t-1})^T\left(F(\mathbf{z}^t) - F(\mathbf{z}^{t-1})\right)}. \tag{3.34}$$

To prevent the step size being overly large, we employ a scaler $\beta^{t+1} \in (0, 1]$ to limit the update so that the objective will descend monotonically

$$\mathbf{z}^{t+1} = \mathbf{z}^t + \beta^t(\tilde{\mathbf{z}}^{t+1} - \mathbf{z}^t). \tag{3.35}$$

We can calculate β^t to minimize the objective $F(\mathbf{z}^{t+1})$ by

$$\beta^t = \frac{(\boldsymbol{\delta}^t)^T \nabla F(\mathbf{z}^t)}{(\boldsymbol{\delta}^t)^T \mathbf{B} \boldsymbol{\delta}^t} \tag{3.36}$$

where $\boldsymbol{\delta}^t = \tilde{\mathbf{z}}^{t+1} - \mathbf{z}^t$. We summarize the updates (3.33) and (3.35) in Algorithm 3 and name it SlDC-GPSR-BB, which is an extension of the single-loop DC-GPSR algorithm since it extends Algorithm 2 by introducing the BB step size.

Algorithm 3: Single-loop DC-GPSR with monotonic BB step size (SlDC-GPSR-BB)

Input: measurements $\mathbf{y}$, measurement matrix $\boldsymbol{\Phi}$ and a small number ϵ

Output: reconstruction $\hat{\mathbf{x}}$

Initialization: $\mathbf{u}^0$, $\mathbf{v}^0$, $\mathbf{z}^0 \leftarrow [(\mathbf{u}^0)^T, (\mathbf{v}^0)^T]^T, \alpha^0$

1: **for** $t = 0, 1, 2, \ldots$ **do**
2: Compute update $\tilde{\mathbf{z}}^{t+1}$ using (3.33).
3: Compute step size β^t using (3.36).
4: Compute update $\mathbf{z}^{t+1}$ using (3.35).
5: Compute step size α^{t+1} using (3.34).
6: Check terminate condition $\|\mathbf{z}^{t+1} - \mathbf{z}^t\|_2 \leq \epsilon$, return to Step 1 if not satisfied; otherwise, terminate with $\mathbf{z}^{t+1} = [(\mathbf{u}^{t+1})^T, (\mathbf{v}^{t+1})^T]^T$ and return the reconstruction $\hat{\mathbf{x}} = \mathbf{u}^{t+1} - \mathbf{v}^{t+1}$.
7: **end for**

3.4 Complexity Analysis

Table 3.1 shows the computational cost of the main computing operations and compares the computational complexities of the proposed algo-

Table 3.1: Computational Complexities for Operations and Algorithms

Operation	$\partial\|\mathbf{z}\|_{K,1}$		$\mathbf{Bz}$
Time complexity	$O(2N\log(2N))$ [11]		$O(MN)$
Algorithm	DlDC-GPSR	SlDC-GPSR-Basic	SlDC-GPSR-BB
Time complexity	$I \times O(MN)$ [12]	$O(MN)$	$O(MN)$

rithms, where N is the dimension of the sparse vector $\mathbf{x}$, and $2N$ is the length of vector $\mathbf{z}$, M is the dimension of the measurements $\mathbf{y}$. Common operations for all three algorithms mainly contain matrix-vector multiplications, vector inner products, vector sums, scaler-vector multiplications and the subgradient of the top-$(K,1)$ norm for a nonnegative vector. Among these operations, the computationally intensive terms have the subgradient computation $\partial\|\mathbf{z}\|_{K,1}$ and the matrix-vector product $\mathbf{Bz}$ whose computational costs are shown in Table 3.1. To analyze the overall time complexities of the proposed algorithms, we consider both the computational costs at each iteration and the convergence rates. The computational costs at each iteration [10] for these algorithms are shown in Table 3.1, while the convergence rates will be illustrated in Section VII.B. By taking into account of the convergence rates, we can conclude that, among the three proposed algorithms, SIDC-GPSR-BB has the lowest time complexity.

3.5 Numerical Results

3.5.1 Experiment Setup and Performance Metrics

We consider a downlink massive MIMO system having a half-wavelength spaced ULA at the BS. The number of BS antennas is set to be $N_t = 256$.

[10] To compare fairly, one iteration is considered from Step 2 to Step 5 for DlDC-GPSR, Step 2 and Step 3 for SlDC-GPSR-Basic and from Step 2 to Step 5 for SlDC-GPSR-BB.

[10] A feasible subgradient computation of $\partial\|\mathbf{z}\|_{K,1}$ requires the indices of the K largest elements to be obtained by sorting the elements, which can be accomplished with the time complexity $O(2N\log(2N))$. This time complexity is only an estimate, as more efficient sorting algorithms can exist.

[12] We assume that an arbitrary outer iteration of DlDC-GPSR has I inner loops. Given a termination condition, the number of inner-loop iterations I can vary for each outer-loop iteration of DlDC-GPSR.

The number of UE antennas is set to be $N_r = 1$ without loss of generality[13]. We consider the narrowband block-fading channels, and set a channel coherence block as $N_c = 600$ symbols. We randomly generate $1,000$ channel vectors according to the Saleh-Valenzuela channel model described in (2.10), where the number of paths is set as $N_p = 3$. For each path, the complex channel gain α_l follows a complex Gaussian distribution; the AoA and AoD, i.e., $\theta_{r,l}$ and $\theta_{t,l}$, are uniformly distributed over $[-\pi/2, \pi/2]$. We transform the generated channel vectors into the angular domain using a DFT. To consider power leakage, we set the number of nonzero channel coefficients as $N_s = 16$ by neglecting the small-value beamspace channel coefficients [14]. Various random matrices can be adopted as the measurement matrix. After evaluating the proposed reconstruction algorithms that adopt various random matrices (such as Gaussian, Bernoulli, and partial Fourier matrices), we conclude that the reconstruction performance of these random matrices is similar. Unless stated otherwise, we adopt the random Gaussian matrix as the measurement matrix $\dot{\mathbf{\Phi}}_G \in \mathbb{R}^{M \times N_t}$, and the pilot matrix can be obtained by $\mathbf{P} = \mathbf{U}_t \dot{\mathbf{\Phi}}_G^T \in \mathbb{C}^{N_t \times M}$. All of the simulations were implemented on a desktop computer equipped with a 3.2 GHz Intel Core i7-8700 CPU with 8GB of physical memory.

We adopt the normalized mean squared error (NMSE) to evaluate the channel reconstruction accuracy. The NMSE is defined as $\frac{1}{n}\sum_{i=1}^{n}\frac{\|\mathbf{H}_i - \hat{\mathbf{H}}_i\|_F^2}{\|\mathbf{H}_i\|_F^2}$, where n is the number of channel samples. We use achievable spectral efficiency to evaluate the influences of pilot overhead and SNR penalty caused by imperfect channel estimation on data communication. The achievable spectral efficiency R/B is defined as $R/B = (1-\alpha)C(\mathrm{SNR}_{\mathrm{eff}})$ [88, eq. (5.195)], where R is the achievable rate, B is the bandwidth, α is the ratio of pilot training in a channel coherence block duration, $C(\mathrm{SNR}_{\mathrm{eff}})$ is the

[13] According to the proposed column-wise broadcasting method to vectorize the beamspace channel matrix $\mathbf{H} \in \mathbb{C}^{N_t \times N_r}$, to reconstruct $\mathbf{H}$ we simply reconstruct its N_r columns independently.

[14] Although in practice the number of nonzero channel coefficients can be unknown and uncertain, in the simulation we fix the number of nonzero channel coefficients for fair performance comparisons. We set $N_s = 16$ because our observations revealed that the 16 largest-magnitude channel coefficients can cover most of the energy of a channel vector, i.e., $\|\mathbf{h}_i\|_2^2$ for $1 \le i \le N_r$.

channel capacity at SNR_{eff}, and SNR_{eff} refers to the efficient SNR by considering imperfect channel estimation [88]. Note that α can be calculated as M/N_c, where N_c is the length of channel coherence block. According to the linear equation $\tilde{\mathbf{R}} = \dot{\mathbf{\Phi}}\mathbf{H} + \tilde{\mathbf{W}}$ in (2.28), we define the system SNR as $\text{SNR} = \frac{\|\dot{\mathbf{\Phi}}\mathbf{H}\|_F^2}{\tilde{\mathbf{W}}_F^2}$, where $\dot{\mathbf{\Phi}}$ is the measurement matrix, $\mathbf{H}$ is the beamspace channel, and $\tilde{\mathbf{W}}$ is the noise.

3.5.2 Illustrations of Beamspace Channel Reconstructions and Algorithm Convergence Property

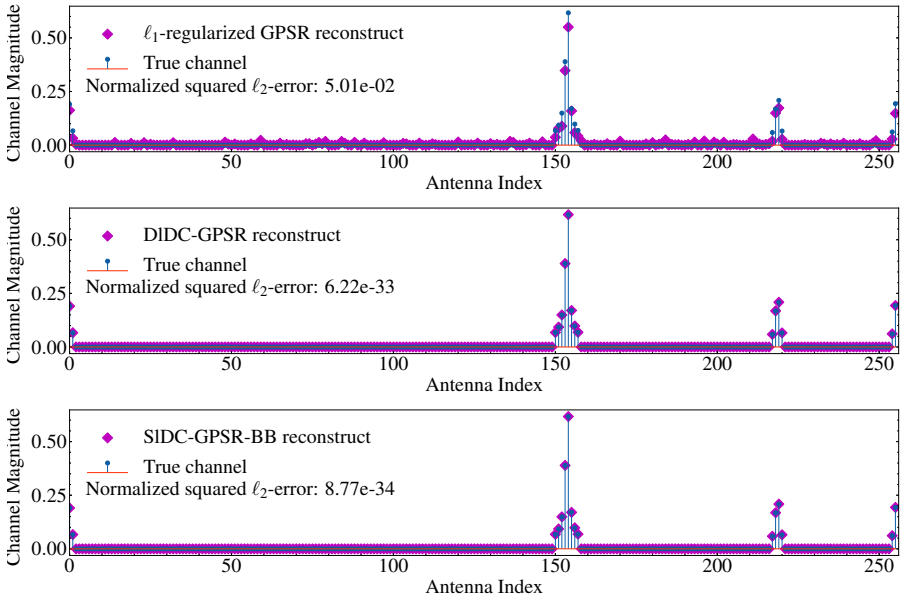


Figure 3.1: Channel magnitudes of a true channel sample $\mathbf{x}_{\text{opt}}$ and the reconstruction $\hat{\mathbf{x}}$ by the conventional ℓ_1-regularized GPSR algorithm, the proposed algorithms DlDC-GPSR and SlDC-GPSR-BB. The normalized squared ℓ_2-error is defined as $\|\mathbf{x}_{\text{opt}} - \hat{\mathbf{x}}\|_2^2 / \|\mathbf{x}_{\text{opt}}\|_2^2$

In this subsection, we evaluate the reconstruction accuracy and convergence rate of DlDC-GPSR (Algorithm 1) and SlDC-GPSR-BB (Algorithm 3) [15] by reconstructing an arbitrary channel vector sample $\mathbf{x}$ and compare

[15] As we have shown, the SlDC-GPSR-Basic (Algorithm 2) has an interesting theoretical

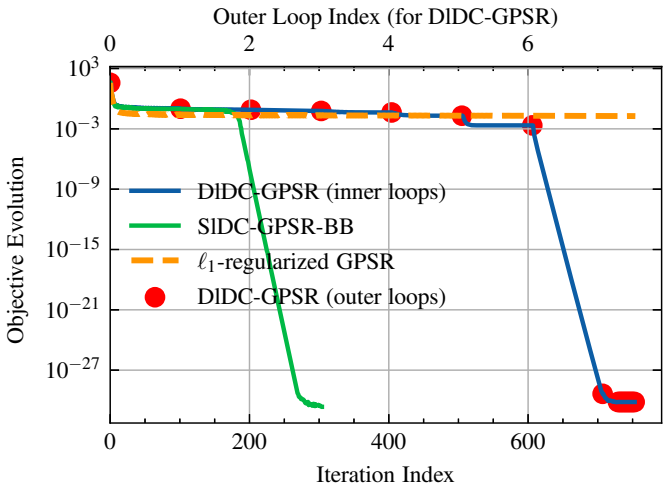


Figure 3.2: Objective values versus iterations for reconstructing a sample of beamspace channel vector. The evaluating objectives are $\frac{1}{2}\|\mathbf{y}-\mathbf{\Phi x}\|_2^2+\rho(\|\mathbf{x}\|_1-\|\mathbf{x}\|_{K,1})$ for DlDC-GPSR and SlDC-GPSR-BB, and is $\frac{1}{2}\|\mathbf{y}-\mathbf{\Phi x}\|_2^2+\rho\|\mathbf{x}\|_1$ for ℓ_1-regularized GPSR

the performance with the conventional ℓ_1-regularized GPSR algorithm [50]. For fair comparisons, we adopt BB step sizes for the inner loops of DlDC-GPSR. The pilot length is set as $M = 128$. Fig. 3.1 shows the magnitudes of the true sparse beamspace channel and the reconstructions. We can observe that the DlDC-GPSR and SlDC-GPSR-BB algorithms can achieve perfect reconstructions with normalized squared ℓ_2-errors 6.22×10^{-33} and 8.77×10^{-34}, whereas the ℓ_1-regularized GPSR reconstruction has noticeable errors with the error 5.01×10^{-2}. By setting the termination condition as $\|\hat{\mathbf{x}}^t - \hat{\mathbf{x}}^{t-1}\|_2 \leq 10^{-30}$, the runtimes are about 0.08, 0.08 and 0.05 seconds for ℓ_1-regularized GPSR, DlDC-GPSR and SlDC-GPSR-BB, respectively. To show their convergence properties within runtimes, we plot objective values versus iterations in Fig. 3.2. It should be noted that the objective of the proposed DlDC-GPSR and SlDC-GPSR-BB algorithms is $\frac{1}{2}\|\mathbf{y}-\mathbf{\Phi x}\|_2^2+\rho(\|\mathbf{x}\|_1-\|\mathbf{x}\|_{K,1})$, whereas the objective of conventional ℓ_1-

interpretation since it can be treated as a pure gradient projection algorithm, but the required step size $1/l$ is often too small to converge within a reasonable period. Therefore, we do not consider Algorithm 2 in simulations.

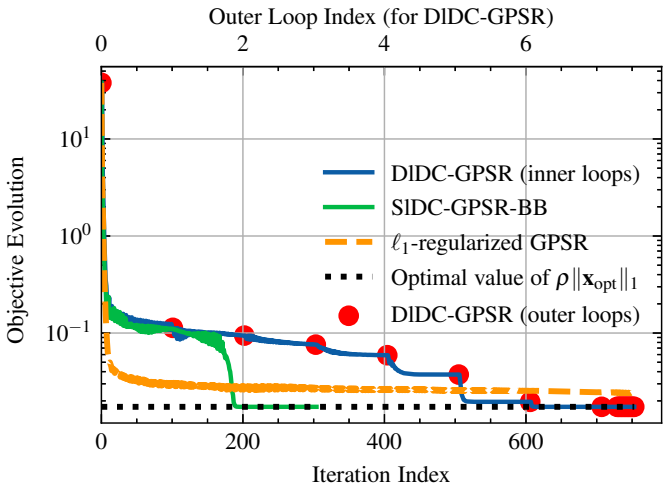


Figure 3.3: Objective values ($\frac{1}{2}\|\mathbf{y}-\mathbf{\Phi x}\|_2^2+\rho\|\mathbf{x}\|_1$) versus iterations for the reconstruction of a sample of beamspace channel vector

regularized GPSR is $\frac{1}{2}\|\mathbf{y}-\mathbf{\Phi x}\|_2^2+\rho\|\mathbf{x}\|_1$. For DlDC-GPSR algorithm, the red dots indicate objective values for outer iterations, and the blue solid line shows the objective values along all inner iterations. We can observe that the proposed DlDC-GPSR algorithm decreases its objective significantly after the sixth outer-step and approaches the optimal value at the eighth step. The SlDC-GPSR-BB algorithm has the fastest convergence and can achieve almost the same objective value as the DlDC-GPSR algorithm. On the contrary, the objective of the ℓ_1-regularized GPSR algorithm converges to a relatively large value. Fig. 3.3 shows the evolution of ℓ_1-norm penalized least-square objective values, i.e., $\|\mathbf{y}-\mathbf{\Phi x}\|_2^2+\rho\|\mathbf{x}\|_1$. In Fig. 3.3, we also plot the optimal values of the penalty term $\rho\|\mathbf{x}_{\text{opt}}\|_1$, where $\mathbf{x}_{\text{opt}}$ represents the true sample of a beamspace channel vector. We can observe that the DlDC-GPSR and SlDC-GPSR-BB algorithms can arrive and stay at the objective value $\rho\|\mathbf{x}_{\text{opt}}\|_1$, which is the optimal value that the objective $\|\mathbf{y}-\mathbf{\Phi x}\|_2^2+\rho\|\mathbf{x}\|_1$ can reach when $\|\mathbf{y}-\mathbf{\Phi}\hat{\mathbf{x}}\|_2^2=0$ and $\rho\|\hat{\mathbf{x}}\|_1=\rho\|\mathbf{x}_{\text{opt}}\|_1$. However, the ℓ_1-regularized GPSR algorithm cannot achieve this optimal value and has a noticeable gap from the optimal value. It is meaningful to observe this minimum objective gap between our proposed algorithms and

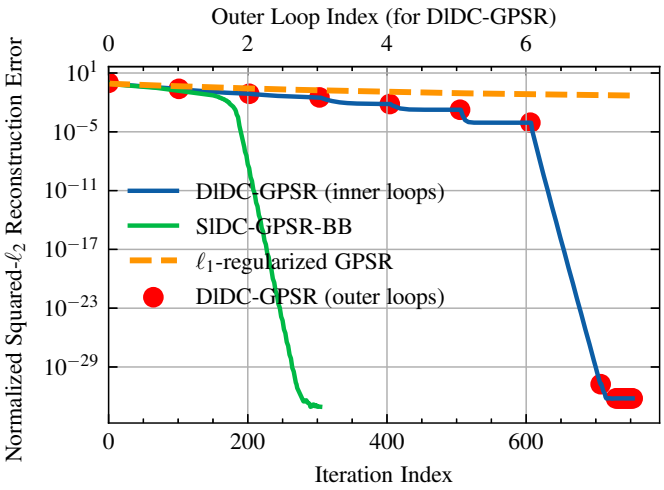


Figure 3.4: Reconstruction error ($\|\hat{\mathbf{x}} - \mathbf{x}_{\text{opt}}\|_2^2 / \|\mathbf{x}_{\text{opt}}\|_2^2$) versus iterations for the reconstruction of a sample of beamspace channel vector

the conventional ℓ_1-regularized GPSR algorithm, because this gap can provide important insight into the approximation error introduced by relaxing the ℓ_0-norm as ℓ_1-norm. Fig. 3.4 shows the normalized squared-ℓ_2 errors of reconstruction versus the number of iterations. We observe that both the DlDC-GPSR and SlDC-GPSR-BB algorithms can achieve accurate reconstruction having errors on the order of 10^{-33} and 10^{-34}, which is far more accurate than the reconstruction by the conventional ℓ_1-regularized GPSR algorithm having an error on the order of 10^{-2}.

3.5.3 Performance Comparisons With Other Algorithms

We compare the reconstruction performance of the proposed algorithms with several existing sparse reconstruction algorithms including the ℓ_1-regularized GPSR [50], ISTA [48], and OMP [45], as well as the existing state-the-art channel estimation schemes including the SD [16] [89] and the SCAPMI algorithms [90].

[16]The SD algorithm is referred to as the support detection-based channel estimation algorithm [89].

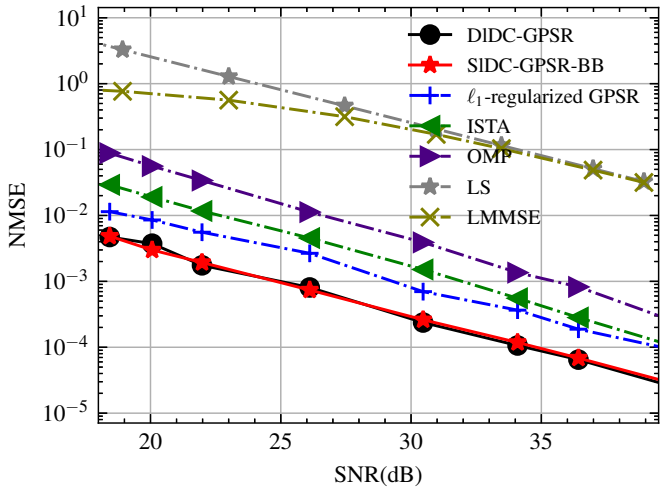


Figure 3.5: Channel reconstruction NMSE versus SNR (dB) with training pilot length $M = 128$

In Fig. 3.5, we compare the channel reconstruction NMSE with conventional channel estimation methods including the LS and LMMSE, and popular sparse reconstruction algorithms including the OMP, ISTA and ℓ_1-regularized GPSR. For conventional LS and LMMSE channel estimation methods, we use the optimal pilot matrix that contains orthonormal training pilot sequences having the length of $M' = 256$. For the sparse reconstruction algorithms including DlDC-GPSR, SlDC-GPSR-BB, OMP, ISTA and ℓ_1-regularized GPSR, we set the training pilot length as $M = 128$. Fig. 3.5 shows that the proposed DlDC-GPSR and SlDC-GPSR-BB algorithms achieve approximately the same accuracy. Even after consuming more training pilots, both LS and LMMSE channel estimation methods have higher channel reconstruction errors than the sparse reconstruction methods. Also, the proposed algorithms DlDC-GPSR and SlDC-GPSR-BB have higher accuracy than the other sparse reconstruction algorithms. The average runtimes required for reconstructing a channel sample are summarized in Table 3.2 for different sparse reconstruction algorithms.

For sparse channel reconstructions, increasing the pilot length can improve reconstruction accuracy. However, longer pilot length will also increase

Table 3.2: Average Runtimes Per Channel Sample Reconstruction

Algorithm	SIDC-GPSR-BB	DIDC-GPSR	ℓ_1-regularized GPSR	ISTA	OMP
SNR = 40 dB	0.03s	0.67s	0.12s	0.61s	0.16s
SNR = 30 dB	0.07s	1.61s	0.19s	0.86s	0.21s
SNR = 18 dB	0.16s	2.39s	0.26s	1.89s	0.31s

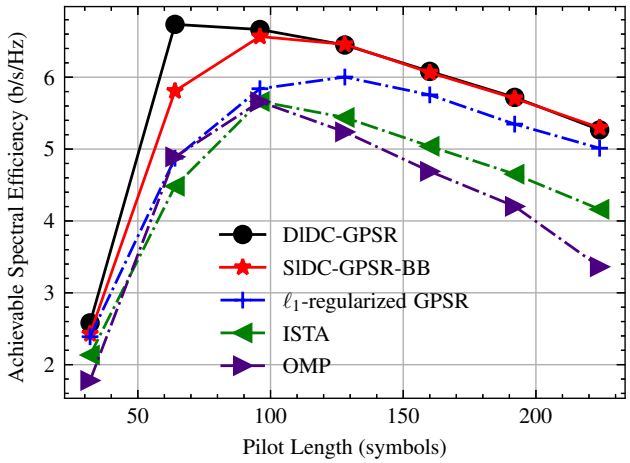


Figure 3.6: Achievable spectral efficiency R/B versus training pilot length at SNR = 25 dB

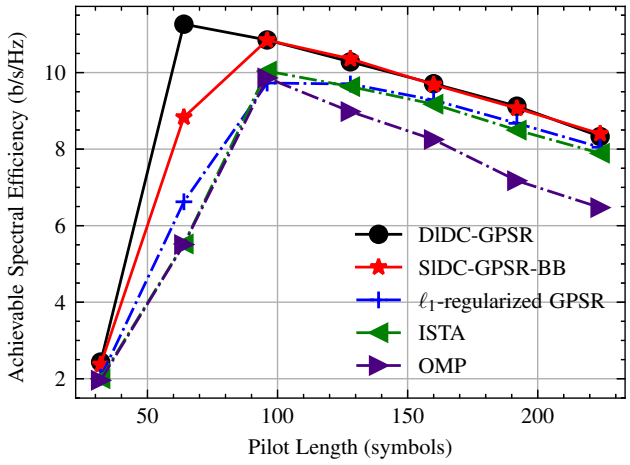


Figure 3.7: Achievable spectral efficiency R/B versus training pilot length at SNR = 40 dB

the training overhead and subsequently degrade the spectral efficiency. In other words, there exists a tradeoff between the training pilot overhead and spectral efficiency. To illustrate this tradeoff for different sparse reconstruction algorithms, we plot the achievable spectral efficiency versus training pilot length M for SNR = 25 dB and SNR = 40 dB in Fig. 3.6 and Fig. 3.7, where the length of a channel coherence block is set as $N_c = 600$ symbols, and the training pilot length is set as $M \in \{32, 64, ..., 256\}$. We can observe that the achievable spectral efficiencies increase when the training pilot length M increases, and then decrease after reaching the peak values. When M has a small value, the channel estimation error can be decreased by increasing the number of training symbols, such that the efficient SNR, i.e., SNR_{eff}, increases and dominates the achievable spectral efficiency. However, when M is sufficiently large, a further increase of M will give a diminishing benefit while the data communication ratio $(1 - \alpha)$ decreases linearly. Fig. 3.6 and Fig. 3.7 show that the proposed DlDC-GPSR and SlDC-GPSR-BB algorithms can attain higher achievable spectral efficiencies than the ℓ_1-regularized GPSR, ISTA and OMP sparse reconstruction algorithms.

We now compare the performance of the proposed algorithms with the state-of-the-art beamspace channel estimation methods including the SD and SCAMPI algorithms. For fair comparisons, we selected a 16×16 lens antenna array size for SCAMPI, and chose $N_t = 256$ for the ULA antenna number for both the SD and the proposed algorithms. The training pilot length is set as $M = 96$. To allow the proposed algorithms to be contrasted fairly with the SD and SCAMPI algorithms, we constructed the same type of measurement matrix used in [89] and [90]. The measurement matrix is a Rademacher matrix, which contains entries drawn from $\{+1, -1\}$ with equal probabilities. Fig. 3.8 shows the results of channel reconstruction NMSE versus SNR. At high SNR, the SD and SCAMPI algorithms appear to reach an error floor of approximately 10^{-2}, while the proposed SlDC-GPSR-BB and DlDC-GPSR algorithms achieve lower reconstruction errors. Fig. 3.9 compares the achievable spectral efficiencies by different channel estimation schemes. The proposed SlDC-GPSR-BB and DlDC-GPSR algorithms

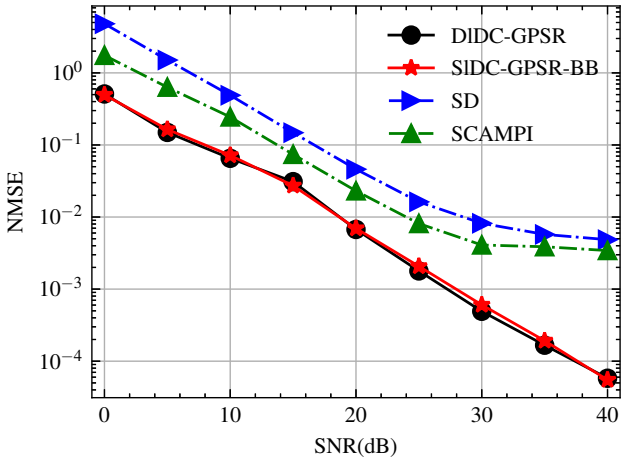


Figure 3.8: NMSE comparison for the training pilot length $M = 96$. The SCAMPI algorithm under uniform distribution is configured to have a 16×16 lens antenna array. The SD algorithm and the proposed SlDC-GPSR-BB and DlDC-GPSR algorithms are configured to have a ULA with 256 antennas.

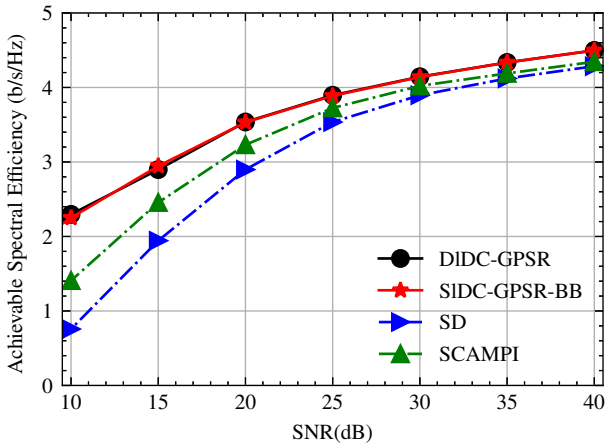


Figure 3.9: Comparison of the achievable spectral efficiency R/B for the training pilot length $M = 96$. The SCAMPI algorithm under uniform distribution is configured to have a 16×16 lens antenna array. The SD algorithm and the proposed SlDC-GPSR-BB and DlDC-GPSR algorithms are configured to have a ULA with 256 antennas.

achieve higher spectral efficiency than the SD and SCAMPI algorithms.

3.6 Summary

We proposed three DC programming gradient projection sparse reconstruction algorithms for massive MIMO beamspace channel estimation, and they are DlDC-GPSR, SlDC-GPSR-Basic and SlDC-GPSR-BB. We designed these algorithms by solving an LS problem having a nonconvex regularizer. The regularizer is simply the difference between an ℓ_1-norm and a top-$(K, 1)$ norm, which was introduced to remove the penalties on the K largest-magnitude elements of the reconstructing vector. We employed DC programming and a gradient projection method to solve the nonconvex sparse reconstruction, and derived different double-loop and single-loop algorithms by different DC decompositions. The double-loop DlDC-GPSR algorithm is simple, accurate, and robust. The SlDC-GPSR-Basic algorithm has simple single-loop updates, and it shares the same theoretical convergence property as the other DC programming methods. We observed that the SlDC-GPSR-Basic updates can be perfectly interpreted as simple gradient projection descent updates. Based on this observation, we adopted the BB step-size strategy and proposed an extension algorithm SlDC-GPSR-BB, which has a significantly improved convergence rate and can achieve approximately the same level accuracy as the double-loop algorithm DlDC-GPSR. Several numerical demonstrations were provided to show the proposed algorithms have high accuracy and efficient implementation.

Chapter 4

Improving Sparse Channel Reconstruction via Data-Driven Measurement Matrices

In Chapter 3, we proposed DC-GPSR algorithms that have improved sparse reconstruction performance than existing channel estimation. However, the proposed algorithms use common random measurement matrices, which are widely-used but suboptimal solutions to measurement matrix design. In this chapter, we explore using data-driven measurement matrices to improve further sparse channel estimation.

4.1 Real-valued Measurement Matrix Design

As described in Section 2.2.3, the sparse channel estimation problem can be formulated into an SMV problem in (2.31) as shown

$$\begin{bmatrix} \Re(\mathbf{r}_i) \\ \Im(\mathbf{r}_i) \end{bmatrix} = \begin{bmatrix} \Re(\dot{\mathbf{\Phi}}) & -\Im(\dot{\mathbf{\Phi}}) \\ \Im(\dot{\mathbf{\Phi}}) & \Re(\dot{\mathbf{\Phi}}) \end{bmatrix} \begin{bmatrix} \Re(\mathbf{h}_i) \\ \Im(\mathbf{h}_i) \end{bmatrix} + \begin{bmatrix} \Re(\mathbf{w}_i) \\ \Im(\mathbf{w}_i) \end{bmatrix} \tag{4.1}$$

where i for $1 \leq i \leq N_r$; $\Re(\cdot)$ and $\Im(\cdot)$ denote the real part and imaginary part of a complex matrix or a vector. Now, we adopt the design criterion to force the measurement matrix to take real values $\mathbf{\Phi} \in \mathbb{R}^{M \times N_t}$, i.e., $\Im(\dot{\mathbf{\Phi}}) = \mathbf{0}$,

$\mathbf{\Phi} = \Re(\dot{\mathbf{\Phi}})$. Thus, eq. (4.1) can be expressed as

$$\begin{bmatrix} \Re(\mathbf{r}) \\ \Im(\mathbf{r}) \end{bmatrix} = \begin{bmatrix} \mathbf{\Phi} & \mathbf{0} \\ \mathbf{0} & \mathbf{\Phi} \end{bmatrix} \begin{bmatrix} \Re(\mathbf{h}) \\ \Im(\mathbf{h}) \end{bmatrix} + \begin{bmatrix} \Re(\mathbf{w}) \\ \Im(\mathbf{w}) \end{bmatrix} \tag{4.2}$$

where the subscript i is omitted. Eq. (4.2) can be equivalently expressed by

$$\begin{aligned} \Re(\mathbf{r}) &= \mathbf{\Phi} \cdot \Re(\mathbf{h}) + \Re(\mathbf{w}) \\ \Im(\mathbf{r}) &= \mathbf{\Phi} \cdot \Im(\mathbf{h}) + \Im(\mathbf{w}). \end{aligned} \tag{4.3}$$

When adopting real-valued measurement matrices by forcing the imaginary part of $\mathbf{\Phi}$ to be zeros, the computational complexity reduces to a half of the original complex-valued matrix-vector multiplication. Moreover, the reconstruction performance of the real-valued matrices, such as the Gaussian matrix or Bernoulli matrix, were tested and found to be similar to the reconstruction performance of the complex-valued random Fourier matrix. We conclude that the real-valued random matrix design can reduce the computational complexity without degrading the reconstruction performance.

We consider two application scenarios to apply the real-valued measurement matrix $\mathbf{\Phi}$ to downlink CSI acquisitions in FDD massive MIMO systems. The first application is to design downlink pilots. Based on (2.27), the pilot matrix $\mathbf{P}$ and the measurement matrix $\mathbf{\Phi}$ can be related by $\mathbf{P} = \mathbf{U}_t\mathbf{\Phi}^T$, where the matrix $\mathbf{U}_t$ is a DFT matrix. Thus, designing a complex-valued pilot matrix [17] $\mathbf{P}$ can be equivalently realized by designing a real-valued measurement matrix $\mathbf{\Phi}$. A practical pilot matrix design needs to consider a power constraint, and ideally each column of $\mathbf{P}$ is expected to have unit ℓ_2-norm to satisfy the power constraint on the pilot sequences transmitted by each BS antenna. This power constraint on the pilot matrix can be equivalently accomplished by normalizing the columns of the measurement matrix $\mathbf{\Phi}$. The second application of the measurement matrix $\mathbf{\Phi}$ is CSI feedback reduction. If ideal CSI is already known at the UE via perfect channel estimations, then the UE can linearly compress the beamspace channel $\mathbf{H}$

[17] Note that the pilot matrix $\mathbf{P}$ is complex valued because $\mathbf{U}_t$ is a complex-valued unitary matrix.

by simply multiplexing the measurement matrix $\mathbf{\Phi}$. Instead of transmitting the high-dimensional beamspace channel $\mathbf{H}$, the UE sends the compressed measurements $\mathbf{Y} = \mathbf{\Phi H}$ back to the BS. At the end of the process, the beamspace channels can be reconstructed at the BS. This process of compressed CSI feedback reduces the CSI feedback overhead significantly.

4.2 Learning Measurement Matrices

For the application of either pilot design or CSI feedback compression, it is essential to design a suitable measurement matrix $\mathbf{\Phi}$ to reconstruct the sparse beamspace channels accurately, since the choice of measurement matrix $\mathbf{\Phi}$ highly influences reconstruction performance. Random matrices are widely adopted in existing sparse channel reconstruction schemes, but they are suboptimal. We will provide improved data-driven measurement matrices acquired via training our proposed model-based autoencoders. In particular, we propose a model-based framework of deep unfolding basis pursuit autoencoders (*BP-AE*), which are designed by unfolding the linear compression and the iterations of basis pursuit sparse reconstructions. Following this framework, we propose two basic autoencoder models to acquire data-driven measurement matrices. Moreover, we propose two extension models to acquire measurement matrices having double columns, which further improves sparse reconstruction performance.

4.2.1 Framework of Deep Unfolding *BP-AE*

For a beamspace channel dataset $\{\mathbf{X}\}$ containing n beamspace channel samples, we represent this dataset by

$$\{[\mathbf{H}_1, \mathbf{H}_2, ..., \mathbf{H}_n]\} = \{[\Re(\mathbf{h}_{b,1}), \Im(\mathbf{h}_{b,1}), \Re(\mathbf{h}_{b,2}), \Im(\mathbf{h}_{b,2}), ..., \Re(\mathbf{h}_{b,n}), \Im(\mathbf{h}_{b,n})]\}. \tag{4.4}$$

Without loss of generalization, we refer to the vector $\mathbf{x} \in \mathbb{R}^N$ as an arbitrary column in the dataset $\{\mathbf{X}\}$, and thus the vector $\mathbf{x}$ represents either $\Re(\mathbf{h}_{b,i})$ or $\Im(\mathbf{h}_{b,i})$ for $1 \leq i \leq n$. Then, we express the sparse channel reconstruction

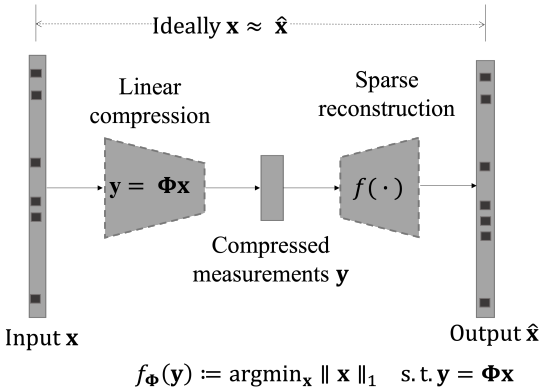


Figure 4.1: Framework of deep unfolding *BP-AE*

problem in a general form by the underdetermined equation

$$\mathbf{y} = \mathbf{\Phi x} + \mathbf{n} \tag{4.5}$$

where $\mathbf{y} \in \mathbb{R}^M$ is the measurement vector, $\mathbf{x} \in \mathbb{R}^N$ is a sparse vector, $\mathbf{n} \in \mathbb{R}^M$ is the noise vector. We will design an autoencoder that can learn the latent representation (code) $\mathbf{y}$ for input data $\mathbf{x}$, and can subsequently output the reconstruction $\hat{\mathbf{x}}$ such that $\hat{\mathbf{x}} \approx \mathbf{x}$. An autoencoder consists of two main parts: an encoder that maps the input into the code, i.e., $\mathbf{y} = g(\mathbf{x})$, and a decoder that maps the code to the reconstruction of the original input, i.e., $\hat{\mathbf{x}} = f(\mathbf{y})$. Here, functions $g(\cdot)$ and $f(\cdot)$ represent the overall nonlinear transformations of the encoder and the decoder respectively. Based on an observation that the compressive sensing and recovery process can be regarded as a realization of a feedforward computation of an autoencoder, we propose an autoencoder framework called deep unfolding *BP-AE*, which mimics the process of linear compression and basis pursuit sparse reconstruction. As shown in Fig. 4.1, the deep unfolding *BP-AE* consists of a linear encoder and a nonlinear decoder, which are represented by

$$g_{\{\mathbf{\Phi}\}}(\mathbf{x}) := \mathbf{\Phi x} \tag{4.6}$$

$$f_{\{\mathbf{\Phi}\}}(\mathbf{y}) := \underset{\mathbf{x}}{\operatorname{argmin}} \|\mathbf{x}\|_1 \quad \text{s.t.} \quad \mathbf{y} = \mathbf{\Phi x} \tag{4.7}$$

where $\|\cdot\|_1$ represents the ℓ_1-norm, which is defined as the sum of absolute values of all elements in a vector. The encoder $g_{\{\mathbf{\Phi}\}}(\mathbf{x})$ in (4.6) performs a linear dimension reduction; the decoder $f_{\{\mathbf{\Phi}\}}(\mathbf{y})$ in (4.7) solves an ℓ_1-minimization sparse reconstruction, which is also known as the basis pursuit in signal processing. We choose to unfold the basis pursuit algorithm because the linear constraint in (4.7) can strictly force the measurement matrix to perform a proper dimensionality-reduction mapping and retain more useful information of the sparse vectors. Another important observation is that we can treat the measurement matrix as the weights used within the deep unfolding *BP-AE*. Therefore, we can parameterize the autoencoder with a trainable measurement matrix $\mathbf{\Phi}$ as the tied weights of both the encoder and the decoder, so that the measurement matrix can be optimized during model training with given dataset by backpropagation algorithms. The loss function of the deep unfolding *BP-AE* is defined as the MSE between input samples and output reconstructed vectors,

$$\mathcal{L}(\mathbf{x}, \hat{\mathbf{x}}) = \frac{1}{n} \sum_{i=1}^{n} \|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2 \tag{4.8}$$

where n is the number of training samples in dataset $\{\mathbf{X}\}$; $\mathbf{x}_i$ represents the ith input sample; $\hat{\mathbf{x}}_i$ represents the ith reconstructed vector. From the viewpoint of feedforward computation, the autoencoder performs a linear dimension reduction jointly with a basis pursuit sparse reconstruction, thus it can be interpreted as a compressive sensing and reconstruction process. From the viewpoint of backward propagation, the autoencoder back propagates and minimizes the reconstruction error in (4.8) by optimizing the measurement matrix, thus it can be interpreted as a stochastic optimizer for the measurement matrix. The optimizer can adopt any standard backpropagation algorithm in deep learning such as the stochastic gradient descent (SGD) algorithm [91]. After training, we can extract the weights $\mathbf{\Phi} \in \mathbb{R}^{M \times N}$ from the trained autoencoder and save it as the acquired data-driven measurement matrix.

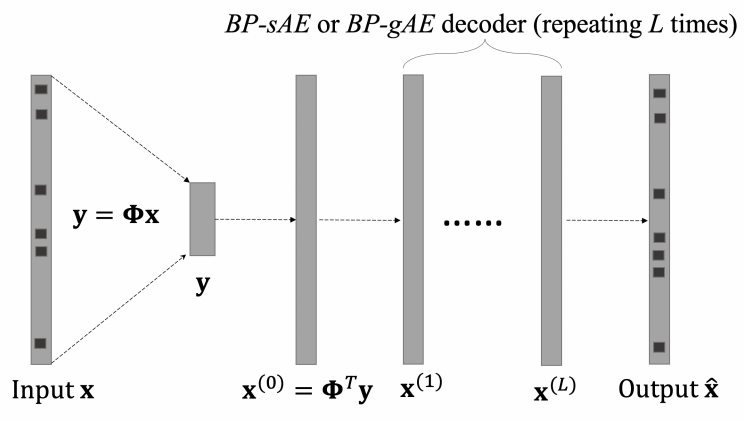


Figure 4.2: Diagram showing the structure of deep *BP-AE*

4.2.2 Specific Structures of Deep Unfolding *BP-AE*

The overall network structure of deep unfolding *BP-AE* is shown in Fig. 4.2. The encoder is simply a linear fully-connected layer without an activation function as shown in (4.6). The measurements $\mathbf{y}$ are the input of the decoder [18]. The first-layer decoder is set as $\mathbf{x}^{(0)} = \mathbf{\Phi}^T\mathbf{y}$. For the decoder layers from $\mathbf{x}^{(1)}$ to $\mathbf{x}^{(L)}$, the network is designed by stacking the unfolding [19] iterations of sparse reconstruction, which can be represented by $\mathbf{x}^{(t+1)} = f_{\mathbf{\Phi}}(\mathbf{x}^{(t)}, \mathbf{y})$ for $0 \leq t \leq L-1$. We derive an iterative solution by solving the basis pursuit sparse reconstruction, then we can specify the decoder structures by unfolding the derived iterations.

To solve the basis pursuit sparse reconstruction problem in (4.7), we adopt the projected subgradient descent method [78], and the update is

[18] We suggest to use noiseless measurements in the proposed autoencoders. We observed that the reconstruction performance negligibly changed after adding an additive Gaussian noise layer to the encoder, but the training time increased significantly.

[19] The method of using an iterative solution to construct a deep neural network is called deep unfolding [64]; this method regards each step of iterations as a single layer of a neural network and is used to transform a traditional iterative algorithm into a stack layers for a neural network. Unlike traditional iterative algorithms having predefined stopping conditions and manually-tuned parameters, the unfolding neural networks have a fixed number of iterative layers and the trainable parameters are learned from the training data.

given by

$$\mathbf{x}^{(t+1)} = \mathit{Proj}\big(\mathbf{x}^{(t)} - \alpha_t \cdot \mathrm{sign}(\mathbf{x}^{(t)})\big) \tag{4.9}$$

where $t > 0$ indicates the tth update, α_t is the step size, $\mathrm{sign}(\cdot)$ represents the sign function that is the subgradient of the ℓ_1-norm term $\|\cdot\|_1$ and $\mathit{Proj}(\cdot)$ represents the projection operation onto the convex set $\{\mathbf{x}' : \mathbf{\Phi x}' = \mathbf{y}\}$. The projection of a vector $\mathbf{z}$ onto the set $\{\mathbf{x}' : \mathbf{\Phi x}' = \mathbf{y}\}$ has the following closed-form solution

$$\mathit{Proj}(\mathbf{z}) = \mathbf{z} + \mathbf{\Phi}^{\dagger}(\mathbf{y} - \mathbf{\Phi z}) \tag{4.10}$$

where $\mathbf{\Phi}^{\dagger} = \mathbf{\Phi}^T(\mathbf{\Phi\Phi}^T)^{-1}$ is the Moore-Penrose pseudo-inverse of $\mathbf{\Phi}$. From (4.9) and (4.10), we substitute the vector $\mathbf{z} = \mathbf{x}^{(t)} - \alpha_t \cdot \mathrm{sign}(\mathbf{x}^{(t)})$ into the projection operation (4.10) and obtain the tth-step update as

$$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + \mathbf{\Phi}^{\dagger}(\mathbf{y} - \mathbf{\Phi x}^{(t)}) - \alpha_t(\mathbf{I} - \mathbf{\Phi}^{\dagger}\mathbf{\Phi}) \cdot \mathrm{sign}(\mathbf{x}^{(t)}). \tag{4.11}$$

Eq. (4.11) is an iterative solution of the basis pursuit sparse reconstruction in (4.7). The step size parameter can be set as $\alpha_t = \alpha/t$ according to the diminishing step size rule [78]. Given a proper starting point $\mathbf{x}^{(1)}$ and a stopping condition, the update (4.11) can be used to obtain the reconstruction $\hat{\mathbf{x}}$. To reduce the intensive computations required to calculate the pseudo-inverse $\mathbf{\Phi}^{\dagger}$, we will simplify the pseudo-inverse computation by replacing $\mathbf{\Phi}^{\dagger}$ with the transpose $\mathbf{\Phi}^T$. Thus, the decoder update $\mathbf{x}^{(t+1)} = f_{\mathbf{\Phi}}(\mathbf{x}^{(t)}, \mathbf{y})$ for $1 \leq t \leq L-1$ can be represented by

$$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + \mathbf{\Phi}^T\mathbf{y} - \mathbf{\Phi}^T\mathbf{\Phi x}^{(t)} - (\alpha/t)(\mathbf{I} - \mathbf{\Phi}^T\mathbf{\Phi}) \cdot \mathrm{sign}(\mathbf{x}^{(t)}). \tag{4.12}$$

We design the decoder structure $\mathbf{x}^{(t+1)} = f_{\mathbf{\Phi}}(\mathbf{x}^{(t)}, \mathbf{y})$ for $0 \leq t \leq L-1$ by unfolding the iterations of (4.12). By specifying the measurements $\mathbf{y}$ used in each step of updates in different ways, we construct two explicit decoder structures named the *BP-sAE* decoder and the *BP-gAE* decoder, which are explained below.

BP-sAE decoder: We specify $\mathbf{y}$ as the measurements only in current

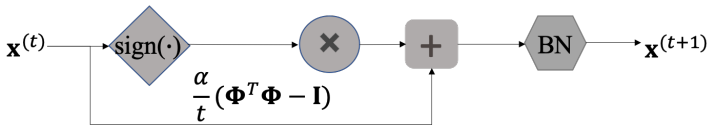


Figure 4.3: Computation graph of a *BP-sAE* decoder; the module BN represents batch normalization

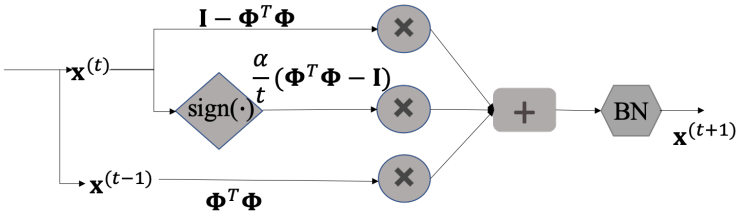


Figure 4.4: Computation graph of a *BP-gAE* decoder; the module BN represents batch normalization

layer, i.e., $\mathbf{y} = \mathbf{\Phi}\mathbf{x}^{(t)}$. By substituting $\mathbf{y} = \mathbf{\Phi}\mathbf{x}^{(t)}$ into the decoder update (4.12), we obtain a simplified update

$$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} - (\alpha/t)(\mathbf{I} - \mathbf{\Phi}^T\mathbf{\Phi}) \cdot \text{sign}(\mathbf{x}^{(t)}). \tag{4.13}$$

Eq. (4.13) is the tth-layer computation for the *BP-sAE* decoder, and its computation graph [20] is shown in Fig. 4.3.

BP-gAE decoder: We specify $\mathbf{y}$ as the measurements obtained in previous layer, i.e., $\mathbf{y} = \mathbf{\Phi}\mathbf{x}^{(t-1)}$. By substituting $\mathbf{y} = \mathbf{\Phi}\mathbf{x}^{(t-1)}$ in the decoder update (4.12), we obtain

$$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + \mathbf{\Phi}^T\mathbf{\Phi}\mathbf{x}^{(t-1)} - \mathbf{\Phi}^T\mathbf{\Phi}\mathbf{x}^{(t)} - (\alpha/t)(\mathbf{I} - \mathbf{\Phi}^T\mathbf{\Phi}) \cdot \text{sign}(\mathbf{x}^{(t)}). \tag{4.14}$$

Eq. (4.14) is the tth-layer computation for the *BP-gAE* decoder, and its computation graph is shown in Fig. 4.4. The advantage of the *BP-gAE* decoder is that we introduce a shortcut from the previous output $\mathbf{x}^{(t-1)}$ by treating the measurements y as $\mathbf{\Phi}\mathbf{x}^{(t-1)}$. We refer to this as pseudo

[20] In deep learning, the computation graph is used to represent a math function in the language of graph theory for visualizing the computation flow.

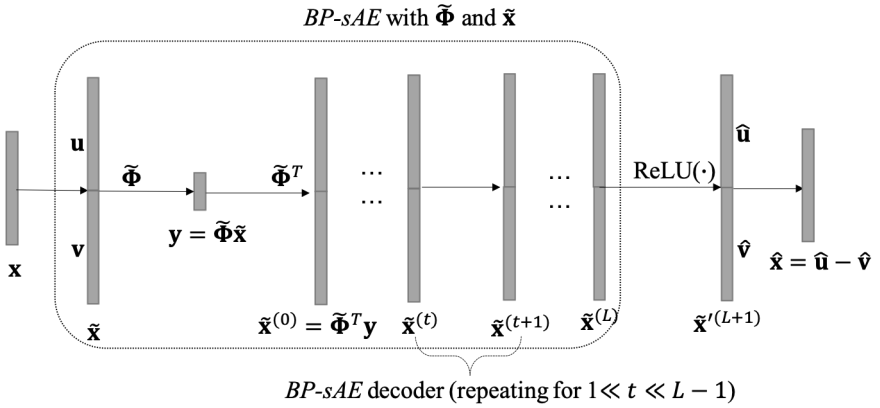


Figure 4.5: Diagram that shows the *BP-sAEcat* extension model; the module BN represents batch normalization and ReLU stands for a rectified linear unit

residual learning and it can improve the autoencoder performance since the introduced special unit $\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{x}^{(t-1)} - \mathbf{\Phi}^T\mathbf{\Phi}\mathbf{x}^{(t)}$can be understood as a special residual learning unit [92].

In the deep unfolding *BP-AE* structure shown in Fig. 4.2, we can specify the decoder computations from $\mathbf{x}^{(1)}$ to $\mathbf{x}^{(L)}$ by repeating the computations of the *BP-sAE* decoder ((4.13) or Fig. 4.3) and the *BP-gAE* decoder ((4.14) or Fig. 4.4) for L times, thus we can construct two autoencoder models called *BP-sAE* and *BP-gAE*. The output layer of the autoencoder is represented by $\hat{\mathbf{x}} = \mathbf{x}^{(L)}$. The loss function is the MSE between input data $\{\mathbf{X}\}$ and output reconstructions $\{\hat{\mathbf{X}}\}$, as shown in (4.8). The trainable variables are $\{\mathbf{\Phi}, \alpha\}$, where $\mathbf{\Phi}$ is configured to be the tied weights of autoencoder and it is also the measurement matrix that we aim to optimize by training the autoencoder.

4.2.3 Extension Models for Learning the Measurement Matrices Having Double Columns

In the previous subsections, we have proposed the *BP-sAE* and *BP-gAE* autoencoders to acquire data-driven measurement matrices having the size

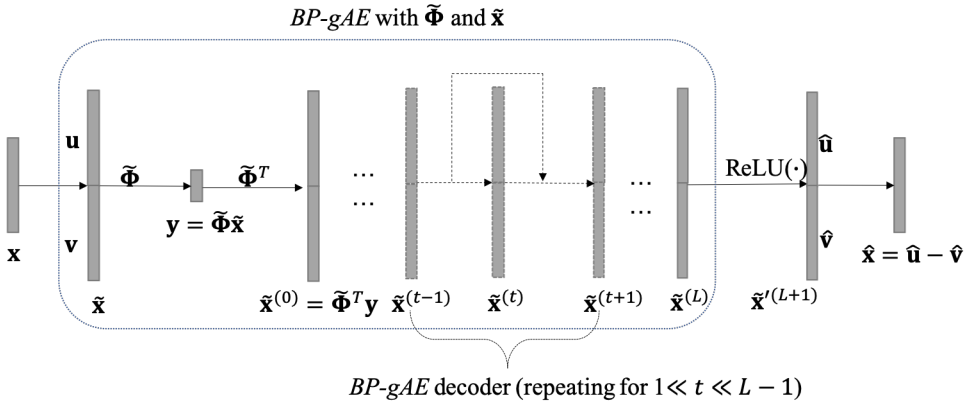


Figure 4.6: Diagram that shows the *BP-gAEcat* extension model; the module BN represents batch normalization and ReLU stands for a rectified linear unit

of $M \times N$, where M is the compressed dimension and N is the dimension of sparse vectors. To further improve the measurement matrix performance, we introduce an artificial feature, i.e., non-negativity, to the dataset and propose two extension models which are referred to as the *BP-sAEcat* and *BP-gAEcat*. Distinct from the *BP-sAE* and *BP-gAE*, the *BP-sAEcat* and *BP-gAEcat* produce measurement matrices having the size of $M \times 2N$.

We perform variable splitting at the input of encoder. The channel vector $\mathbf{x}$ is represented by the difference between its positive part and negative part, i.e., $\mathbf{x} = \mathbf{u} - \mathbf{v}$, where

$$\mathbf{u} = \mathbf{x}_+, \quad \mathbf{v} = (-\mathbf{x})_+, \tag{4.15}$$

where $\mathbf{x}_+$ denotes retaining the positive components of vector $\mathbf{x}$ and setting the other elements as zeros, i.e., $(\mathbf{x}_+)_i = \max\{\mathbf{x}_i, 0\}$ for $i = 1, \cdots, N$; $(-\mathbf{x})_+$ denotes retaining the positive components of the vector $-\mathbf{x}$ and setting the other elements be zeros, i.e., $((-\mathbf{x})_+)_i = \max\{-\mathbf{x}_i, 0\}$ for $i = 1, \cdots, N$. Thus, the problem (4.7) can be rewritten as

$$\begin{aligned} \min_{\mathbf{u},\mathbf{v}} \quad & \|(\mathbf{u} - \mathbf{v})\|_1 \\ \text{s.t.} \quad & \mathbf{\Phi}(\mathbf{u} - \mathbf{v}) = \mathbf{y}, \mathbf{u} \geq \mathbf{0}, \mathbf{v} \geq \mathbf{0}. \end{aligned} \tag{4.16}$$

We use the double-sized channel vector $\tilde{\mathbf{x}}$ to denote the concatenation of $\mathbf{u}$ and $\mathbf{v}$, i.e., $\tilde{\mathbf{x}} = [\mathbf{u}^T, \mathbf{v}^T]^T$, and rewrite the problem (4.16) as

$$\min_{\tilde{\mathbf{x}}} \quad \|\tilde{\mathbf{x}}\|_1 \qquad \text{s.t.} \quad [\mathbf{\Phi}, -\mathbf{\Phi}]\tilde{\mathbf{x}} = \mathbf{y}, \tilde{\mathbf{x}} \geq \mathbf{0}. \tag{4.17}$$

We use the double-column sized matrix $\tilde{\mathbf{\Phi}} \in \mathbb{R}^{M \times 2N}$ to replace the matrix $[\mathbf{\Phi}, -\mathbf{\Phi}]$, where $\mathbf{\Phi} \in \mathbb{R}^{M \times N}$, and the problem (4.17) can be simplified to be

$$\min_{\tilde{\mathbf{x}}} \quad \|\tilde{\mathbf{x}}\|_1 \qquad \text{s.t.} \quad \tilde{\mathbf{\Phi}}\tilde{\mathbf{x}} = \tilde{\mathbf{y}}, \tilde{\mathbf{x}} \geq \mathbf{0}. \tag{4.18}$$

Eq. (4.18) and Eq. (4.7) express the same sparse reconstruction problem, but they have different formulations. First, the variables $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{\Phi}}$ in problem (4.18) are double-sized when compared with the variables $\mathbf{x}$ and $\mathbf{\Phi}$ in problem (4.7). Second, we impose an auxiliary nonnegativity constraint $\tilde{\mathbf{x}} \geq \mathbf{0}$ on problem (4.18).

We apply the autoencoders *BP-sAE* and *BP-gAE* to the intermediate variables $\tilde{\mathbf{x}}$. At the output layer, we first apply the ReLU [21] activation function to $\tilde{\mathbf{x}}^{(L)}$ and obtain

$$\tilde{\mathbf{x}}^{(L+1)} = \text{ReLU}\left(\tilde{\mathbf{x}}^{(L)}\right). \tag{4.19}$$

Then, we treat the first half of $\tilde{\mathbf{x}}^{(L+1)}$ as the reconstruction of the positive part of $\mathbf{x}$, which can be expressed as $\hat{\mathbf{u}} = \hat{\mathbf{x}}_+$. Next, we treat the second half of $\tilde{\mathbf{x}}^{(L+1)}$ as the reconstruction of the negative part of $\mathbf{x}$, which can be expressed as $\hat{\mathbf{v}} = (-\hat{\mathbf{x}})_+$. Thus, we obtain the final reconstruction as

$$\hat{\mathbf{x}} = \hat{\mathbf{u}} - \hat{\mathbf{v}} \tag{4.20}$$

where $\hat{\mathbf{u}}$ is the first-half slice of vector $\tilde{\mathbf{x}}^{(L+1)}$, and $\hat{\mathbf{v}}$ is the second-half slice of vector $\tilde{\mathbf{x}}^{(L+1)}$. Fig. 4.5 and Fig. 4.6 display the structures of the *BP-sAEcat* and *BP-gAEcat* extension autoencoder models respectively. Compared with the data-driven measurement matrix $\mathbf{\Phi} \in \mathbb{R}^{M \times N}$ acquired

[21] ReLU is a type of activation function defined as $\text{ReLU}(x) = \max(0, x)$ when x is a scalar. When the input is a vector, $\text{ReLU}(\cdot)$ is applied in an element-wise manner.

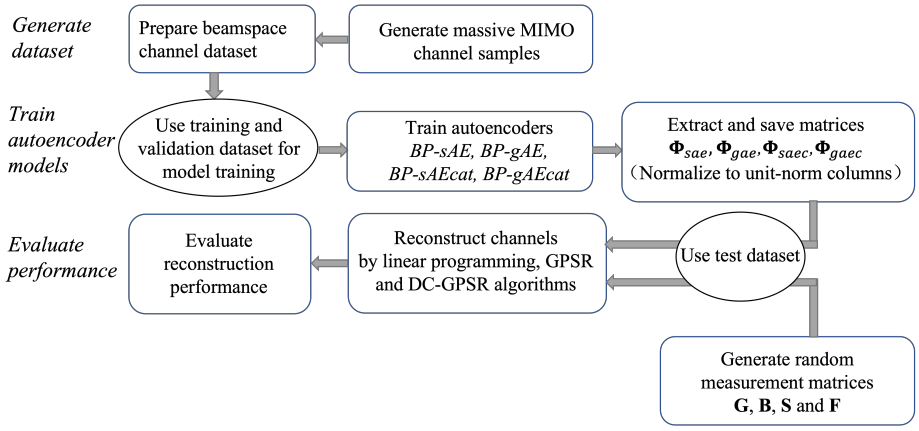


Figure 4.7: Diagram of the performed experiment

by $BP\text{-}sAE$ and $BP\text{-}gAE$, the data-driven measurement matrix $\tilde{\mathbf{\Phi}} \in \mathbb{R}^{M\times 2N}$ acquired by the extension models $BP\text{-}sAEcat$ and $BP\text{-}gAEcat$ have double-sized columns. To use the measurement matrix $\tilde{\mathbf{\Phi}}$ for sparse reconstructions, we only need to perform variable splitting and concatenation at the input as shown in (4.15), and perform slicing and subtraction at the output as shown in (4.20).

4.3 Numerical Results

We have proposed four deep unfolding $BP\text{-}AE$ autoencoder models to acquire data-driven measurement matrices. In this section, we will evaluate the reconstruction performance of the acquired data-driven measurement matrices and compare the results with several common random matrices. To this end, we adopt three sparse reconstruction algorithms including linear programming recovery [93], GPSR [50], and the recently proposed DC-GPSR [94] to solve sparse reconstructions. We also design a toy experiment presented in the Appendix C to show visually that the learned measurement matrix can adapt with the structural feature of the given dataset.

4.3.1 Experiment Setup and Autoencoder Training

The performed experiment [22] contains three phases including dataset generation, model training and performance evaluation. A diagram of the performed experiment is shown in Fig. 4.7. In the phase of dataset generation, we randomly generate $50,000$ spatial-domain channel realizations according to the channel model in (2.10). We consider a massive MIMO system having 256 antennas at the BS and a single antenna at each of the UEs. We generate spatial channels for each user and obtain a dataset containing spatial-domain channel vectors. For each channel, three scattering clusters were selected, including one line-of-sight path and two non-line-of-sight path. The Ricean K-factor of the scattering clusters is set to be 13.2 dB [95]. The AoDs of scattering clusters follow a uniform distribution between $[-\pi/2, \pi/2]$, and the complex path gains follow zero-mean Gaussian distribution. The spatial-domain channel vectors are transformed into sparse beamspace channel vectors with the sparsity level $N_s = 16$ since it was observed that the 16 largest-magnitude channel components are sufficient to occupy most of the channel energy. We stack the real part and imaginary part of each beamspace channel vector column-wise, and obtain a dataset $\{\mathbf{X}\}$ consisting of $100,000$ real-form sparse vectors, which are $\{\mathbf{X}\} = \{[\mathbf{H}_1, ..., \mathbf{H}_{50000}]\} = \{[\Re(\mathbf{h}_{b,1}), \Im(\mathbf{h}_{b,1}), ..., \Re(\mathbf{h}_{b,50000}), \Im(\mathbf{h}_{b,50000})]\}$. We normalize each beamspace channel vector such that the performance is independent of the path loss, i.e., $\|\mathbf{H}_i\|_F = \|\mathbf{h}_{b,i}\|_2 = 1$ for $i \in \{1, ..., 50000\}$. For simplicity, we will use the column vector $\mathbf{x} \in \mathbb{R}^{256}$ to represent an arbitrary sample in the dataset $\{\mathbf{X}\}$. We split the samples of the dataset into training, validation and testing datasets by the ratio of 0.96/0.02/0.02. The training and validation datasets are used to train autoencoder models, and the testing dataset will be used for performance evaluations.

In the model training phase, we train the proposed autoencoder models. The depth of decoder is set to be 15 layers, i.e., $L = 15$. The measurement matrix is initialized with a truncated normal distribution that has a standard

[22] Our experiment was performed on a desktop computer equipped with a 3.2 GHz Intel Core i7-8700 CPU with 8GB of physical memory.

deviation of $\sigma = 1/\sqrt{256}$. The trainable step size parameter α is initialized to be $\alpha = 1.0$, and its value is optimized during training. The SGD is used as the optimizer to train the autoencoders. The training parameters are set as follows: learning rate is 0.01, batch size is 128 and the maximum number of training epochs [23] is set as $1,000$. Reconstruction errors were checked for every 5 epochs over the validation dataset. The tolerance of the training epochs is set to 5 to allow the early stopping of the training process if validation errors cannot be reduced within 5 epochs. The training processes are repeated over different compressed dimensions between $24 \leq M \leq 72$. After training concludes, we extract the weights as the data-driven measurement matrices $\mathbf{\Phi}_{sae}$, $\mathbf{\Phi}_{gae}$, $\mathbf{\Phi}_{saec}$ and $\mathbf{\Phi}_{gaec}$ from the trained autoencoder models *BP-sAE*, *BP-gAE*, *BP-sAEcat* and *BP-gAEcat*, respectively.

In the performance evaluation phase, we evaluate the reconstruction performance by applying the learned data-driven measurement matrices to classical sparse reconstruction algorithms, and the results will be presented in the following sections. The trained autoencoders can be used to perform reconstructions directly and therefore are used to perform sparse channel reconstructions over the testing dataset. The testing NMSE [24] of the proposed autoencoder models for different compressed dimensions M are shown in Table 4.1. We can observe that the test errors decrease when the compressed dimension M increases. The *BP-gAE* has lower test errors among the four types of autoencoders. This result confirms that the proposed residual learning module for *BP-gAE*, i.e., the term $\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{x}^{(t-1)}$ in (4.14), can improve data fitting compared with the *BP-sAE* model. The *BP-gAE* has a test error below 0.01 for $M \geq 64$; therefore, it is valid to perform sparse reconstructions using the trained autoencoder *BP-gAE* when the measurements are sufficient. However, it should be noted that the *BP-AE* models are designed to acquire data-driven measurement matrices, instead of to be used for sparse reconstructions. Therefore, we propose to apply the learned measurement matrices to the classical sparse reconstruction algorithms such

[23] An epoch refers to a full pass of the training algorithm over the entire training dataset.

[24] The NMSE is defined as $\mathbb{E}\left[\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 / \|\mathbf{x}\|_2^2\right]$, where $\mathbf{x}$ is a sample from the testing dataset.

Table 4.1: Reconstruction NMSE Over Testing Dataset of the Proposed Autoencoder Models

NMSE / Model / M	$BP\text{-}sAE$	$BP\text{-}gAE$	$BP\text{-}sAEcat$	$BP\text{-}gAEcat$
24	0.201	0.147	0.047	0.047
32	0.143	0.064	0.037	0.039
40	0.104	0.030	0.038	0.034
48	0.084	0.020	0.027	0.031
56	0.057	0.012	0.027	0.024
64	0.057	**0.009**	0.024	0.022
72	0.041	**0.008**	0.022	0.022

that improved reconstructions can be achieved than using random matrices. The performance of the corresponding reconstructions are displayed and discussed in the following sections.

4.3.2 Reconstructions by Linear Programming Recovery

In this section, we will evaluate reconstruction performance by linear programming recovery using various measurement matrices over the testing dataset. We denote $\mathbf{\Phi}_{sae} \in \mathbb{R}^{M\times 256}$, $\mathbf{\Phi}_{gae} \in \mathbb{R}^{M\times 256}$, $\mathbf{\Phi}_{saec} \in \mathbb{R}^{M\times 512}$ and $\mathbf{\Phi}_{gaec} \in \mathbb{R}^{M\times 512}$ as the learned data-driven matrices by $BP\text{-}sAE$, $BP\text{-}gAE$, $BP\text{-}sAEcat$ and $BP\text{-}gAEcat$ respectively. We choose four types of random matrices for comparisons including the random Gaussian matrix, the random Bernoulli matrix [25], the partial Fourier matrix, and the random selection matrix [26]. For the learned matrices $\mathbf{\Phi}_{sae}$, $\mathbf{\Phi}_{gae}$ and the random Gaussian matrix $\mathbf{G} \in \mathbb{R}^{M\times 256}$, the random Bernoulli matrix $\mathbf{B} \in \mathbb{R}^{M\times 256}$, the random selection matrix $\mathbf{S} \in \mathbb{R}^{M\times 256}$ and the partial Fourier matrix $\mathbf{F} \in \mathbb{R}^{M\times 256}$, we use the linear programming method to solve the sparse reconstruction problem in (4.7). For the learned matrices $\mathbf{\Phi}_{saec}$, $\mathbf{\Phi}_{gaec}$ and the random Gaussian matrix $\mathbf{G}_c \in \mathbb{R}^{M\times 512}$, random Bernoulli matrix $\mathbf{B}_c \in \mathbb{R}^{M\times 512}$, random selection matrix $\mathbf{S}_c \in \mathbb{R}^{M\times 512}$ and the partial Fourier matrix $\mathbf{F}_c \in \mathbb{R}^{M\times 512}$, we

[25] For a random Bernoulli matrix, entries are -1 or 1 with equal probability.
[26] For a random selection matrix, entries are 0 or 1 with equal probability.

Table 4.2: Reconstruction NMSE by Linear Programming Recovery with Different Compressed Dimensions M for Various Measurement Matrices

NMSE \ M / Matrix	32	40	48	56	64	72
$\mathbf{\Phi}_{sae}$	**0.01**	**0.006**	**0.003**	**0.0006**	**0.0005**	$\mathbf{9 \cdot 10^{-5}}$
$\mathbf{\Phi}_{gae}$	**0.01**	**0.003**	**0.001**	**0.0005**	$\mathbf{7 \cdot 10^{-6}}$	$\mathbf{2 \cdot 10^{-6}}$
$\mathbf{G}$	0.1	0.07	0.04	0.02	0.006	0.0006
$\mathbf{B}$	0.1	0.06	0.04	0.02	0.005	0.0005
$\mathbf{F}$	0.1	0.06	0.02	0.07	0.07	0.02
$\mathbf{S}$	0.1	0.06	0.04	0.02	0.005	0.0004
$\mathbf{\Phi}_{saec}$	**0.06**	**0.03**	**0.004**	**0.001**	$\mathbf{3 \cdot 10^{-15}}$	$\mathbf{3 \cdot 10^{-15}}$
$\mathbf{\Phi}_{gaec}$	**0.004**	**0.0006**	$\mathbf{5 \cdot 10^{-5}}$	$\mathbf{5 \cdot 10^{-30}}$	$\mathbf{5 \cdot 10^{-30}}$	$\mathbf{6 \cdot 10^{-30}}$
$\mathbf{G}_c$	0.1	0.07	0.04	0.02	0.006	0.0009
$\mathbf{B}_c$	0.1	0.07	0.04	0.02	0.004	0.0005
$\mathbf{F}_c$	0.1	0.07	0.05	0.06	$1 \cdot 10^{-6}$	0.02
$\mathbf{S}_c$	0.2	0.09	0.05	0.02	0.006	0.0007

use linear programming with an auxiliary nonnegativity constraint to solve the sparse reconstruction problem in (4.18).

Table 4.2 shows the reconstruction NMSE for different matrices with different compressed dimensions M. The NMSE generally decreases when the compressed dimension M increases for all matrices. The learned data-driven matrices $\mathbf{\Phi}_{sae}$, $\mathbf{\Phi}_{gae}$, $\mathbf{\Phi}_{saec}$ and $\mathbf{\Phi}_{gaec}$ significantly outperform the random matrices for all compressed dimensions of M in terms of reconstruction NMSE. The learned matrix $\mathbf{\Phi}_{gae}$ has lower reconstruction NMSE than the learned matrix $\mathbf{\Phi}_{sae}$; the learned matrices $\mathbf{\Phi}_{saec}$ and $\mathbf{\Phi}_{gaec}$ have lower reconstruction errors than the learned matrices $\mathbf{\Phi}_{sae}$ and $\mathbf{\Phi}_{gae}$. The learned matrix $\mathbf{\Phi}_{gaec}$ can reconstruct accurately with an NMSE on the order of 10^{-30} when the compressed dimensions are $M = 48$, $M = 56$ and $M = 72$. Table 4.3 shows the accurate reconstruction percentages, which refers to the ratio of the number of accurate reconstructed samples over the total number of samples in the testing dataset. We consider that a sample is accurately reconstructed when the normalized squared ℓ_2-error

Table 4.3: Accurate Reconstruction Percentages by Linear Programming Recovery with Different Compressed Dimensions M for Various Measurement Matrices

Percentage (%) \ M / Matrix	24	32	40	48	56	64	72
$\mathbf{\Phi}_{sae}$	**11.2**	**44.2**	**76.6**	**89**	**97.4**	**99.3**	**99.6**
$\mathbf{\Phi}_{gae}$	**11.9**	**53.8**	**78.5**	**92.5**	**98**	**99.8**	**99.9**
$\mathbf{G}$	0	0	0.1	1.6	32.1	61.9	95.7
$\mathbf{B}$	0	0	0	3.3	32.8	70	95.2
$\mathbf{F}$	0.1	0.1	0.1	16.7	0.3	0.0	8.5
$\mathbf{S}$	0	0	0.1	4.6	24.4	74.7	98.1
$\mathbf{\Phi}_{saec}$	**16.9**	**61.9**	**87.3**	**98.9**	**99.9**	**100**	**100**
$\mathbf{\Phi}_{gaec}$	**21.9**	**60.1**	**90.7**	**98.1**	**100**	**100**	**100**
$\mathbf{G}_c$	0	0	0	1.8	21.8	68.4	92.4
$\mathbf{B}_c$	0	0	0	1.9	32.5	77.7	96
$\mathbf{F}_c$	0	0	1	0.7	0.1	99.8	31.8
$\mathbf{S}_c$	0	0	0	1.6	22.5	67.5	95.7

of its reconstruction is no more than 10^{-8}, i.e., $\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 \leq 10^{-8}$. In the small compressed dimension range, such as for $24 \leq M \leq 40$, the random matrices cannot reconstruct accurately and show almost no accurate reconstructions, while the data-driven measurement matrices have significantly higher accurate reconstruction percentages. For example, the learned matrices $\mathbf{\Phi}_{gae}$, $\mathbf{\Phi}_{saec}$ and $\mathbf{\Phi}_{gaec}$ can achieve over 50% accurate recoveries for compressed dimension $M = 32$, whereas the random matrices have the percentages which are almost all zeros. When $M = 40$ the data-driven matrix $\mathbf{\Phi}_{gaec}$ achieves over 90% accurate reconstructions, while the percentages for random matrices are all below 1%. From Table 4.2 and Table 4.3, we can infer that the learned matrices can achieve more accurate reconstructions using fewer measurements. At the same level of reconstruction accuracy, the learned data-driven matrices can reduce the requiring measurements by approximately $2N_s$, where $N_s = 16$ represents the sparsity level. For example, in Table 4.2, random matrices can achieve an NMSE on the order of 10^{-3} when $M = 64$, while the learned matrix $\mathbf{\Phi}_{gaec}$ can achieve the same

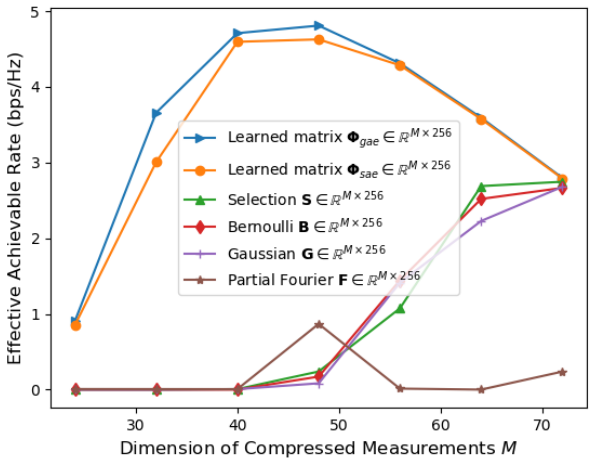


Figure 4.8: Effective achievable rates versus compressed dimension M for different measurement matrices

accuracy at $M = 32$. In Table 4.3, the learned matrix $\mathbf{\Phi}_{gaec}$ can achieve a reconstruction percentage over 90% at the compressed dimension $M = 40$, while the required least number of measurements is $M = 64$ for the random matrix $\mathbf{F}_c$ and is $M = 72$ for the other types of random matrices.

A larger compressed dimension M will lead to higher reconstruction accuracy, but a lower spectrum efficiency. To analyze the tradeoff between compressed dimension M and recovery accuracy, we define the effective achievable rate as $R_e = R_0(1 - M/N_c)P_r$ [37], where R_0 is the average achievable rate for a user when perfect CSI is available and it is calculated to be $R_0 = 10$ bps/Hz, M/N_c is the pilot occupation ratio in a transmission block, N_c is the block length that is set as 100 symbols, and P_r is the probability of successful recoveries. The effective achievable rates for various measurement matrices are shown in Fig. 4.8 and Fig. 4.9. The learned data-driven matrices significantly outperform random matrices in terms of effective achievable rates. In particular, the learned matrices $\mathbf{\Phi}_{sae}$ and $\mathbf{\Phi}_{gae}$ have higher effective achievable rates than random matrices $\mathbf{G}$, $\mathbf{B}$, $\mathbf{F}$ and $\mathbf{S}$, and the learned matrices $\mathbf{\Phi}_{saec}$ and $\mathbf{\Phi}_{gaec}$ have higher effective achievable

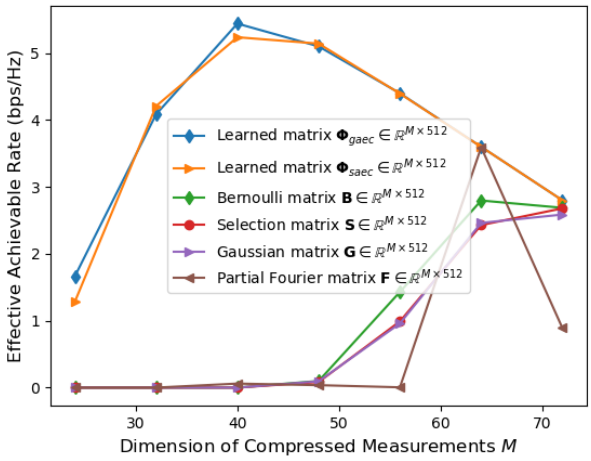


Figure 4.9: Effective achievable rates versus compressed dimension M for different measurement matrices

rates than random matrices $\mathbf{G}_c$, $\mathbf{B}_c$, $\mathbf{F}_c$ and $\mathbf{S}_c$. However, it should be noted that the learned matrices $\mathbf{\Phi}_{saec}$ and $\mathbf{\Phi}_{gaec}$ have higher effective achievable rates than the learned matrices $\mathbf{\Phi}_{sae}$ and $\mathbf{\Phi}_{gae}$. Moreover, the maximum effective achievable rates occur at smaller compressed dimensions M for the learned matrices when compared with random matrices.

4.3.3 Reconstructions Using the DC-GPSR Algorithm

The DC-GPSR algorithm [94] was used to perform reconstructions with different measurement matrices after choosing a sample of a beamspace channel. In Fig. 4.10, we plot the magnitudes of the original channel and reconstructed channels using the learned matrices $\mathbf{\Phi}_{sae} \in \mathbb{R}^{32\times256}$ and $\mathbf{\Phi}_{gae} \in \mathbb{R}^{32\times256}$, the random Gaussian matrix $\mathbf{G} \in \mathbb{R}^{32\times256}$ and random Bernoulli matrix $\mathbf{B} \in \mathbb{R}^{32\times256}$. The normalized squared ℓ_2-error is on the order of 10^{-2} for the random Bernoulli and Gaussian matrices, while the errors are on the order of 10^{-3} for the learned matrix $\mathbf{\Phi}_{sae}$ and are on the order of 10^{-6} for the learned matrix $\mathbf{\Phi}_{gae}$. In Fig. 4.11, we show the magnitudes of the original channel and reconstructed channels using the learned

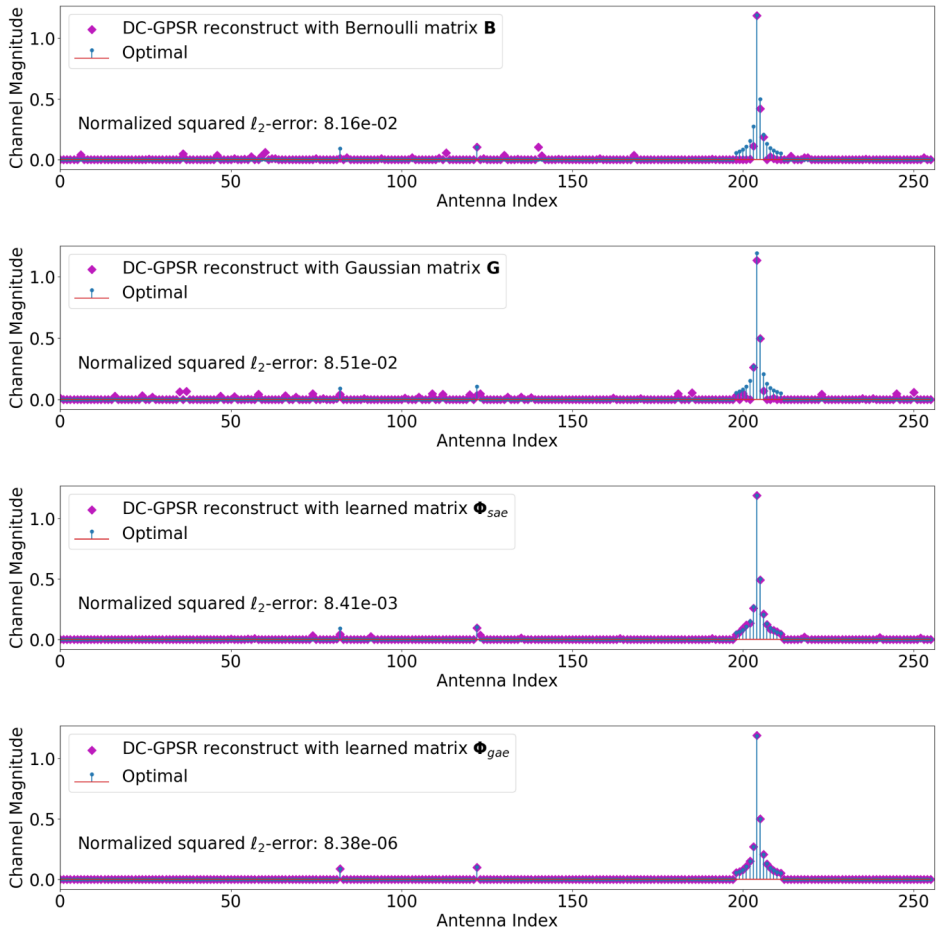


Figure 4.10: Sparse channel reconstruction illustrations produced using the DC-GPSR algorithm with measurement matrices of the size 32×256

matrices $\mathbf{\Phi}_{saec} \in \mathbb{R}^{48\times512}$, $\mathbf{\Phi}_{gaec} \in \mathbb{R}^{48\times512}$, the random Gaussian matrix $\mathbf{G} \in \mathbb{R}^{48\times512}$ and random Bernoulli matrix $\mathbf{B} \in \mathbb{R}^{48\times512}$. The normalized squared ℓ_2-errors are on the orders of 10^{-1} and 10^{-2} respectively for the random Bernoulli and Gaussian matrices, whereas the reconstructions are much more accurate using the learned matrices $\mathbf{\Phi}_{gaec}$ and $\mathbf{\Phi}_{saec}$, which achieve normalized squared ℓ_2-errors on the order of 10^{-32} and 10^{-33}.

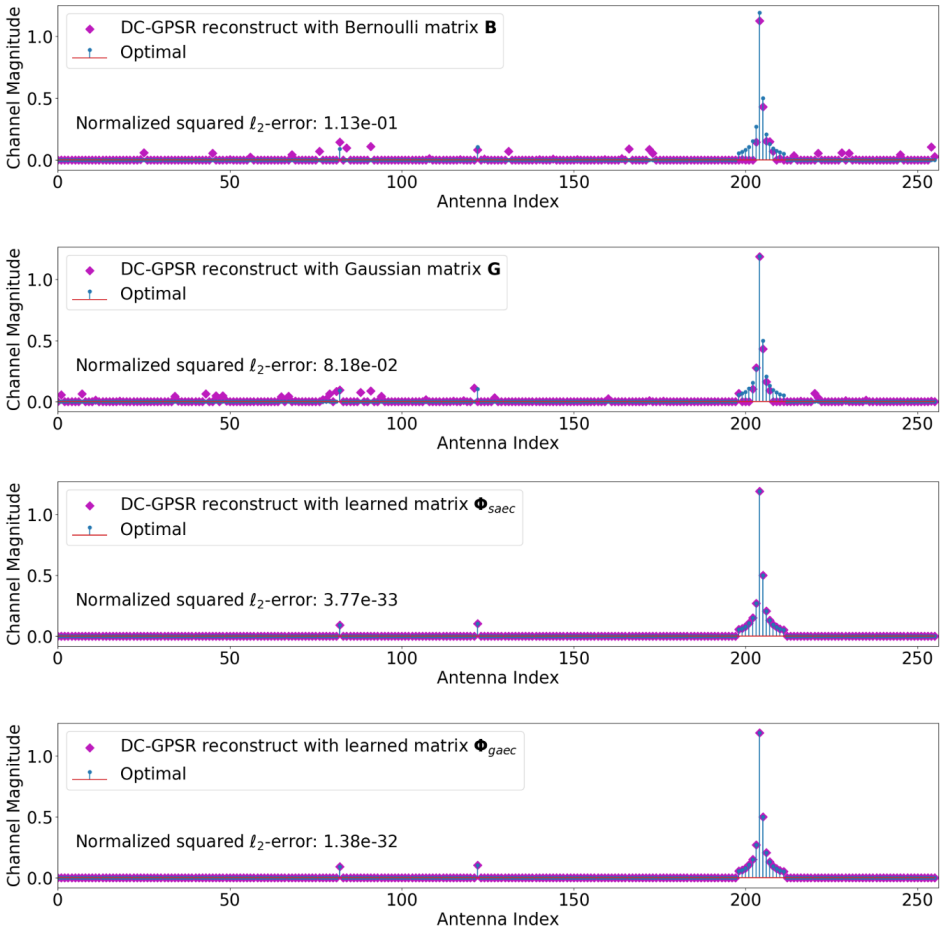


Figure 4.11: Sparse channel reconstruction illustrations produced using the DC-GPSR algorithm with measurement matrices of the size 48×512

4.3.4 Reconstructions Using the GPSR Algorithm

The GPSR algorithm was adopted to perform reconstructions in noisy scenarios using $1,000$ channel samples. The results of NMSE versus SNR are plotted in Fig. 4.12 - Fig. 4.15 for the compressed dimensions $M \in \{32, 40, 56, 72\}$. The GPSR algorithm was proposed to solve the ℓ_1-norm constrained LS optimization for sparse reconstructions [50]. By comparing the four plots in Fig. 4.12 - Fig. 4.15, the NMSE is generally more accurate

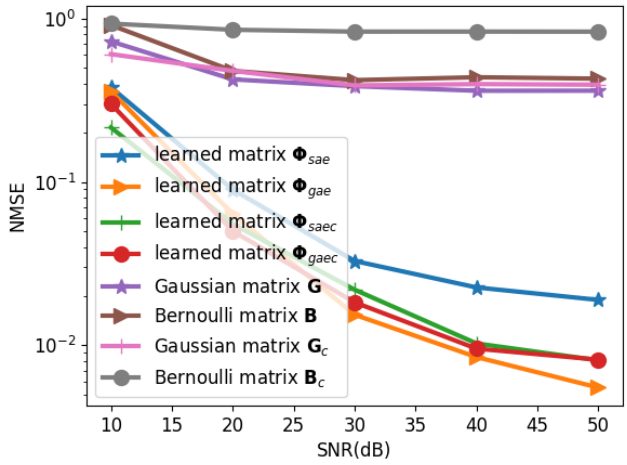


Figure 4.12: Reconstruction NMSE versus SNR produced using the GPSR algorithm at compressed dimensions $M = 32$ for the learned matrices $\mathbf{\Phi}_{sae} \in \mathbb{R}^{M\times 256}$, $\mathbf{\Phi}_{gae} \in \mathbb{R}^{M\times 256}$, $\mathbf{\Phi}_{saec} \in \mathbb{R}^{M\times 512}$, $\mathbf{\Phi}_{gaec} \in \mathbb{R}^{M\times 512}$ and random matrices $\mathbf{G} \in \mathbb{R}^{M\times 256}$, $\mathbf{B} \in \mathbb{R}^{M\times 256}$, $\mathbf{G}_c \in \mathbb{R}^{M\times 512}$, $\mathbf{B}_c \in \mathbb{R}^{M\times 512}$

at a larger compressed dimension for both the learned matrices and the random matrices. For all of the compressed dimensions, the learned matrices show lower reconstruction errors than the random matrices. As shown in Fig. 4.12 and Fig. 4.13, at small compressed dimensions such as $M = 32$ and $M = 40$, the learned matrices show distinct reconstructions errors, and the learned matrix $\mathbf{\Phi}_{gae}$ shows the best reconstruction accuracy. As shown in Fig. 4.14 and Fig. 4.15, at large compressed dimensions such as $M = 56$ and $M = 72$, the learned matrices show similar reconstruction accuracy, and their NMSE plots coincide together. The random matrices can achieve lower reconstruction errors at larger compressed dimensions; but even at $M = 72$, the random matrices have less accurate reconstructions than the learned data-driven matrices.

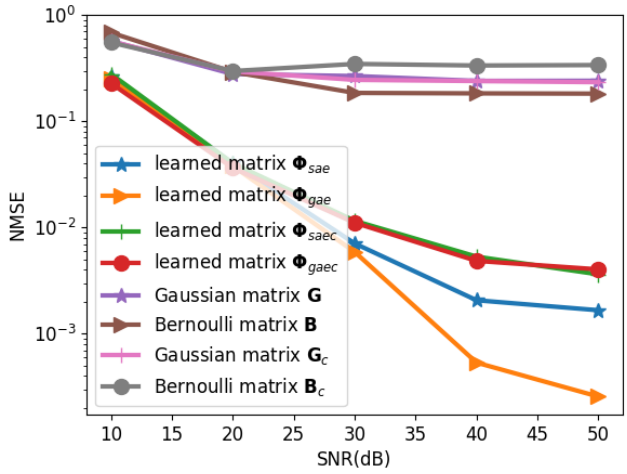


Figure 4.13: Reconstruction NMSE versus SNR produced using the GPSR algorithm at compressed dimensions $M = 40$ for the learned matrices $\mathbf{\Phi}_{sae} \in \mathbb{R}^{M\times 256}$, $\mathbf{\Phi}_{gae} \in \mathbb{R}^{M\times 256}$, $\mathbf{\Phi}_{saec} \in \mathbb{R}^{M\times 512}$, $\mathbf{\Phi}_{gaec} \in \mathbb{R}^{M\times 512}$ and random matrices $\mathbf{G} \in \mathbb{R}^{M\times 256}$, $\mathbf{B} \in \mathbb{R}^{M\times 256}$, $\mathbf{G}_c \in \mathbb{R}^{M\times 512}$, $\mathbf{B}_c \in \mathbb{R}^{M\times 512}$

4.3.5 Complexity-Accuracy Tradeoffs of the Learned Matrices for Practical Applications

We have proposed four autoencoder models to acquire different data-driven matrices. In practical applications, we need to consider the tradeoffs between the computational complexity and the reconstruction performance to choose an appropriate data-driven measurement matrix. In terms of reconstruction performance, the matrices $\mathbf{\Phi}_{saec}$ and $\mathbf{\Phi}_{gaec}$ perform similarly and they can achieve better reconstructions than $\mathbf{\Phi}_{sae}$ and $\mathbf{\Phi}_{gae}$. One plausible reason is that the larger-size matrices $\mathbf{\Phi}_{saec}$ and $\mathbf{\Phi}_{gaec}$ have more variables to be optimized in the neural network, which means more degrees of freedom. Another reason is that the introduced nonnegativity constraint $\tilde{\mathbf{x}} \geq \mathbf{0}$ in (4.18) provides an obvious feature that can be more easily learned by the neural network, and the corresponding adaptation to this auxiliary data feature can be automatically adapted to by the measurement matrices $\mathbf{\Phi}_{saec}$ and $\mathbf{\Phi}_{gaec}$. On the other hand, the reconstruction of the matrix $\mathbf{\Phi}_{gae}$

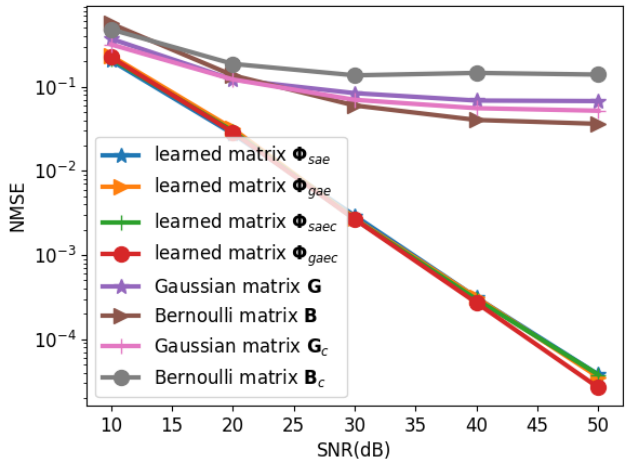


Figure 4.14: Reconstruction NMSE versus SNR by GPSR algorithm at compressed dimensions $M = 56$ for the learned matrices $\mathbf{\Phi}_{sae} \in \mathbb{R}^{M\times 256}$, $\mathbf{\Phi}_{gae} \in \mathbb{R}^{M\times 256}$, $\mathbf{\Phi}_{saec} \in \mathbb{R}^{M\times 512}$, $\mathbf{\Phi}_{gaec} \in \mathbb{R}^{M\times 512}$ and random matrices $\mathbf{G} \in \mathbb{R}^{M\times 256}$, $\mathbf{B} \in \mathbb{R}^{M\times 256}$, $\mathbf{G}_c \in \mathbb{R}^{M\times 512}$, $\mathbf{B}_c \in \mathbb{R}^{M\times 512}$

shows noticeable improvements unlike the matrix $\mathbf{\Phi}_{sae}$. This result confirms that our proposed residual learning module in *BP-gAE* can improve the measurement matrix optimization.

The training complexities for acquiring these four types of data-driven matrices are $\mathbf{\Phi}_{sae} < \mathbf{\Phi}_{gae} < \mathbf{\Phi}_{saec} < \mathbf{\Phi}_{gaec}$. Apart from training complexity, another issue to be considered is the computational complexity in compressive sensing reconstruction algorithms. For linear programming recovery, the matrices $\mathbf{\Phi}_{saec}$ and $\mathbf{\Phi}_{gaec}$ are twice as difficult to compute than the matrices $\mathbf{\Phi}_{sae}$ and $\mathbf{\Phi}_{gae}$ since the optimization problem requires solving dimensions that are twice as big. However, when we use the GPSR and DC-GPSR algorithms for reconstructions, the computational complexities for the four types of learned matrices are the same. We also should consider the dimension limitations when using the matrices in specific applications. For example, the downlink pilot matrix design requires the matrix size to be $M \times N$ to match the unknown channel vectors of the dimension N, whereas the CSI feedback compression can use matrices having sizes of both $M \times N$

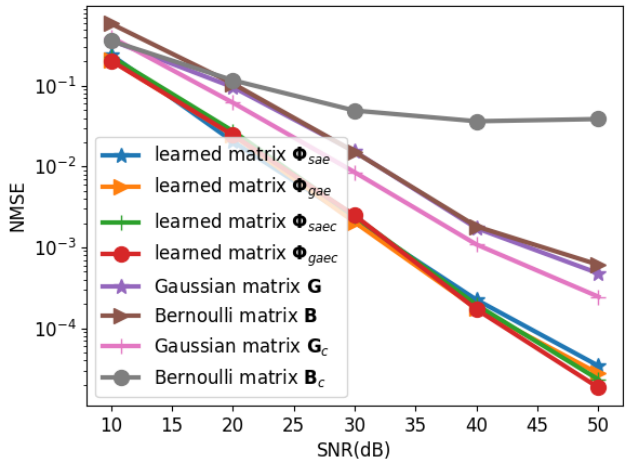


Figure 4.15: Reconstruction NMSE versus SNR by GPSR algorithm at compressed dimensions $M = 72$ for the learned matrices $\mathbf{\Phi}_{sae} \in \mathbb{R}^{M\times 256}$, $\mathbf{\Phi}_{gae} \in \mathbb{R}^{M\times 256}$, $\mathbf{\Phi}_{saec} \in \mathbb{R}^{M\times 512}$, $\mathbf{\Phi}_{gaec} \in \mathbb{R}^{M\times 512}$ and random matrices $\mathbf{G} \in \mathbb{R}^{M\times 256}$, $\mathbf{B} \in \mathbb{R}^{M\times 256}$, $\mathbf{G}_c \in \mathbb{R}^{M\times 512}$, $\mathbf{B}_c \in \mathbb{R}^{M\times 512}$

and $M \times 2N$, since the UE can transform the known beamspace channel vectors into $\tilde{\mathbf{h}}_b = [(\mathbf{h})_{b,+}^T, (-\mathbf{h})_{b,+}^T]^T$ of the dimension $2N$; then we can use a measurement matrix of the size $M \times 2N$ to compress the channel vectors. Essentially, the matrices $\mathbf{\Phi}_{saec}$ and $\mathbf{\Phi}_{gaec}$ are suggested to be used in CSI feedback and the matrices $\mathbf{\Phi}_{sae}$ and $\mathbf{\Phi}_{gae}$ are suitable to be used for downlink pilot designs.

4.4 Summary

We proposed a data-driven approach to acquire measurement matrices. Specifically, we proposed a generic model-based autoencoder framework called deep unfolding *BP-AE* that consists of a linear dimensional-reduction encoder and a decoder that stacks the iterations to mimic basis pursuit sparse reconstructions. Under this framework, we constructed two model-based autoencoders *BP-sAE* and *BP-gAE* as well as two extension models *BP-sAEcat* and *BP-gAEcat*. The proposed autoencoder models

are parameterized by the measurement matrix so that they can be trained to optimize the measurement matrices given the sparse beamspace channel dataset. We provided numerical demonstrations to show that the learned data-driven measurement matrices can work well with general compressive sensing reconstruction algorithms, such as the linear programming, GPSR and DC-GPSR algorithms. The learned data-driven matrices can attain higher achievable rates for CSI acquisitions than conventional random matrices, because more accurate reconstructions can be accomplished using fewer measurements. Therefore, the learned data-driven measurement matrices can be applied to design pilots or compress CSI feedback to reduce the overheads of downlink CSI acquisitions. This work demonstrates a useful application of deep learning techniques to design massive MIMO systems. By exploiting the data-driven method to improve the measurement matrix for traditional compressive sensing and reconstructions, we believe that our study opens up new perspectives for parameter optimizations of classical model-based algorithms using deep learning.

Chapter 5

Joint Pilot Design and Channel Estimation via Deep Compressive Sensing

In Chapter 3 and Chapter 4, it was shown that sparse channel reconstructions can be improved by optimizing either the sparse reconstruction algorithm or the measurement matrix. Using a standard DFT dictionary for sparse representation, designing a measurement matrix is equivalent to designing the pilot matrix. However, individual optimization on the reconstruction algorithm or the pilot design does not necessarily lead to optimal results on channel estimations. In this chapter, we explore the data-driven method for joint pilot design and sparse reconstructions. We first propose novel iterative algorithms for sparse reconstruction, then we unfold the proposed iterative algorithms into deep networks to construct model-based autoencoders. The trained autoencoders can be used to acquire the data-driven measurement matrix and estimate the channels in real-time.

5.1 Trimmed Ridge Regression Algorithms for Sparse Reconstructions

In this section, we propose to use the trimmed ridge regression for sparse reconstruction, followed by a derivation of iterative algorithms to solve the trimmed ridge regression. Different from the common ridge regression, the trimmed ridge regression has a trimmed ℓ_2-norm regularizer that removes the penalties on a few large-magnitude elements to achieve sparse solutions. The

trimmed ridge regression is a non-convex optimization, and we solve it using DC programming and gradient projection descent. We first derive a single-loop iterative solution via a special DC decomposition. Then, we design two practical step size strategies and propose two accelerated algorithms based on the derived iterative solution. Finally, the proposed accelerated algorithms are unfolded into deep networks to construct autoencoders, which will be presented in the next section.

5.1.1 Trimmed Ridge Regression for Sparse Reconstruction

As described in Section 4.1, the sparse channel reconstruction problem reduces to the SMV problems in (4.3). We generally express the SMV problem as

$$\mathbf{y} = \mathbf{\Phi}\mathbf{x} + \mathbf{n} \tag{5.1}$$

where $\mathbf{y} \in \mathbb{R}^M$, $\mathbf{\Phi} \in \mathbb{R}^{M\times N}$, $\mathbf{x} \in \mathbb{R}^N$, $\mathbf{n} \in \mathbb{R}^M$, and $M \ll N$. Here, we use the vector $\mathbf{x}$ to denote the real part or the imaginary part of a beamspace channel vector uniquely, i.e., $\mathbf{x} = \Re(\mathbf{h})$ or $\mathbf{x} = \Im(\mathbf{h})$. The underdetermined linear system (5.1) can be solved using the following constrained LS optimization

$$\begin{aligned} \min_{\mathbf{x}} \quad & \|\mathbf{y} - \mathbf{\Phi}\mathbf{x}\|_2^2 \\ \text{s.t.} \quad & \|\mathbf{x}\|_0 \le K. \end{aligned} \tag{5.2}$$

The ℓ_0-norm constraint $\|\mathbf{x}\|_0 \le K$ in (5.2) results in an NP-hard optimization.

We introduce the top-$(K,2)$ norm $\|\mathbf{x}\|_{(K,2)}^2$ to replace the ℓ_0-norm constraint $\|\mathbf{x}\|_0 \le K$ [96]. The top-$(K,2)$ norm is the sum of squares of the K elements that have largest magnitudes for the vector $\mathbf{x}$, i.e.,

$$\|\mathbf{x}\|_{(K,2)}^2 := x_{(1)}^2 + x_{(2)}^2 + \cdots + x_{(K)}^2 \tag{5.3}$$

where $x_{(i)}$ denotes the element whose magnitude is the ith-largest among the N elements of the vector $\mathbf{x}$, i.e., $|x_{(1)}| \ge |x_{(2)}| \ge \cdots \ge |x_{(K)}|$. By using the top-$(K,2)$ norm, we have $\|\mathbf{x}\|_2^2 - \|\mathbf{x}\|_{(K,2)}^2 = 0 \leftrightarrow \|\mathbf{x}\|_0 \le K$. Thus, we

can rewrite the sparse reconstruction optimization in (5.2) as

$$\begin{aligned} \min_{\mathbf{x}} \quad & \|\mathbf{y}-\mathbf{\Phi}\mathbf{x}\|_2^2 \\ \text{s.t.} \quad & \|\mathbf{x}\|_2^2 - \|\mathbf{x}\|_{(K,2)}^2 = 0. \end{aligned} \tag{5.4}$$

By using the Lagrange multiplier ρ, the constrained optimization (5.4) can be transformed into an unconstrained optimization problem

$$\min_{\mathbf{x}} \quad \frac{1}{2}\|\mathbf{y}-\mathbf{\Phi}\mathbf{x}\|_2^2 + \rho(\|\mathbf{x}\|_2^2 - \|\mathbf{x}\|_{(K,2)}^2) := F(\mathbf{x}). \tag{5.5}$$

The optimization in (5.5) is referred to as the trimmed ridge regression. The trimmed ridge regression is similar to a ridge regression, which is $\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{y}-\mathbf{\Phi}\mathbf{x}\|_2^2 + \rho\|\mathbf{x}\|_2^2$, but they have different regularizers. The trimmed ridge regression benefits from the novel regularizer to achieve sparse and accurate solutions since it removes the penalties on K largest-magnitude elements of the reconstructing vector.

5.1.2 Solving Trimmed Ridge Regression Under the DC Programming Framework

We perform a special DC decomposition and derive a single-loop iterative solution to the trimmed ridge regression by applying the gradient projection descent under the DC programming framework. The trimmed ridge regression in (5.5) can be equivalently written as

$$F(\mathbf{x}) = \frac{l}{2}\|\mathbf{x}\|_2^2 - \frac{l}{2}\|\mathbf{x}\|_2^2 + \frac{1}{2}\|\mathbf{y}-\mathbf{\Phi}\mathbf{x}\|_2^2 + \rho(\|\mathbf{x}\|_2^2 - \|\mathbf{x}\|_{K,2}^2) \tag{5.6}$$

where $l \geq 0$ is a Lipschitz constant of the least square objective $\frac{1}{2}\|\mathbf{y}-\mathbf{A}\mathbf{x}\|_2^2$. Then, we perform the following DC decomposition on the objective function in (5.6), and obtain

$$\min_{\mathbf{x}} \underbrace{\frac{l}{2}\|\mathbf{x}\|_2^2 + \rho\|\mathbf{x}\|_2^2}_{f(\mathbf{x})} - \underbrace{\left(\frac{l}{2}\|\mathbf{x}\|_2^2 - \frac{1}{2}\|\mathbf{y}-\mathbf{\Phi}\mathbf{x}\|_2^2 + \rho\|\mathbf{x}\|_{K,2}^2\right)}_{g(\mathbf{x})} \tag{5.7}$$

where the functions $f(\mathbf{x})$ and $g(\mathbf{x})$ are convex. The convexity of $g(\mathbf{x})$ can be ensured by confirming the convexity of $\frac{l}{2}\|\mathbf{x}\|_2^2 - \frac{1}{2}\|\mathbf{y} - \mathbf{\Phi}\mathbf{x}\|_2^2$, which is shown in Appendix B.

Next, we split the positive and negative parts of $\mathbf{x}$ by letting $\mathbf{u} = (\mathbf{x})_+, \mathbf{v} = (-\mathbf{x})_+$, where $\mathbf{u} = (\mathbf{x})_+, \mathbf{v} = (-\mathbf{x})_+$, and where $(\cdot)_+$ is the positive-taking operation that retains the positive elements and sets the other elements to be zeros. More precisely, $(\mathbf{x})_+$ represents the operation of $(x)_+ = \max\{0, x\}$ for each element x in vector $\mathbf{x}$; $(-\mathbf{x})_+$ represents the operation of $(-x)_+ = \max\{0, -x\}$ for each element $-x$ in vector $-\mathbf{x}$. By denoting $\mathbf{z} = [\mathbf{u}^T, \mathbf{v}^T]^T$, $\mathbf{A} = [\mathbf{\Phi}, -\mathbf{\Phi}]$ we can express (5.7) as

$$\min_{\mathbf{z}\succeq\mathbf{0}} \quad \underbrace{\frac{l}{2}\|\mathbf{z}\|_2^2 + \rho\|\mathbf{z}\|_2^2}_{f(\mathbf{z})} - \underbrace{\left(\frac{l}{2}\|\mathbf{z}\|_2^2 - \frac{1}{2}\|\mathbf{y} - \mathbf{A}\mathbf{z}\|_2^2 + \rho\|\mathbf{z}\|_{K,2}^2\right)}_{g(\mathbf{z})}. \tag{5.8}$$

To solve (5.8), we apply the DC programming procedure by computing the gradient (or subgradient) $\partial g(\mathbf{z}^{(t)})$ for an arbitrary tth iteration and then solve the convex subproblem $\min_{\mathbf{z}\succeq\mathbf{0}} f(\mathbf{z}) - \mathbf{z}^T\partial g(\mathbf{z}^{(t)})$ to obtain $\mathbf{z}^{(t+1)}$. In other words, we perform the following two steps repeatedly until convergence:

$$\begin{aligned} &\text{(a)} \quad \partial g(\mathbf{z}^{(t)}) = l\mathbf{z}^{(t)} - (\mathbf{y} - \mathbf{A}\mathbf{z}^{(t)}) + 2\rho\overline{\mathbf{z}}_K^{(t)} \\ &\text{(b)} \quad \mathbf{z}^{(t+1)} = \arg\min_{\mathbf{z}\succeq\mathbf{0}} \big\{\frac{l}{2}\|\mathbf{z}\|_2^2 + \rho\|\mathbf{z}\|_2^2 - \mathbf{z}^T\partial g(\mathbf{z}^{(t)})\big\} \end{aligned} \tag{5.9}$$

where $\overline{\mathbf{z}}_K^{(t)}$ is the gradient of top-$(K, 2)$ norm while noting that $\overline{\mathbf{z}}_K^{(t)} = z_{(1)} + z_{(2)} + \cdots + z_{(K)}$, where $z_{(i)}$ denotes the element whose magnitude is the ith-largest among the elements of the vector $\mathbf{x}$, i.e., $z_{(1)} \geq z_{(2)} \geq \cdots \geq z_{(K)}$.

The step (b) in (5.9) has a closed-form optimal solution that can be rewritten as

$$\min_{\mathbf{z}\succeq\mathbf{0}} \quad \frac{l + 2\rho}{2}\big(\|\mathbf{z}\|_2^2 - \mathbf{z}^T\partial g(\mathbf{z}^{(t)})\big). \tag{5.10}$$

Next, problem (5.10) can be expressed as

$$\min_{\mathbf{z}\succeq\mathbf{0}} \quad \frac{l + 2\rho}{2}\|\mathbf{z} - \frac{1}{l + 2\rho}\partial g(\mathbf{z}^{(t)})\|_2^2. \tag{5.11}$$

The objective of (5.11) is to minimize a quadratic function with respect to the vector $\mathbf{z}$ over the set $\mathbf{z} \succeq \mathbf{0}$. We can obtain a closed-form optimal solution to (5.11), which is simply the Euclidean projection of $\frac{1}{l+2\rho}\partial g(\mathbf{z}^{(t)})$ onto the nonnegative orthant. The optimal solution to (5.11) can be expressed as follows

$$\mathbf{z}^* = \left(\frac{1}{l+2\rho}\partial g(\mathbf{z}^{(t)})\right)_+. \tag{5.12}$$

The DC programming steps in (5.9) reduce to

$$\begin{aligned} &\text{(a)} \quad \partial g(\mathbf{z}^{(t)}) = l\mathbf{z}^{(t)} - (\mathbf{y} - \mathbf{A}\mathbf{z}^{(t)}) + 2\rho\overline{\mathbf{z}}_K^{(t)} \\ &\text{(b)} \quad \mathbf{z}^{(t+1)} = \left(\frac{1}{l+2\rho}\partial g(\mathbf{z}^{(t)})\right)_+. \end{aligned} \tag{5.13}$$

Next, we substitute $\partial g(\mathbf{z}^{(t)})$ in step (a) into step (b) of (5.13) and obtain the tth update for $\mathbf{z}^{(t)}$ expressed as

$$\begin{aligned} \mathbf{z}^{(t+1)} &= \left(\frac{1}{l+2\rho}\left(l\mathbf{z}^{(t)} + 2\rho\mathbf{z}^{(t)} - 2\rho\mathbf{z}^{(t)} - \mathbf{A}^T(\mathbf{y} - \mathbf{A}\mathbf{z}^{(t)}) + 2\rho\overline{\mathbf{z}}_K^{(t)}\right)\right)_+ \\ &= \big(\mathbf{z}^{(t)} - \frac{1}{l+2\rho}\underbrace{\left(\mathbf{A}^T(\mathbf{y} - \mathbf{A}\mathbf{z}^{(t)}) + 2\rho\mathbf{z}^{(t)} - 2\rho\overline{\mathbf{z}}_K^{(t)}\right)}_{\nabla F(\mathbf{z}^{(t)})}\big)_+. \end{aligned} \tag{5.14}$$

Thus, we obtain (5.14) as an iterative solution for solving the optimization in (5.8), which is an equivalent expression of the trimmed ridge regression for sparse reconstruction in (5.5). Since (5.14) is derived from the DC programming method, the update in (5.14) shares the same global convergence property as DC programming.

5.1.3 Practical Algorithms for Trimmed Ridge Regression

When examining (5.14) it should be noted that it is a gradient projection descent update with a special step size of $1/(l+2\rho)$, where $l = \lambda_{\max}(\mathbf{A}^T\mathbf{A})$ is the maximal eigenvalue of $\mathbf{A}^T\mathbf{A}$ and ρ is the regularization parameter. The tth-step update of the gradient projection descent iteration can be expressed

as

$$\mathbf{z}^{(t+1)} = \left(\mathbf{z}^{(t)} - \frac{1}{l+2\rho}\nabla F(\mathbf{z}^{(t)})\right)_+ \tag{5.15}$$

where $(\cdot)_+$ is the projection operation that projects a vector onto the nonnegative orthant [27], and

$$\nabla F(\mathbf{z}^{(t)}) = \mathbf{A}^T(\mathbf{y} - \mathbf{A}\mathbf{z}^{(t)}) + 2\rho\mathbf{z}^{(t)} - 2\rho\bar{\mathbf{z}}_K^{(t)}. \tag{5.16}$$

It should be noted that (5.15) is a valid iterative solution for trimmed ridge regression. However, in practice, the fixed step size $1/(l+2\rho)$ is too small to converge in a reasonable amount of time. Next, we design two variable step size strategies and propose two accelerated algorithms for trimmed ridge regression.

Monotonic BB Step-size Gradient Projection Descent (MoGPD): Using a monotonic BB step size strategy, the tth-step update can be expressed as

$$\begin{aligned} \mathbf{w}^{(t+1)} &= \left(\mathbf{z}^{(t)} - \alpha^{(t)}\nabla F(\mathbf{z}^{(t)})\right)_+, \\ \mathbf{z}^{(t+1)} &= \mathbf{z}^{(t)} + \beta^{(t)}(\mathbf{w}^{(t+1)} - \mathbf{z}^{(t)}). \end{aligned} \tag{5.17}$$

where $\alpha^{(t)} > 0$ is the step size for the tth update and $\beta^{(t)} \in (0, 1]$ is the extrapolation parameter that prevents the step size from being too large to ensure the objective value descends monotonically.

Nesterov's accelerated gradient projection descent (NaGPD): Using the Nesterov's acceleration method, the tth-step update of the accelerated gradient projection iterations can be expressed as

$$\begin{aligned} \mathbf{w}^{(t+1)} &= \left(\mathbf{z}^{(t)} - \alpha^{(t)}\nabla F(\mathbf{z}^{(t)})\right)_+, \\ \mathbf{z}^{(t+1)} &= \mathbf{w}^{(t+1)} + \beta^{(t)}(\mathbf{w}^{(t+1)} - \mathbf{w}^{(t)}). \end{aligned} \tag{5.18}$$

In (5.17) and (5.18), $\nabla F(\mathbf{z}^{(t)})$ represents the gradient of the function $F(\cdot)$ in terms of $\mathbf{z}^{(t)}$ which is defined in (5.16).

[27] The nonnegative orthant is a generalization of the first quadrant in N dimensions for $N \geq 2$, i.e., $\mathbb{R}_+^N = \{\mathbf{x} = (x_1, x_2, ..., x_N) | x_1 \geq 0, x_2 \geq 0, ..., x_N \geq 0\}$.

We summarize the updates (5.17) and (5.18) as sparse reconstruction algorithms Algorithm 4 and Algorithm 5, which can be used to solve the sparse channel reconstruction with either random measurement matrices or data-driven measurement matrices. In this chapter, we focus on developing data-driven methods; therefore, we further unfold Algorithm 4 and Algorithm 5 into deep networks to build deep compressive sensing autoencoders.

Algorithm 4: Trimmed ridge regression via monotone gradient projection descent (Trr-MoGPD)

Input: measurements $\mathbf{y}$, matrix $\mathbf{A}$ and a small number ϵ
Output: reconstructed $\hat{\mathbf{x}}$
Initialization: $\mathbf{u}^{(0)}$, $\mathbf{v}^{(0)}$, $\mathbf{z}^{(0)} \leftarrow [(\mathbf{u}^{(0)})^T, (\mathbf{v}^{(0)})^T]^T$

1: **for** $t = 1, 2, \ldots$ **do**
2: $\quad \nabla F(\mathbf{z}^{(t)}) \leftarrow \mathbf{A}^T(\mathbf{y} - \mathbf{A}\mathbf{z}^{(t)}) + \rho(\mathbf{z}^{(t)} - \overline{\mathbf{z}}_K^{(t)})$
3: $\quad \mathbf{w}^{(t+1)} \leftarrow \left(\mathbf{z}^{(t)} - \alpha^{(t)} \nabla F(\mathbf{z}^{(t)})\right)_+$
4: $\quad \mathbf{z}^{(t+1)} \leftarrow \mathbf{z}^{(t)} + \beta^{(t)}(\mathbf{w}^{(t+1)} - \mathbf{z}^{(t)})$
5: $\quad$ Check terminate condition $\|\mathbf{z}^{(t+1)} - \mathbf{z}^{(t)}\|_2 \leq \epsilon$, return to Step 1 if not satisfied; otherwise, terminate with $\mathbf{z}^{(t+1)} = [(\mathbf{u}^{(t+1)})^T, (\mathbf{v}^{(t+1)})^T]^T$, and return $\hat{\mathbf{x}} = \mathbf{u}^{(t+1)} - \mathbf{v}^{(t+1)}$.
6: **end for**

5.2 Trimmed Ridge Regression Autoencoder for Joint Data-Driven Pilot Design and Channel Reconstruction

In the last section, two sparse reconstruction algorithms were derived starting from the trimmed ridge regression for sparse reconstruction and applying gradient projection descent under the DC programming framework to solve it. In this section, we build model-based autoencoders to augment the proposed algorithms. More specifically, we use a deep unfolding technique to transform the proposed algorithms into stacked layers of deep networks by treating each update step of the iterations as a layer of a deep network.

Algorithm 5: Trimmed ridge regression via Nesterov's accelerated gradient projection descent (Trr-NaGPD)

Input: measurements $\mathbf{y}$, matrix $\mathbf{A}$ and a small number ϵ
Output: reconstructed $\hat{\mathbf{x}}$
Initialization: $\mathbf{u}^{(0)}$, $\mathbf{v}^{(0)}$, $\mathbf{z}^{(0)} \leftarrow [(\mathbf{u}^{(0)})^T, (\mathbf{v}^{(0)})^T]^T$

1: **for** $t = 1, 2, \ldots$ **do**
2: $\nabla F(\mathbf{z}^{(t)}) \leftarrow \mathbf{A}^T(\mathbf{y} - \mathbf{A}\mathbf{z}^{(t)}) + \rho(\mathbf{z}^{(t)} - \overline{\mathbf{z}}_K^{(t)})$
3: $\mathbf{w}^{(t+1)} = \left(\mathbf{z}^{(t)} - \alpha^{(t)}\nabla F(\mathbf{z}^{(t)})\right)_+$
4: $\mathbf{z}^{(t+1)} = \mathbf{w}^{(t+1)} + \beta^{(t)}(\mathbf{w}^{(t+1)} - \mathbf{w}^{(t)})$
5: Check terminate condition $\|\mathbf{z}^{(t+1)} - \mathbf{z}^{(t)}\|_2 \leq \epsilon$, return to Step 1 if not satisfied; otherwise, terminate with $\mathbf{z}^{(t+1)} = [(\mathbf{u}^{(t+1)})^T, (\mathbf{v}^{(t+1)})^T]^T$, and return $\hat{\mathbf{x}} = \mathbf{u}^{(t+1)} - \mathbf{v}^{(t+1)}$.
6: **end for**

The unfolded deep networks are then used as the decoders of the built autoencoders named the trimmed ridge regression autoencoders (TrrAE). In the following subsections, we first present the framework of trimmed ridge regression autoencoders and then present its two types of decoder structures by unfolding Trr-MoGPD or Trr-NaGPD algorithms. Then, we discuss the pipeline of using the proposed deep compressive sensing autoencoders for joint pilot design and beamspace channel estimation.

5.2.1 The Trimmed Ridge Regression Autoencoder

The framework of TrrAE is shown in Fig. 5.1, consisting of a single-layer encoder from $\mathbf{x}$ to $\mathbf{y}$ and a multi-layer decoder from $\mathbf{y}$ to $\hat{\mathbf{x}}$. The encoder performs a linear dimension reduction $\mathbf{y} = \mathbf{\Phi}\mathbf{x}$. The input sparse vector $\mathbf{x}$ is projected onto the subspace spanned by the columns of the measurement matrix $\mathbf{\Phi}$ to get the compressed measurement vector $\mathbf{y}$. The decoder plays the role of the sparse reconstruction from $\mathbf{y}$ to $\hat{\mathbf{x}}$ in a feed-forward manner. The decoder is constructed to mimic the trimmed ridge regression algorithms Trr-MoGPD or Trr-NaGPD to obtain the reconstructed $\hat{\mathbf{x}}$. We can interpret the decoder by three functional parts, which are the initializing layer, the unfolding layers and the output layer.

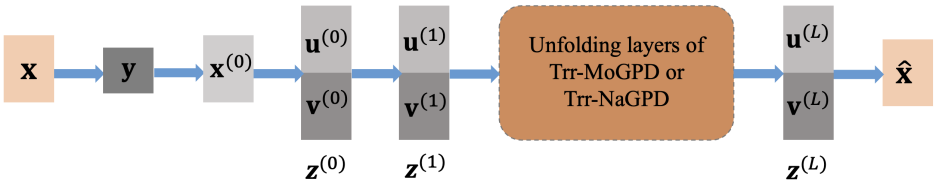


Figure 5.1: The framework of the TrrAE

Initializing decoder layer ($\mathbf{y}$ to $\mathbf{z}^{(0)}$): The first two layers can be understood as the initialization of a sparse reconstruction algorithm, that originates from the input of the compressed measurement matrix $\mathbf{r}$ and is used to get the initialized positive vector $\mathbf{z}^{(0)}$. For the first layer, instead of simply initializing $\mathbf{x}^{(0)}$ to be a zero vector as is done in the conventional sparse recovery strategy, we set

$$\mathbf{x}^{(0)} = \mathbf{\Phi}^T \mathbf{y}. \tag{5.19}$$

The second layer assists with preparing an equivalent positive initialization vector $\mathbf{z}^{(0)}$ by separating and concatenating the positive and negative part of $\mathbf{x}^{(0)}$, that is $\mathbf{z}^{(0)} = [(\mathbf{u}^{(0)})^T, (\mathbf{v}^{(0)})^T]^T$, where $\mathbf{u}^{(0)} = \text{ReLU}(\mathbf{x}^{(0)})$ and $\mathbf{v}^{(0)} = \text{ReLU}(-\mathbf{x}^{(0)})$.

Unfolding layers of trimmed ridge regression ($\mathbf{z}^{(1)}$ to $\mathbf{z}^{(L)}$): The second part of the decoder performs a forward update from $\mathbf{z}^{(1)}$ to $\mathbf{z}^{(L)}$, which consists of L structural-identical blocks. Each block is a one-step iteration of the sparse reconstruction algorithm. More specifically, we cast the update steps of the gradient projection descent or its acceleration algorithm into stacked layers of the decoder, and set the measurement matrix $\mathbf{\Phi}$ as the tied weight matrix for each layer. We empirically set the extrapolation parameter to be $\beta^{(t)} = \frac{t}{t+3}$. We construct two types of structures for the unfolding reconstruction layers by unfolding the proposed Trr-MoGPD and Trr-NaGPD updates. The single-layer structures of the unfolding Trr-MoGPD and Trr-NaGPD updates are shown in Fig. 5.2 and Fig. 5.3. It is worth noting that the ReLU activation function in the unfolding layers is the exact replacement of the projection operations $(\cdot)_+$ in the Trr-MoGPD and Trr-NaGPD updates.

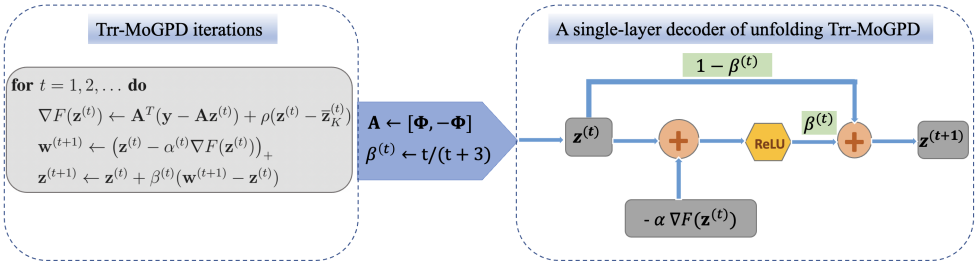


Figure 5.2: A single-layer structure of the unfolding Trr-MoGPD

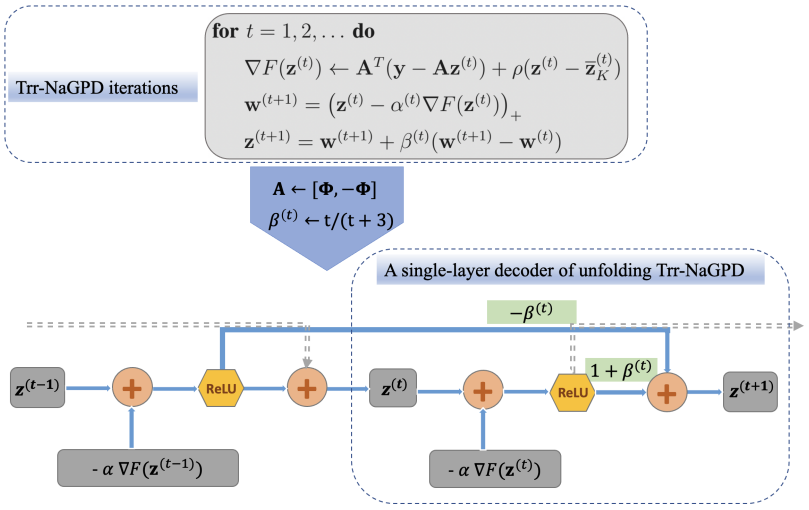


Figure 5.3: A single-layer structure of the unfolding Trr-NaGPD

Output layer: The last layer is the output layer, in which we obtain the recovered sparse vector $\hat{\mathbf{x}}$ from $\mathbf{z}^{(L)}$. After setting $\mathbf{z}^{(L)} = [z_1^{(L)}, ..., z_{2N}^{(L)}]$, we have

$$\hat{\mathbf{x}} = \mathbf{u}^{(L)} - \mathbf{v}^{(L)} \tag{5.20}$$

where $\mathbf{u}^{(L)} = [z_1^{(L)}, ..., z_N^{(L)}]$ and $\mathbf{v}^{(L)} = [z_{N+1}^{(L)}, ..., z_{2N}^{(L)}]$.

Loss function: The encoder and decoder are jointly trained together to minimize the mean squared error of the reconstruction. The loss function is defined as

$$\mathcal{L}(\mathbf{x}, \hat{\mathbf{x}}) = \frac{1}{n}\sum_{i=1}^{n}\|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2 \tag{5.21}$$

where n denotes the number of training data samples.

Trainable parameters: The trainable parameters are $\{\mathbf{\Phi}, \rho, \alpha\}$. In particular, we cast the measurement matrix $\mathbf{\Phi}$ as the tied weights for both the encoder and multiple layers of decoder so that the measurement matrix can be optimized while training. Given the training dataset, the weight matrix $\mathbf{\Phi}$ is optimized by backpropagation to minimize the loss function in (5.21). Then, the optimized measurement matrix $\mathbf{\Phi}_{opt}$ can be extracted after training the autoencoder. Once we obtain the optimized measurement matrix $\mathbf{\Phi}_{opt}$, we can then obtain the optimized pilots using $\mathbf{P}_{opt} = \mathbf{U}_t \mathbf{\Phi}_{opt}^T$.

The processing pipeline when using the TrrAE for data-driven pilot design and channel estimation is shown in Fig. 5.4. The proposed TrrAE is first trained offline to acquire the optimized data-driven measurement matrix for pilot design, then the trained decoder is implemented online to estimate the beamspace channels in real time.

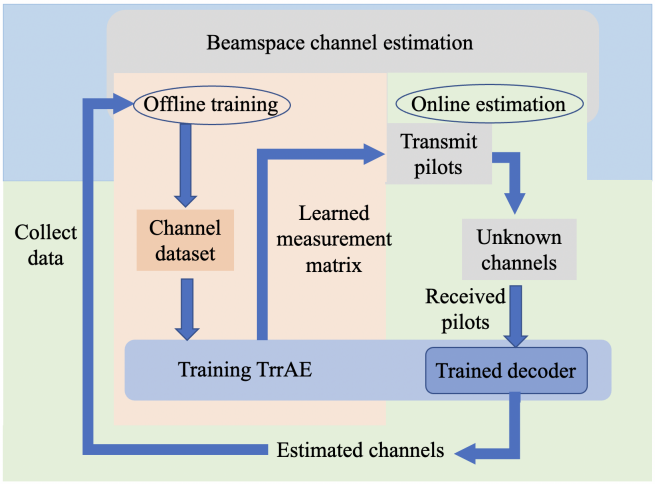


Figure 5.4: Processing pipeline that shows the offline training process and the online estimation process for designing data-driven pilots and estimating the beamspace channels

5.2.2 Theoretical Insight for Joint Learning of Measurement Matrix and Sparse Reconstruction

The model-based TrrAE autoencoders have been proposed to learn the measurement matrix $\mathbf{\Phi}$ and the best reconstruction $\mathcal{F}(\cdot) := \mathbb{R}^M \rightarrow \mathbb{R}^N$ jointly. In this section, we provide theoretical insights regarding the design of an autoencoder for learning optimal measurement matrix and reconstruction [28]. Given a measurement matrix $\mathbf{\Phi}$, we assume a dataset $\mathcal{D}_{train} = \{(\mathbf{x}_1, \mathbf{y}_1), ..., (\mathbf{x}_n, \mathbf{y}_n)\}$, where $\mathbf{y}_i = \mathbf{\Phi}\mathbf{x}_i$ for $1 \leq i \leq n$, and n denotes the number of samples in the training dataset. According to the infomax principle [97, 98], the measurement matrix should be optimized to maximize the mutual information of $\mathbf{x}$ and $\mathbf{y}$, that is

$$\begin{aligned}
\hat{\mathbf{\Phi}} &= \arg\max_{\mathbf{\Phi}} I(\mathbf{x}; \mathbf{y}) \\
&= \arg\max_{\mathbf{\Phi}} H(\mathbf{x}) - H(\mathbf{x}|\mathbf{y}) \\
&= \arg\max_{\mathbf{\Phi}} \mathbb{E}[\log(p(\mathbf{x}|\mathbf{y}))] \\
&\overset{(a)}{=} \arg\max_{\mathbf{\Phi}} \lim_{n\to\infty} \frac{1}{n} \sum_{i=1}^{n} \log(p(\mathbf{x}_i|\mathbf{y}_i)) \\
&= \arg\max_{\mathbf{\Phi}} \lim_{n\to\infty} e^{\frac{1}{n}\sum_{i=1}^{n} \ln(p(\mathbf{x}_i|\mathbf{y}_i))} \\
&= \arg\max_{\mathbf{\Phi}} \lim_{n\to\infty} \Big(\prod_{i=1}^{n} p(\mathbf{x}_i|\mathbf{y}_i)\Big)^{(1/n)}
\end{aligned} \tag{5.22}$$

where the equality (a) holds due to the law of large numbers; $p(\cdot)$ represents the probability mass function (PMF). Eq. (5.22) implies that the optimal measurement matrix $\mathbf{\Phi}$ can be obtained by maximizing the conditional probability of ground-truth samples $\mathbf{x}$ given the compressed measurements $\mathbf{y}$ in an asymptotic setting. For the real-world dataset having limited number of samples we have

$$\hat{\mathbf{\Phi}} = \arg\max_{\mathbf{\Phi}} \prod_{i=1}^{n} p(\mathbf{x}_i|\mathbf{y}_i) \tag{5.23}$$

[28] It is worth mentioning that this section is not a rigorous mathematical proof. Instead, we show some understanding of the proposed method from the theoretical perspective.

Eq. (5.23) implies that the optimal pilots $\mathbf{\Phi}$ can be obtained by maximizing the probability of the samples $\{\mathbf{x}_i\}$ in the training dataset given the compressed measurements $\{\mathbf{y}_i\}$ for $1 \leq i \leq n$.

Now we consider jointly optimizing the pilots sequences $\mathbf{\Phi}$ and a reconstruction function $\mathcal{F}(\cdot) := \mathbb{R}^M \to \mathbb{R}^N$. We further assume that the reconstruction function $\mathcal{F}_{\mathbf{\Theta}}(\cdot)$, which is parameterized by $\mathbf{\Theta}$, reconstructs the samples $\hat{\mathbf{x}}_i = \mathcal{F}_{\mathbf{\Theta}}(\mathbf{y}_i)$ for $1 \leq i \leq n$. Then, similar to (5.23), we can define the optimal measurement matrix and the best reconstruction function as the ones that maximize the conditional probability of original training samples given the reconstructions, as shown below

$$(\hat{\mathbf{\Phi}}, \hat{\mathbf{\Theta}}) = \arg\max_{\mathbf{\Phi},\mathbf{\Theta}} \prod_{i=1}^{n} p(\mathbf{x}_i|\hat{\mathbf{x}}_i). \tag{5.24}$$

In the asymptotic setting, we can rewrite (5.24) as

$$\begin{aligned}(\hat{\mathbf{\Phi}}, \hat{\mathbf{\Theta}}) &= \arg\max_{\mathbf{\Phi},\mathbf{\Theta}} \lim_{N\to\infty} \prod_{i=1}^{n} p(\mathbf{x}_i|\hat{\mathbf{x}}_i) \\ &= \arg\max_{\mathbf{\Phi},\mathbf{\Theta}} \lim_{N\to\infty} \prod_{i=1}^{n} p(\mathbf{x}_i|\mathbf{y}_i = \mathbf{\Phi}\mathbf{x}_i; \mathbf{\Theta}) \\ &= \arg\max_{\mathbf{\Phi},\mathbf{\Theta}} \mathbb{E}_{\mathbf{x}}[\log p(\mathbf{x}|\mathbf{y} = \mathbf{\Phi}\mathbf{x}; \mathbf{\Theta})].\end{aligned} \tag{5.25}$$

Since the true PMF of $p(\mathbf{x}|\hat{\mathbf{x}})$ is unknown, we assume a parametric PMF that has the probability of $\tilde{p}(\mathbf{x}|\hat{\mathbf{x}})$. The Kullback–Leibler divergence is non-negative, i.e.,

$$D_{KL}(p\|\tilde{p}) = \sum_{\mathbf{x}} p(\mathbf{x}|\hat{\mathbf{x}}) \log \frac{p(\mathbf{x}|\hat{\mathbf{x}})}{\tilde{p}(\mathbf{x}|\hat{\mathbf{x}})} \geq 0 \tag{5.26}$$

and therefore

$$\mathbb{E}_{\mathbf{x}}[\log \tilde{p}(\mathbf{x}|\hat{\mathbf{x}})] \leq \mathbb{E}_{\mathbf{x}}[\log p(\mathbf{x}|\hat{\mathbf{x}})]. \tag{5.27}$$

Eq. (5.27) indicates that we can create a lower bound of the true mutual information between $\mathbf{x}$ and $\hat{\mathbf{x}}$ using the parametric distribution $\tilde{p}(\mathbf{x}|\hat{\mathbf{x}})$. Since $\hat{\mathbf{x}}$ is associated with the measurements $\mathbf{y} = \mathbf{\Phi}\mathbf{x}$ and $\mathbf{\Theta}$, we can express (5.27)

as

$$\mathbb{E}_{\mathbf{x}}[\log \tilde{p}(\mathbf{x}|\mathbf{y} = \mathbf{\Phi}\mathbf{x}; \mathbf{\Theta})] \leq \mathbb{E}_{\mathbf{x}}[\log p(\mathbf{x}|\mathbf{y} = \mathbf{\Phi}\mathbf{x}; \mathbf{\Theta})]. \tag{5.28}$$

Thus, we can maximize the lower bound of the true mutual information to optimize $\mathbf{\Phi}$ and $\mathbf{\Theta}$ approximately. Accordingly, we maximize the opposite of conditional entropy, i.e., $\mathbb{E}_{\mathbf{x}}[\log p(\mathbf{x}|\mathbf{y} = \mathbf{\Phi}\mathbf{x}; \mathbf{\Theta}))]$ to optimize $\mathbf{\Phi}$ and $\mathbf{\Theta}$ jointly

$$\begin{aligned} \mathbf{\Phi}', \mathbf{\Theta}' &= \arg\max_{\mathbf{\Phi},\mathbf{\Theta}} \mathbb{E}_{\mathbf{x}}[\log \tilde{p}(\mathbf{x}|\mathbf{y} = \mathbf{\Phi}\mathbf{x}; \mathbf{\Theta})] \\ &= \arg\max_{\mathbf{\Phi},\mathbf{\Theta}} \lim_{n\to\infty} \prod_{i=1}^{n} \tilde{p}(\mathbf{x}_i|\mathbf{y}_i = \mathbf{\Phi}\mathbf{x}_i; \mathbf{\Theta}) \\ &= \arg\max_{\mathbf{\Phi},\mathbf{\Theta}} \lim_{n\to\infty} \prod_{i=1}^{n} \tilde{p}(\mathbf{x}_i|\hat{\mathbf{x}}). \end{aligned} \tag{5.29}$$

Eq. (5.25), (5.28) and (5.29) provide insight as to why it is reasonable to learn pilot matrix and reconstructions jointly in an end-to-end deep network. Assuming an estimation error $\boldsymbol{\epsilon} = \mathbf{x} - \hat{\mathbf{x}}$ whose entries follow a Gaussian distribution, i.e., $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \lambda\mathbf{I})$, allows us to obtain $\tilde{p}(\mathbf{x}_i|\hat{\mathbf{x}}) \sim \mathcal{N}(\hat{\mathbf{x}}, \lambda\mathbf{I})$. Then, the maximization in (5.29) can be realized by minimizing the MSE, i.e., $\frac{1}{n}\sum_{i=1}^{n}\|\mathbf{x}_i - \hat{\mathbf{x}}_i\|_2^2$, which is set as the loss function of the entire deep network.

5.3 Numerical Results

In this section, we evaluate the performance of the proposed TrrAE-MoGPD and TrrAE-NaGPD autoencoders, where the TrrAE-MoGPD is the TrrAE with the unfolding Trr-MoGPD decoder, and TrrAE-NaGPD is the TrrAE with the unfolding Trr-NaGPD decoder. We choose the model-based deep learning method LISTA, and the generic deep learning network CNN as the baselines for performance comparisons. The LISTA method is implemented by changing the decoder in our proposed TrrAE framework to the LISTA network shown in Fig. 2.6 [1]. The generic CNN method is set up according to the implementation in [68], which proposed a DNN network consisting of a linear encoder and stacked CNN decoder. Also, we compare

the channel reconstruction performance with the previous proposed schemes, including the scheme proposed in Chapter 3, i.e., the DlDC-GPSR algorithm with the random Gaussian measurement matrix, and the scheme proposed in Chapter 4, i.e., DlDC-GPSR algorithm with a data-driven measurement matrix learned by the autoencoder *BP-gAE*. In the following subsections, we first present the experiment set up, and then provide the performance evaluation results.

5.3.1 Experiment Setup

We consider a massive MIMO system having $N_t = 256$ antennas at the BS and a single antenna at the UEs [29], i.e., $N_r = 1$. We randomly generate $10,000$ channel vectors according to the narrowband Saleh-Valenzuela channel model in (2.10), and set the number of paths as $N_p = 3$. For each path, the complex channel gain α_l follows a complex Gaussian distribution; the AoA and AoD, i.e., $\theta_{r,l}$ and $\theta_{t,l}$, are uniformly distributed over $[-\pi/2, \pi/2]$. We transform the generated channel vectors into beamspace channel vectors using a standard DFT dictionary as shown in (2.11). To consider power leakage, we set the number of nonzero channel coefficients as $N_s = 16$ by neglecting the small-value beamspace channel coefficients [30]. We stack the real part and imaginary part of each beamspace channel vector column-wise, and obtain a dataset $\{\mathbf{X}\}$ consisting of $20,000$ real-form sparse vectors, which are $\{\mathbf{X}\} = \{[\Re(\mathbf{h}_1), \Im(\mathbf{h}_1), \Re(\mathbf{h}_2), \Im(\mathbf{h}_2, ...)]\}$. We split the samples of the full dataset using a ratio of 0.8/0.1/0.1 into training, validation and testing datasets, where the training and validation datasets are used to train autoencoder models, and the testing dataset is used for performance evaluations.

[29] For $N_r > 1$, to reconstruct the beamspace channel matrix $\mathbf{H} \in \mathbb{C}^{N_t \times N_r}$, we can simply reconstruct its N_r columns independently.

[30] Although in practice the number of nonzero channel coefficients can be unknown and uncertain, in the simulation we fix the number of nonzero channel coefficients for fair performance comparisons. We set $N_s = 16$ because our observations revealed that the 16 largest-magnitude channel coefficients can cover most of the energy of a channel vector, i.e., $\|\mathbf{h}_i\|_2^2$ for $1 \leq i \leq N_r$.

5.3.2 Performance Comparisons With Other Deep Learning Methods

In this section, we evaluate the channel reconstruction errors when the pilot lengths and noise levels are varied. Fig. 5.5 shows that in noiseless scenarios the reconstruction accuracy of the proposed TrrAE-NaGPD and TrrAE-MoGPD is higher than the LISTA and CNN methods, and particularly the TrrAE-NaGPD has the best reconstruction accuracy. In noisy scenarios, as shown in Fig. 5.6 and Fig. 5.7, when the noise level is low to median, the proposed autoencoders are more accurate than the LISTA and CNN methods. When the noise level is high, Fig. 5.8 shows that all the schemes have similar reconstruction errors, but the proposed TrrAE-NaGPD and TrrAE-MoGPD still achieve lower reconstruction errors when the number of pilots are sufficiently large. Comparing the proposed TrrAE-NaGPD and TrrAE-MoGPD, we observe that the TrrAE-NaGPD outperforms the TrrAE-MoGPD in noiseless and low-noise scenarios shown in Fig. 5.5 and Fig. 5.6 respectively; at higher nose levels, their reconstruction performance is similar as seen in Fig. 5.7 and Fig. 5.8.

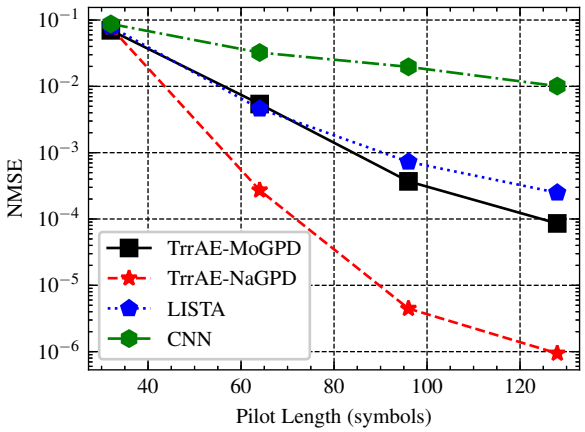


Figure 5.5: The NMSE versus pilot length for noiseless scenarios

Next, we investigate how the NMSE changes over different noise levels. We plot the NMSE against different noise standard deviations when the pilot length is $M = 64$ in Fig. 5.9. We then plot the NMSE against different noise

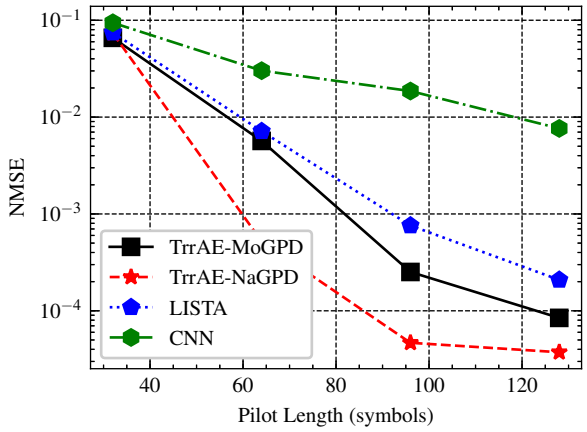


Figure 5.6: The NMSE versus pilot length when the standard deviation of noise is $\sigma = 0.001$

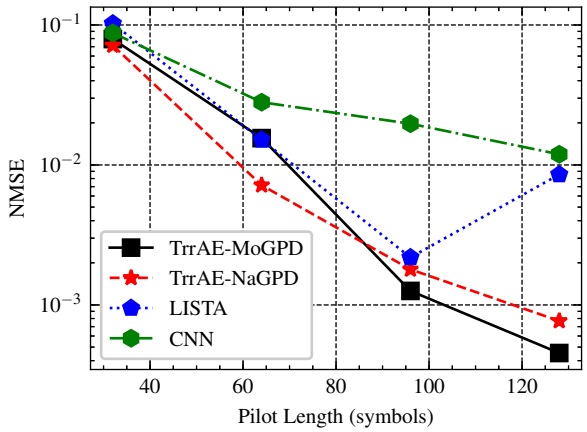


Figure 5.7: The NMSE versus pilot length when the standard deviation of noise is $\sigma = 0.01$

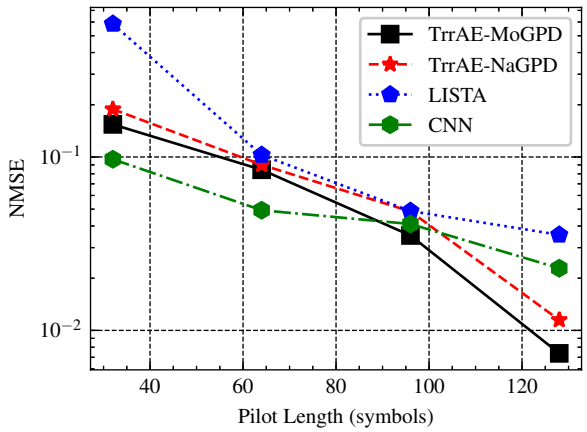


Figure 5.8: The NMSE versus pilot length when the standard deviation of noise is $\sigma = 0.1$

standard deviations when the pilot length $M = 128$ in Fig. 5.10. We can observe that the behavior of the CNN is relatively consistent at different noise levels, while the TrrAE-NaGPD, TrrAE-MoGPD, and LISTA show reduced reconstruction NMSE when decreasing the noise power. When compared with the other deep learning methods, the proposed TrrAE-NaGPD and TrrAE-MoGPD autoencoders are shown to be more accurate. The proposed TrrAE-NaGPD has the best performance and shows significant advantages especially when exposed to low noise levels, such as when $\sigma = 0.001$ or $\sigma = 0$.

Next, we compare the achievable rates of different methods against different pilot lengths when taking reconstruction errors into account. The achievable rate is computed as the lower bound of efficient spectrum efficiency, which is given by [67]

$$C = (1 - \frac{M}{N_c})\mathbb{E}\left[\log_2(1 + \frac{1}{\sigma^2 + \epsilon}\frac{\|\hat{\mathbf{h}}\|_2^2}{N})\right] \tag{5.30}$$

where ϵ is the NMSE of the estimated channel $\hat{\mathbf{h}}$; N_c is the length of a channel coherence block which we set to be $N_c = 300$ symbols. The achievable rates versus pilot lengths in noisy scenarios are shown in Fig. 5.11. We can observe

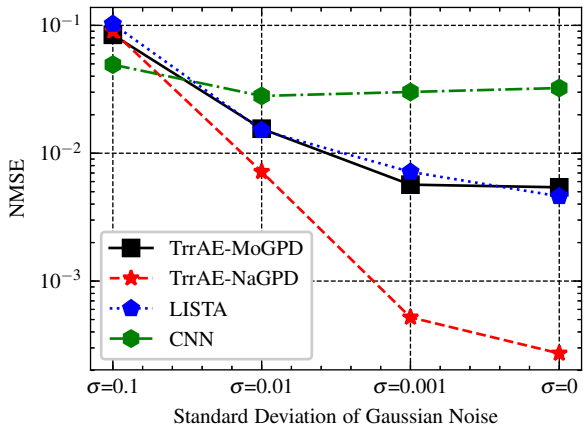


Figure 5.9: The NMSE at different noise strengths when the pilot length is $M = 64$ (symbols)

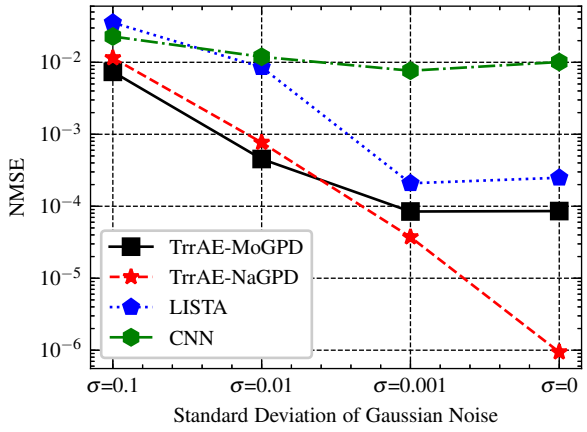


Figure 5.10: The NMSE at different noise strengths when the pilot length is $M = 128$ (symbols)

that the achievable rate of the CNN method has relatively consistent performance at a lower achievable rate even when the pilot lengths increase. The achievable rates of the other methods increase until a maximum is reached and then decrease afterwards. The proposed TrrAE-NaGPD autoencoder achieves the highest achievable rate around 8.9 bps/Hz at the pilot length $M = 64$, while the proposed TrrAE-MoGPD autoencoder achieves its high-

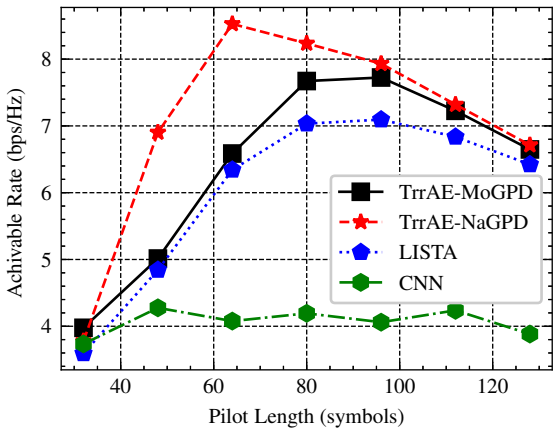


Figure 5.11: The efficient achievable rate versus pilot length when the standard deviation of noise is $\sigma = 0.001$

est achievable rate rate around 7.8 bps/Hz at the pilot length $M = 96$, which are both higher than the best achievable rate of LISTA. We can conclude that, compared with other deep learning methods, such as the generic CNN and the unfolding network LISTA, the proposed autoencoders use less pilots and are more accurate when used to estimate massive MIMO sparse channels.

5.3.3 Performance Comparisons With Previous Proposed Sparse Channel Reconstruction Schemes

In this thesis, we have explored different sparse channel reconstruction schemes: in Chapter 3, we explored novel sparse reconstruction algorithms, while adopting the common random measurement matrices; in Chapter 4, we explored the data-driven measurement matrices that can be used to further improve the performance of the proposed sparse reconstruction algorithms; in this chapter, we explored the end-to-end data-driven method for joint measurement matrix design and channel reconstruction. Now, we compare the sparse channel reconstruction performance of the different schemes. We choose the proposed schemes having the best reconstruction accuracy in each chapter, which includes the DlDC-GPSR algorithm with a random

Gaussian matrix in Chapter 3, the DlDC-GPSR algorithm with the data-driven measurement matrix learned by the *BP-gAE* in Chapter 4, and the TrrAE-NaGPD in this chapter, and evaluate the reconstruction NMSE versus different noise levels for pilot lengths $M = 48$, $M = 64$ and $M = 80$ in Fig. 5.12, Fig. 5.13 and Fig. 5.14, respectively.

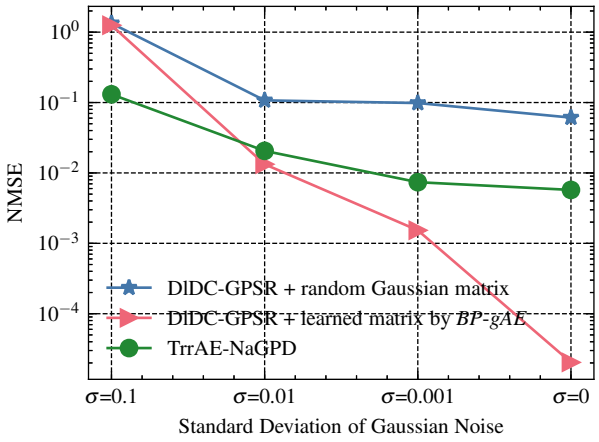


Figure 5.12: Reconstruction errors of our proposed schemes when the pilot length is $M = 48$

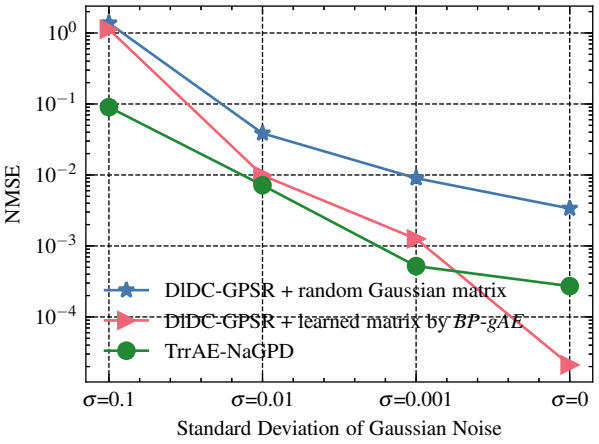


Figure 5.13: Reconstruction errors of our proposed schemes when the pilot length is $M = 64$

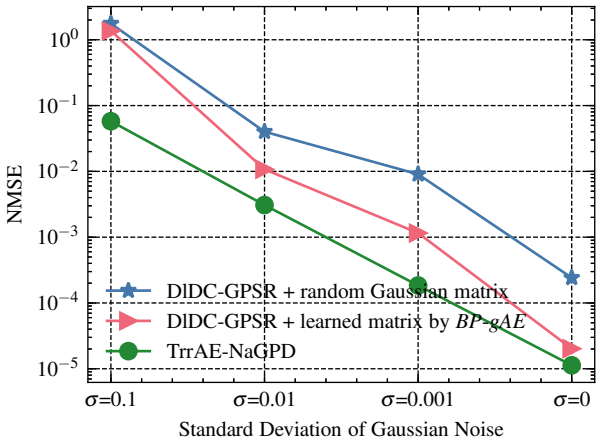


Figure 5.14: Reconstruction errors of our proposed schemes when the pilot length is $M = 80$

Figure 5.12 shows that for the pilot length $M = 48$, the proposed DlDC-GPSR algorithm with the learned measurement matrix achieves higher reconstruction accuracy when the noise is low or absent. Fig. 5.13 and Fig. 5.14 show that the proposed TrrAE-NaGPD becomes more accurate when increasing the pilot length. In Fig. 5.13, the DlDC-GPSR algorithm with the learned measurement matrix still has lower reconstruction error in noiseless scenarios, whereas Fig. 5.14 shows the joint learning measurement matrix and reconstruction scheme performed by TrrAE-NaGPD remains more accurate at all noise levels. Finally, it is important to mention the computational speed advantage of using the trained TrrAE for online channel reconstruction. In Table 3.2 of Chapter 3, it was shown that the computation times using various iterative algorithms are from tens of milliseconds to a few seconds for each channel sample reconstruction. However, the trained TrrAE only takes 1.2-1.5 seconds when reconstructing $1,000$ channel samples [31], which means the computation speed using the trained TrrAE is between 10 and $1,000$ times faster than using the iterative algorithms.

[31] The channel reconstruction tests using the TrrAE were implemented using the same computer hardware that was previously used for the experiments in Chapter 3. The desktop computer is equipped with a 3.2 GHz Intel Core i7-8700 CPU with 8GB of physical memory.

5.4 Summary

In this chapter, we explored model-based deep learning methods for joint data-driven pilot design and sparse channel estimation. We first proposed the trimmed ridge regression to formulate the sparse reconstruction problem, and then we derived an iterative solution for trimmed ridge regression using DC programming and gradient projection descent. Based on the derived iterative solution, we proposed two novel algorithms for sparse reconstructions and then unfolded the iterative algorithms into deep networks that can be used as sparse reconstruction decoders. We proposed the model-based autoencoder framework that consists of a fully connected linear encoder and a unfolded sparse reconstruction decoder. By unfolding the proposed Trr-NaGPD and Trr-MoGPD sparse reconstruction algorithms as the decoder, we construct two autoencoders named as the TrrAE-NaGPD and TrrAE-MoGPD. Numerical results show that, compared with other deep networks such as the generic CNN and the unfolding sparse reconstruction network LISTA, the proposed autoencoders achieve higher accuracy while using less pilots at reconstructing sparse channels. Furthermore, it should be noted that practical implementations require massive MIMO channels to be estimated accurately and in real time, and therefore fast computations are important. The sparse channel reconstructions using trained autoencoders are well suited for practical implementations since they can maintain satisfactory accuracy and are computationally faster than using the iterative algorithms.

Chapter 6

Conclusions

In this chapter, we summarize the contributions of this thesis. We also present several potential future research topics related to our work.

6.1 Summary of Contributions

In this thesis, we investigated sparse channel estimation for massive MIMO systems. Starting from the conventional compressive sensing framework, we first improved sparse reconstruction algorithms, and then we explored the use of a data-driven method to improve the measurement matrix design. Furthermore, we extended the data-driven method by using it to perform joint measurement matrix design and sparse reconstructions. Here we summarize the main contributions in each chapter of this thesis.

Contribution 1: We reformulated the sparse reconstruction optimization problems and proposed novel algorithms that significantly improved the performance of sparse reconstructions compared to conventional convex relaxation methods. By introducing the top-$(K, 1)$ norm, the sparse reconstruction problem is reformulated into an LS minimization penalized by a trimmed ℓ_1-norm regularizer, which removes the regularization on the K largest magnitude elements compared to the common ℓ_1-norm regularizer. The resulting optimization problem is nonconvex, but it is expected to produce more accurate solutions. Then, the DC programming framework and gradient projection descent are applied to solve the resulting nonconvex optimizations. Several DC-GPSR algorithms, including the DlDC-GPSR, SlDC-GPSR and SlDC-GPSR-BB, have been proposed and show significant advantages over existing reconstruction algorithms. These advantages result in more accurate reconstructions and faster runtimes.

Contribution 2: We proposed a model-based deep learning method to acquire data-driven measurement matrices, which were demonstrated to improve upon the reconstruction performance of conventional random measurement matrices. The data-driven measurement matrices also further improved the channel reconstruction accuracy when using the proposed DC-GPSR algorithms. To acquire data-driven measurement matrices, we use a deep unfolding technique to construct several model-based autoencoders named the deep unfolding *BP-AE*. The subgradient descent algorithm of basis pursuit for sparse reconstruction is unfolded into a multi-layer deep network, which functions as the decoder and is jointly trained with a single-layer dimensionality-reduction encoder. We set the measurement matrix as the tied weights for each layer of the constructed autoencoders so that the measurement matrix can be optimized by backpropagation algorithms. Using the acquired data-driven measurement matrices instead of random matrices, sparse reconstruction algorithms can attain more accurate reconstructions using fewer pilots, thereby leading to a higher achievable rate in massive MIMO systems.

Contribution 3: After separately optimizing the sparse reconstruction algorithm and measurement matrix under the conventional compressive sensing framework, we explored a joint optimization method for both the pilot design and the sparse reconstruction; this method was named deep compressive sensing, which was built by the model-based deep learning method and is an alternative to the conventional compressive sensing framework. We first reformulate the sparse reconstruction problem into an LS regression with a trimmed ridge regularizer, which removes the penalties on the K largest-magnitude elements from the common ℓ_2-norm regularizer. An iterative solution was derived for the trimmed ridge regression by applying DC gradient projection descent, and then the solution was developed into two algorithms: the Trr-MoGPD and Trr-NaGPD. By unfolding the proposed trimmed ridge regression algorithms into learnable deep networks, novel model-based autoencoders TrrAE are proposed. The proposed autoencoders are trained offline to obtain the data-driven pilots, and then the trained decoders can be used online for channel estimations in real time.

When compared with other deep learning methods, the proposed joint pilot learning and channel estimation scheme using TrrAE can achieve faster channel reconstructions with higher accuracy using fewer pilots. Moreover, when compared with our previous proposed schemes which employ iterative sparse reconstruction algorithms, the TrrAE can perform channel reconstruction significantly faster because a batch of unknown channels can be reconstructed non-iteratively and simultaneously in a parallel manner using the TrrAE trained decoder.

6.2 Future Works

Some potential future works related to this thesis are summarized in this section. In Chapter 3, we proposed several DC-GPSR algorithms. It is of interest to construct novel autoencoders by unfolding the DC-GPSR algorithms into deep networks for sparse reconstruction, and subsequently comparing the performance with the TrrAE proposed in Chapter 5. Another potential future work is to apply the Trr-NaGPD and Trr-MoGPD algorithms proposed in Chapter 5 with either random measurement matrices or the data-driven measurement matrices acquired in Chapter 4, and compare their channel reconstruction performance with the DC-GPSR algorithms proposed in Chapter 3. In Chapter 4, we proposed a data-driven measurement matrix design, and have shown significant improvements in sparse reconstructions over random matrices. It is of interest to analyse the obtained data-driven measurement matrix to better understand the aspects of its performance that make it superior over random matrices. In Chapter 5, we successfully unfolded the Trr-NaGPD and Trr-MoGPD algorithms into neural networks and built the TrrAE-NaGPD and TrrAE-MoGPD autoencoders. It is important to obtain theoretical insights regarding the empirical success of unfolding iterative algorithms into neural networks to solve sparse reconstruction problems. Moreover, it is of interest to investigate the convergence rate of the unfolded algorithms and study the tradeoffs between convergence rate and reconstruction accuracy.

Bibliography

[1] V. Monga, Y. Li, and Y. C. Eldar. Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing. *IEEE Signal Process. Mag.*, 38(2):18–44, 2021. → pages xii, 7, 34, 36, 101

[2] M. R. Raghavendra and K. Giridhar. Improving channel estimation in OFDM systems for sparse multipath channels. *IEEE Signal Process. Lett.*, 12(1):52–55, Jan. 2005. → pages 3

[3] Jose L. Paredes, Gonzalo R. Arce, and Zhongmin Wang. Ultra-wideband compressed sensing: Channel estimation. *IEEE J. Sel. Topics Signal Process.*, 1(3):383–395, Otc. 2007. → pages 3

[4] Georg Tauböck, Franz Hlawatsch, Daniel Eiwen, and Holger Rauhut. Compressive estimation of doubly selective channels in multicarrier systems: Leakage effects and sparsity-enhancing processing. *IEEE J. Sel. Topics Signal Process.*, 4(2):255–271, Apr. 2020. → pages 3

[5] Christian R. Berger, Shengli Zhou, James C. Preisig, and Peter Willett. Sparse channel estimation for multicarrier underwater acoustic communication: From subspace methods to compressed sensing. *IEEE Trans. Signal Process.*, 58(3):1708–1721, Mar. 2020. → pages 3

[6] S. F. Cotter and B. D. Rao. Sparse channel estimation via matching pursuit with application to equalization. *IEEE Trans. Commun.*, 50(3):374–377, Mar. 2002. → pages 3

[7] Z. Qin, J. Fan, Y. Liu, Y. Gao, and G. Y. Li. Sparse representation for wireless communications: A compressive sensing approach. *IEEE Signal Process. Mag.*, 35(3), May 2018. → pages 3, 5, 6

[8] Z. Gao, L. Dai, S. Han, C. I, Z. Wang, and L. Hanzo. Compressive sensing techniques for next-generation wireless communications. *IEEE Wireless Commun.*, 25(3):144–153, June 2018. → pages 3

[9] W. U. Bajwa, J. Haupt, A. M. Sayeed, and R. Nowak. Compressed channel sensing: A new approach to estimating sparse multipath channels. *Proc. IEEE*, 98(6):1058–1076, June 2010. → pages 3, 13, 14, 15, 16

[10] D. L. Donoho. Compressed sensing. *IEEE Trans. Inform. Theory*, 52(4):1289–1306, Apr. 2006. → pages 4, 5, 21

[11] E. J. Candès and M. B. Wakin. An introduction to compressive sampling. *IEEE Signal Process. Mag.*, 25(2):21–30, Mar. 2008. → pages 4

[12] R. G. Baraniuk. Compressive sensing. *IEEE Signal Process. Mag.*, 24(4):118–121, July 2007. → pages 4

[13] J. Brady, N. Behdad, and A. M. Sayeed. Beamspace MIMO for millimeter-wave communications: System architecture, modeling, analysis, and measurements. *IEEE Trans. Antennas Propag.*, 61(7):3814–3827, July 2013. → pages 4

[14] X. Li, J. Fang, H. Li, and P. Wang. Millimeter wave channel estimation via exploiting joint sparse and low-rank structures. *IEEE Trans. Wireless Commun.*, 17(2):1123–1133, Feb. 2018. → pages 4

[15] X. Lin, S. Wu, C. Jiang, L. Kuang, J. Yan, and L. Hanzo. Estimation of broadband multiuser millimeter wave massive MIMO-OFDM channels by exploiting their sparse structure. *IEEE Trans. Wireless Commun.*, 17(6), June 2018. → pages

[16] Z. Gao, C. Hu, L. Dai, and Z. Wang. Channel estimation for millimeter-wave massive MIMO with hybrid precoding over frequency-selective fading channels. *IEEE Commun. Lett.*, 20(6):1259–1262, June 2016. → pages

[17] J. Shen, J. Zhang, E. Alsusa, and K. B. Letaief. Compressed CSI acquisition in FDD massive MIMO: How much training is needed? *IEEE Trans. Wireless Commun.*, 15(6):4145–4156, June 2016. → pages 5, 6

[18] C. R. Berger, Z. Wang, J. Huang, and S. Zhou. Application of compressive sensing to sparse channel estimation. *IEEE Commun. Mag.*, 48(11):164–174, Nov. 2010. → pages 5, 6

[19] J. W. Choi, B. Shim, and S. Chang. Downlink pilot reduction for massive MIMO systems via compressed sensing. *IEEE Commun. Lett.*, 19(11):1889–1892, Nov. 2015. → pages

[20] Zhen Gao, Linglong Dai, Wei Dai, Byonghyo Shim, and Zhaocheng Wang. Structured compressive sensing-based spatio-temporal joint channel estimation for FDD massive MIMO. *IEEE Trans. Commun.*, 64(2):601–617, Feb. 2016. → pages 4

[21] C.-C. Tseng, J.-Y. Wu, and T.-S. Lee. Enhanced compressive downlink CSI recovery for FDD massive MIMO systems using weighted block l_1-minimization. *IEEE Trans. Commun.*, 64(3):1055–1067, Mar. 2016. → pages 4, 6

[22] Y. Lim and C. Chae. Compressed channel feedback for correlated massive MIMO systems. In *2014 IEEE International Conference on Communications Workshops (ICC)*, pages 360–364, June 2014. → pages 4

[23] Z. Liu, S. Sun, Q. Gao, and H. Li. CSI feedback based on spatial and frequency domains compression for 5G multi-user massive MIMO systems. In *2019 IEEE/CIC International Conference on Communications in China (ICCC)*, pages 834–839, Changchun, China, Aug. 2019. → pages

[24] H. Song, W. Seo, and D. Hong. Compressive feedback based on sparse approximation for multiuser MIMO systems. *IEEE Trans. Veh. Technol.*, 59(2):1017–1023, Feb. 2010. → pages

[25] M. E. Eltayeb, T. Y. Al-Naffouri, and H. R. Bahrami. Compressive sensing for feedback reduction in MIMO broadcast channels. *IEEE Trans. Commun.*, 62(9):3209–3222, Sept. 2014. → pages

[26] Z. Gao, L. Dai, Z. Wang, and S. Chen. Spatially common sparsity based adaptive channel estimation and feedback for FDD massive MIMO. *IEEE Trans. Signal Process.*, 63(23):6169–6183, Dec. 2015. → pages 4

[27] H. Almosa, S. Mosleh, E. Perrins, and L. Liu. Downlink channel estimation with limited feedback for FDD multi-user massive MIMO with spatial channel correlation. In *Proc. IEEE ICC*, pages 1–6, Kansas City, MO, May 2018. → pages 7

[28] Wenqian Shen, Linglong Dai, Yi Shi, Byonghyo Shim, and Zhaocheng Wang. Joint channel training and feedback for FDD massive MIMO systems. *IEEE Trans. Veh. Technol.*, 65(10):8762–8767, Otc. 2016. → pages

[29] X. Rao and V. K. N. Lau. Distributed compressive CSIT estimation and feedback for FDD multi-user massive MIMO systems. *IEEE Trans. Signal Process.*, 62(12):3261–3271, June 2014. → pages 4

[30] A. Liu, F. Zhu, and V. K. N. Lau. Closed-loop autonomous pilot and compressive CSIT feedback resource adaptation in multi-user FDD massive MIMO systems. *IEEE Trans. Signal Process.*, 65(1):173–183, Jan. 2017. → pages 4

[31] E. J. Candès, J. K. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Commun. Pure Appl. Math.*, 59(8):1207–1223, Aug. 2006. → pages 4, 21

[32] Y. Wang, Z. Tian, S. Feng, and P. Zhang. Efficient channel statistics estimation for millimeter-wave MIMO systems. In *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, pages 3411–3415, Mar. 2016. → pages 4

[33] Z. Gao, C. Hu, L. Dai, and Z. Wang. Channel estimation for millimeter-wave massive MIMO with hybrid precoding over frequency-selective fading channels. *IEEE Commun. Lett.*, 20(6):1259–1262, June 2016. → pages 4

[34] X. Gao, L. Dai, S. Han, C. I, and X. Wang. Reliable beamspace channel estimation for millimeter-wave massive MIMO systems with lens antenna array. *IEEE Trans. Wireless Commun.*, 16(9):6010–6021, Sept. 2017. → pages

[35] J. Rodríguez-Fernández, N. González-Prelcic, K. Venugopal, and R. W. Heath Jr. Frequency-domain compressive channel estimation for frequency-selective hybrid millimeter wave MIMO systems. *IEEE Trans. Wireless Commun.*, 17(5):2946–2960, May 2018. → pages

[36] J. P. González-Coma, J. Rodríguez-Fernández, N. González-Prelcic, L. Castedo, and R. W. Heath Jr. Channel estimation and hybrid precoding for frequency selective multiuser mmWave MIMO systems. *IEEE J. Sel. Topics Signal Process.*, 12(2):353–367, May 2018. → pages 4

[37] A. Alkhateeb, G. Leus, and R. W. Heath Jr. Compressed sensing based multi-user millimeter wave systems: How many measurements are needed? In *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, pages 2909–2913, Apr. 2015. → pages 5, 6, 79

[38] J. W. Choi, B. Shim, Y. Ding, B. Rao, and D. I. Kim. Compressed sensing for wireless communications: Useful tips and tricks. *IEEE Commun. Surveys Tut.*, 19(3):1527–1550, Third Quart. 2017. → pages 5, 6

[39] R. A. DeVore. Deterministic constructions of compressed sensing matrices. *Journal of complexity*, (4):918=925, 2007. → pages 5, 6, 23

[40] J. Bourgain, S. Dilworth, K. Ford, S. Konyagin, , and D. Kutzarova. Explicit constructions of RIP matrices and related problems. *Duke Mathematical Journal*, 159(1):145–185, 2011. → pages 23

[41] Z. Xu. Deterministic sampling of sparse trigonometric polynomials. *Journal of Complexity*, 27:133–140, 2011. → pages 5, 6

[42] R. Obermeier and J. A. Martinez-Lorenzo. Sensing matrix design via mutual coherence minimization for electromagnetic compressive imaging applications. *IEEE Trans. Comput. Imag.*, 3(2):217–229, June 2017. → pages 5, 6

[43] M. Lotfi and M. Vidyasagar. A fast noniterative algorithm for compressive sensing using binary measurement matrices. *IEEE Trans. Signal Process.*, 66(15):4079–4089, Aug. 2018. → pages 5, 6

[44] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad. Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition. In *Conf. Rec. 27th Asilomar Conf. Signals, Syst. Comput.*, volume 1, pages 40–44, 1993. → pages 6, 25

[45] Joel A. Tropp and Anna C. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Trans. Inform. Theory*, 53(12):4655–4666, Dec. 2007. → pages 6, 56

[46] D. Needell and J. A. Tropp. CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(3):301–321, 2008. → pages 6, 25

[47] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *The Annals of Statistics*, 32(2):407–499, 2004. → pages 6, 25

[48] Thomas Blumensath and Mike E. Davies. Iterative thresholding for sparse approximations. *Journal of Fourier Analysis and Applications*, 14:629–654, 2008. → pages 6, 34, 56

[49] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imag. Sci.*, 2(1):183–202, 2009. → pages 6

[50] M. A. T. Figueiredo, R. D. Nowak, and S. J. Wright. Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems. *IEEE J. Sel. Topics Signal Process.*, 1(4):586–597, Dec 2007. → pages 6, 43, 54, 56, 73, 82

[51] S. J. Wright, R. D. Nowak, and M. A. T. Figueiredo. Sparse reconstruction by separable approximation. *IEEE Trans. Signal Process.*, 57(7):2479–2493, July 2009. → pages 6, 34

[52] H. Almosa, S. Mosleh, E. Perrins, and L. Liu. Modified cs-based downlink channel estimation with temporal correlation in FDD massive MIMO systems. In *Wireless Telecommunications Symposium (WTS)*, pages 1–7, Phoenix, AZ, USA, Apr. 2018. → pages 7

[53] T. Wang, C. Wen, S. Jin, and G. Y. Li. Deep learning-based CSI feedback approach for time-varying massive MIMO channels. *IEEE Wireless Commun. Lett.*, 8(2):416–419, Apr. 2019. → pages 7

[54] C. K. Wen, W. T. Shih, and S. Jin. Deep learning for massive MIMO CSI feedback. *IEEE Wireless Commun. Lett.*, 7(5):748–751, Oct. 2018. → pages

[55] Q. Yang, M. B. Mashhadi, and D. Gündüz. Deep convolutional compression for massive MIMO CSI feedback. In *2019 IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6, Oct. 2019. → pages 7

[56] H. He, C.-K. Wen, S. Jin, and G. Y. Li. Deep learning-based channel estimation for beamspace mmWave massive MIMO systems. *IEEE Wireless Commun. Lett.*, 7(5):852–855, Otc. 2018. → pages 8

[57] C.-J. Chun, J.-M. Kang, and Il-Min Kim. Deep learning-based joint pilot design and channel estimation for multiuser MIMO channels. *IEEE Commun. Lett.*, 23(11):1999–2003, Nov. 2019. → pages 7, 8

[58] H. Huang, J. Yang, H. Huang, Y. Song, and G. Gui. Deep learning for super-resolution channel estimation and DOA estimation based massive

MIMO system. *IEEE Trans. Veh. Technol.*, 67(9):8549–8560, Sept. 2018. → pages

[59] C. Lu, W. Xu, H. Shen, J. Zhu, and K. Wang. MIMO channel information feedback using deep recurrent network. *IEEE Commun. Lett.*, 23(1):188–191, Jan. 2019. → pages

[60] Y. Jang, G. Kong, M. Jung, S. Choi, and I. Kim. Deep autoencoder based CSI feedback with feedback errors and feedback delay in FDD massive MIMO systems. *IEEE Wireless Commun. Lett.*, 8(3):833–836, June 2019. → pages

[61] Q. Cai, C. Dong, and K. Niu. Attention model for massive MIMO CSI compression feedback and recovery. In *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–5, Marrakesh, Morocco, Apr. 2019. → pages

[62] Z. Liu, L. Zhang, and Z. Ding. Exploiting bi-directional channel reciprocity in deep learning for low rate massive MIMO CSI feedback. *IEEE Wireless Commun. Lett.*, 8(3):889–892, June 2019. → pages 7

[63] H. He, S. Jin, C. Wen, F. Gao, G. Y. Li, and Z. Xu. Model-driven deep learning for physical layer communications. *IEEE Wireless Commun.*, 26(5):77–83, Oct. 2019. → pages 7

[64] J. Hershey, J. L. Roux, and F. Weninger. Deep unfolding: Model-based inspiration of novel deep architectures. *Mitsubishi Electric Research Laboratories, Tech. Rep. TR2014-117*, Aug. 2014. → pages 7, 34, 67

[65] D. L. Donoho, A. Maleki, and A. Montanari. Message passing algorithms for compressed sensing. *Natl. Acad. Sci.*, 106(45):18914–18919, 2009. → pages 8, 34

[66] C. Metzler, A. Mousavi, and R. Baraniuk. Learned D-AMP: Principled neural network based compressive image recovery. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1772–1783, Long Beach, CA, USA, 2017. → pages 8, 34

[67] C.-J. Chun, J.-M. Kang, and Il-Min Kim. Deep learning-based channel estimation for massive MIMO systems. *IEEE Commun. Lett.*, 8(4):1228–1231, Aug. 2019. → pages 8, 105

[68] X. Ma and Z. Gao. Data-driven deep learning to design pilot and channel estimator for massive MIMO. *IEEE Trans. Veh. Technol.*, 69(5):5677–5682, 2020. → pages 8, 101

[69] M. B. Mashhadi and D. Gündüz. Pruning the pilots: Deep learning-based pilot design and channel estimation for MIMO-OFDM systems. *IEEE Trans. Wireless Commun*, Early Access, 2021. → pages 8

[70] X. Ma, Z. Gao, F. Gao, and M. D. Renzo. Model-driven deep learning based channel estimation and feedback for millimeter-wave massive hybrid MIMO systems. *IEEE J. Sel. Areas Commun.*, 39(8):2388–2406, 2021. → pages 8

[71] David Tse and Viswanath Pramod. *Fundamentals of Wireless Communication*. Cambridge Univ. Press, Cambridge, U.K., 2005. → pages 13, 18

[72] Irina Rish and Genady Ya. Grabarnik. *Sparse Modeling: Theory, Algorithms, and Applications*. Chapman & Hall/CRC Press, Boca Raton, Florida, 2015. → pages 21, 24, 25, 28, 38

[73] E. J. Candès, J. K. Romberg, and T. Tao. Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Inform. Theory*, 52(2):489–509, 2006. → pages 21, 25

[74] Y. C. Eldar and G. Kutyniok. *Compressed Sensing: Theory and Applications*. Cambridge Univ. Press, 2012. → pages 22, 23, 24, 25

[75] J. Haupt, L. Applebaum, and R. Nowak. On the restricted isometry of deterministically subsampled Fourier matrices. In *Proc. 44th Annual Conference on Information Sciences and Systems (CISS)*, Princeton, NJ, Mar 2010. → pages 23

[76] P. Indyk. Explicit constructions for compressed sensing of sparse signals. In *Proc. ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 30–33, San Francisco, CA, Jan. 2008. → pages 23

[77] J. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, (1):1–21, 2010. → pages 26

[78] S. Boyd, L. Xiao, and A. Mutapcic. Subgradient methods. *Notes for EE392o Stanford University Autumn*, 2003-2004. → pages 26, 67, 68

[79] I. Sutskever, J. Martens, G. Dahl, and G. Hinton. On the importance of initialization and momentum in deep learning. pages 1139–1147, 2013. → pages 33

[80] D.P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR*, San Diego, CA, May 2015. → pages 33

[81] G. Hinton, N. Srivastava, and K. Swersky. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. *Univ. Toronto, Toronto, ON, Canada*, 2012. → pages 33

[82] K. Gregor and Y. LeCun. Learning fast approximations of sparse coding. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*, pages 399–406, 2010. → pages 34

[83] Jun-ya Gotoh, Akiko Takeda, and Katsuya Tono. DC formulations and algorithms for sparse optimization problems. *Mathematical Programming*, 169(1):141–176, 2018. → pages 38, 41

[84] P. D. Tao and L. T. H. An. Convex analysis approach to DC programming: Theory, algorithms and applications. *Acta Mathematica Vietnamica*, 22(1):289–355, 1997. → pages 41

[85] B. K. Sriperumbudur and G. R. Lanckriet. A proof of convergence of the concave-convex procedure using Zangwill's theory. *Neural Computation*, 24(6):1391–1407, 2012. → pages 41

[86] Hoai An Le Thi and Tao Pham Dinh. DC programming and DCA: Thirty years of developments. *Math. Program.*, 169:5–68, 2018. → pages 41, 49

[87] T. P. Dinh and H. A. L. Thi. Recent advances in DC programming and DCA. *Transactions on Computational Intelligence XIII*, 8432:1–37, 2014. → pages 41

[88] R. W. Heath Jr. and Angel Lozano. *Foundations of MIMO Communication.* Cambridge Univ. Press, Cambridge, U.K., 2018. → pages 52, 53

[89] L. Dai, X. Gao, S. Han, C.-L. I, and X. Wang. Beamspace channel estimation for millimeter-wave massive MIMO systems with lens antenna array. In *IEEE Int. Conf. Commun. China (ICCC)*, Chengdu, China, July 2016. → pages 56, 59

[90] Jie Yang, Chao-Kai Wen, Shi Jin, and Feifei Gao. Beamspace channel estimation in mmWave systems via cosparse image reconstruction technique. *IEEE Trans. Commun.*, 66(10):4767–4782, Otc. 2018. → pages 56, 59

[91] Z. Qin, H. Ye, G. Y. Li, and B. F. Juang. Deep learning in physical layer communications. *IEEE Wireless Commun.*, 26(2):93–99, Apr. 2019. → pages 66

[92] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pages 770–778, Las Vegas, NV, USA, June 2016. → pages 70

[93] E. J. Candes and T. Tao. Decoding by linear programming. *IEEE Trans. Inform. Theory*, 51(12):4203–4215, Dec. 2005. → pages 73

[94] Pengxia Wu, Hui Ma, and Julian Cheng. Sparse channel reconstruction with nonconvex regularizer via DC programming for massive MIMO

systems. In *IEEE Globecom*, Taipei, Taiwan, Dec. 2020. → pages 73, 80

[95] D. Zhu, J. Choi, and R. W. Heath. Auxiliary beam pair enabled AoD and AoA estimation in closed-loop large-scale millimeter-wave MIMO systems. *IEEE Trans. Wireless Commun.*, 16(7):4770–4785, July 2017. → pages 74

[96] K. Tono, A. Takeda, and J. Gotoh. Efficient DC algorithm for constrained sparse optimization. 2017. [Online]. Available: arXiv:1701.08498. → pages 89

[97] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley, New York, 1991. → pages 99

[98] C. E. Shannon. A mathematical theory of communication. *ACM SIGMOBILE Mobile Comput. Commun. Rev.*, (1):3–55, 2001. → pages 99

Appendix A

Proof of Theorem 3.1

Suppose that $\mathbf{x}_{\rho^*}$ is an optimal solution to (3.6) given ρ^*, then $\mathbf{x}_{\rho^*}$ is also optimal to (3.5) as long as $\|\mathbf{x}_{\rho^*}\|_0 \leq K$ (or $\|\mathbf{x}_{\rho^*}\|_1 - \|\mathbf{x}_{\rho^*}\|_{K,1} = 0$) is satisfied. Thus, we only need to prove that $\|\mathbf{x}_{\rho^*}\|_0 > K$ is infeasible. We assume that $\mathbf{x}_{\rho^*}$ is an optimal solution to (3.6) with $\rho^* > \max_i\{|(\mathbf{\Phi}^T\mathbf{y})_i| + q(\|\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{e}_i\|_2 + |(\mathbf{\Phi}^T\mathbf{\Phi})_{ii}|/2)\}$. For $\|\mathbf{x}_{\rho^*}\|_0 > K$, we construct a feasible solution to (3.5) as $\mathbf{x}' = \mathbf{x}_{\rho^*} - x_i\mathbf{e}_i$, where i represents the index of the ith-largest-magnitude element of vector $\mathbf{x}_{\rho^*}$; $\mathbf{e}_i$ represents the unit vector in which the ith element is one while the other elements are zeros and x_i represents the ith element of vector $\mathbf{x}_{\rho^*}$. By writing the objective of (3.6) as

$$F(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{x} - (\mathbf{\Phi}^T\mathbf{y})^T\mathbf{x} + \rho^*\|\mathbf{x}\|_1 - \rho^*\|\mathbf{x}\|_{K,1} \tag{A.1}$$

we have

$$\begin{aligned}
&F(\mathbf{x}_{\rho^*}) - F(\mathbf{x}') \\
&= F(\mathbf{x}_{\rho^*}) - F(\mathbf{x}_{\rho^*} - x_i\mathbf{e}_i) \\
&= \frac{1}{2}\mathbf{x}_{\rho^*}^T\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{x}_{\rho^*} - (\mathbf{\Phi}^T\mathbf{y})^T\mathbf{x}_{\rho^*} - \frac{1}{2}(\mathbf{x}_{\rho^*} - x_i\mathbf{e}_i)^T\mathbf{\Phi}^T\mathbf{\Phi}(\mathbf{x}_{\rho^*} - x_i\mathbf{e}_i) \\
&\quad + (\mathbf{\Phi}^T\mathbf{y})^T(\mathbf{x}_{\rho^*} - x_i\mathbf{e}_i) + \rho^*|x_i| \\
&= x_i\mathbf{e}_i^T\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{x}_{\rho^*} - \frac{1}{2}x_i^2(\mathbf{\Phi}^T\mathbf{\Phi})_{i,i} - x_i(\mathbf{\Phi}^T\mathbf{y})_i + \rho^*|x_i| \\
&\geq -x_i\mathbf{e}_i^T\mathbf{\Phi}^T\mathbf{\Phi}\mathbf{x}_{\rho^*} - \frac{1}{2}x_i^2(\mathbf{\Phi}^T\mathbf{\Phi})_{i,i} - x_i(\mathbf{\Phi}^T\mathbf{y})_i + \rho^*|x_i| \\
&\geq -|x_i|\|\mathbf{e}_i^T\mathbf{\Phi}^T\mathbf{\Phi}\|_2\|\mathbf{x}_{\rho^*}\|_2 - \frac{1}{2}|x_i|\|\mathbf{x}_{\rho^*}\|_2|(\mathbf{\Phi}^T\mathbf{\Phi})_{i,i}| - |x_i| \cdot |(\mathbf{\Phi}^T\mathbf{y})_i| + \rho^*|x_i| \\
&= |x_i|\big(\rho - \|\mathbf{e}_i^T\mathbf{\Phi}^T\mathbf{\Phi}\|_2\|\mathbf{x}_{\rho^*}\|_2 - \frac{1}{2}\|\mathbf{x}_{\rho^*}\|_2|(\mathbf{\Phi}^T\mathbf{\Phi})_{i,i}| - |(\mathbf{\Phi}^T\mathbf{y})_i|\big) > 0
\end{aligned} \tag{A.2}$$

which contradicts the assumption that $\mathbf{x}_{\rho^*}$ is an optimal solution to (3.6).

Appendix B

Proof of Theorem 3.2

We first derive the Lipschitz constant $l = \lambda_{\max}(\mathbf{\Phi}^T\mathbf{\Phi})$, then we prove the convexity of $h(\mathbf{x})$. We denote the LS objective by the function $w(\mathbf{x}) = \frac{1}{2}\|\mathbf{y} - \mathbf{\Phi}\mathbf{x}\|_2^2$. Since $w(\mathbf{x})$ is smooth if and only if its gradient function is Lipschitz continuous, we assume there exists $l < \infty$, which is named as a Lipschitz constant, such that

$$\|\nabla w(\mathbf{x}) - \nabla w(\mathbf{z})\|_2 \leq l\|\mathbf{x} - \mathbf{z}\|_2. \tag{B.1}$$

For $w(\mathbf{x}) = \frac{1}{2}\|\mathbf{y} - \mathbf{\Phi}\mathbf{x}\|_2^2$, we have

$$\begin{aligned} \|\nabla w(\mathbf{x}) - \nabla w(\mathbf{z})\|_2 &= \|\mathbf{\Phi}^T(\mathbf{\Phi}\mathbf{x} - \mathbf{y}) - \mathbf{\Phi}^T(\mathbf{\Phi}\mathbf{z} - \mathbf{y})\|_2 \\ &= \|\mathbf{\Phi}^T\mathbf{\Phi}(\mathbf{x} - \mathbf{z})\|_2 \\ &\leq \left|\left|\left|\mathbf{\Phi}^T\mathbf{\Phi}\right|\right|\right|_2 \|\mathbf{x} - \mathbf{z}\|_2 \\ &= \lambda_{max}(\mathbf{\Phi}^T\mathbf{\Phi})\|\mathbf{x} - \mathbf{z}\|_2 \end{aligned} \tag{B.2}$$

where $\left|\left|\left|\cdot\right|\right|\right|_2$ represents the spectral norm of a matrix, and $\lambda_{max}(\cdot)$ represents the largest eigenvalue of a matrix. Thus, we obtain the Lipschitz constant $l = \lambda_{\max}(\mathbf{\Phi}^T\mathbf{\Phi})$.

For $h(\mathbf{x}) = \frac{l}{2}\|\mathbf{x}\|_2^2 - \frac{1}{2}\|\mathbf{y} - \mathbf{\Phi}\mathbf{x}\|_2^2$, we have the Hessian matrix $\nabla^2 h(\mathbf{x})$ as

$$\begin{aligned} \nabla^2 h(\mathbf{x}) &= \frac{\partial(\nabla h(\mathbf{x}))}{\partial \mathbf{x}} \\ &= \frac{\partial(l\mathbf{x}^T - \mathbf{\Phi}^T(\mathbf{\Phi}\mathbf{x} - \mathbf{y}))}{\partial \mathbf{x}} \\ &= l\mathbf{I} - \mathbf{\Phi}^T\mathbf{\Phi} \end{aligned} \tag{B.3}$$

where $\mathbf{I}$ is the identity matrix. For $l = \lambda_{\max}(\mathbf{\Phi}^T\mathbf{\Phi})$, we have $l\mathbf{I} - \mathbf{\Phi}^T\mathbf{\Phi} \succeq \mathbf{0}$,

which means that the Hessian matrix $l\mathbf{I} - \mathbf{\Phi}^T\mathbf{\Phi}$ of $h(\mathbf{x})$ is semidefinite positive. Thus, $h(\mathbf{x})$ is a convex function of vector $\mathbf{x}$.

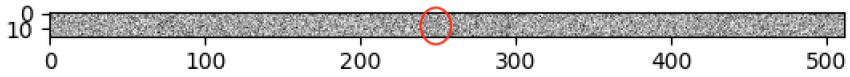
Appendix C

A Toy Experiment With a Manually Structured Dataset

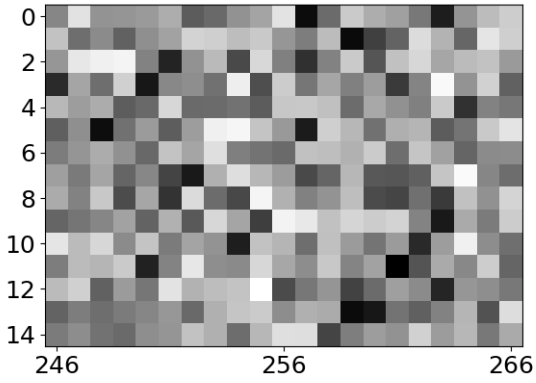
To show the data adaption of the data-driven matrices visually, we construct a toy dataset that has a certain visualizable structural feature. We randomly generate $20,000$ spatial-domain channel realizations [30] according to the channel model in (2.10). We transform the generated spatial-domain channel vectors into beamspace channel vectors $\mathbf{x}' \in \mathbb{R}^{256}$, where the sparsity level is set to be three, i.e., $S = 3$. Furthermore, we construct a dataset consisting of the sample vectors $\mathbf{x}_c = [\mathbf{x}'^T, \mathbf{0}^T]^T$, where $\mathbf{x}_c \in \mathbb{R}^{512}$, $\mathbf{x}' \in \mathbb{R}^{256}$ and the all-zero vectors $\mathbf{0}^T$ has the length 256. We use this manually structured dataset to train the *BP-sAE* and *BP-gAE* autoencoders proposed in Chapter 4, and visually illustrate the learned data-driven measurement matrices. The compressed dimension is configured to be $M = 15$. For model training, we initialize the trainable measurement matrix using the Gaussian matrix $\mathbf{G} \in \mathbb{R}^{15\times512}$ in which the elements follow a standard Gaussian distribution with row-wise normalization. After training, the data-driven matrix learned by *BP-sAE* is denoted by $\mathbf{\Phi}'_{sae} \in \mathbb{R}^{15\times512}$; the data-driven matrix learned by *BP-gAE* is denoted by $\mathbf{\Phi}'_{gae} \in \mathbb{R}^{15\times512}$.

We illustrate the Gaussian matrix $\mathbf{G}$, the learned matrices $\mathbf{\Phi}'_{sae}$ and $\mathbf{\Phi}'_{gae}$ in Fig. C.1, Fig. C.2 and Fig. C.3. The greyscale images for the matrix $\mathbf{G}$, $\mathbf{\Phi}'_{sae}$ and $\mathbf{\Phi}'_{gae}$ are shown in Fig. C.1 (a), Fig. C.2 (a) and Fig. C.3 (a), respectively. The greyscale for Gaussian matrix $\mathbf{G}$ exhibits a uniform pat-

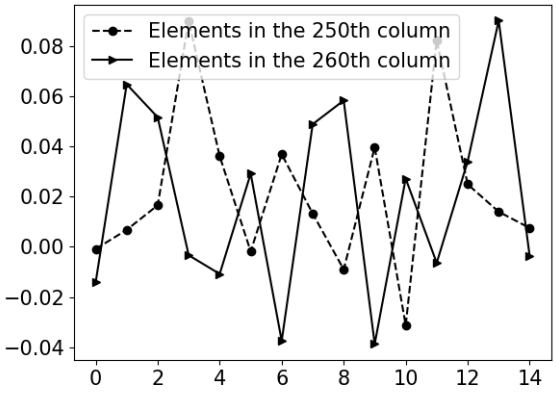
[30] The number of scattering clusters is set to be three; the AoDs of scattering clusters follow a uniform distribution between $[-\pi/2, \pi/2]$; the scattering clusters include one line-of-sight path and two non-line-of-sight paths; the Ricean K-factor is set to be 13.2dB; the complex path gains follow a zero-mean Gaussian distribution; the number of antennas at the BS is set to be 256.



(a) Greyscale illustration of the random Gaussian matrix **G**

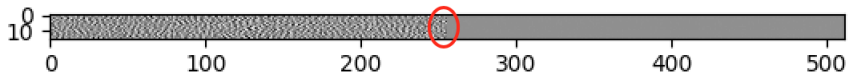


(b) Zoomed in region of the matrix **G** which is denoted by the red circle in (a)

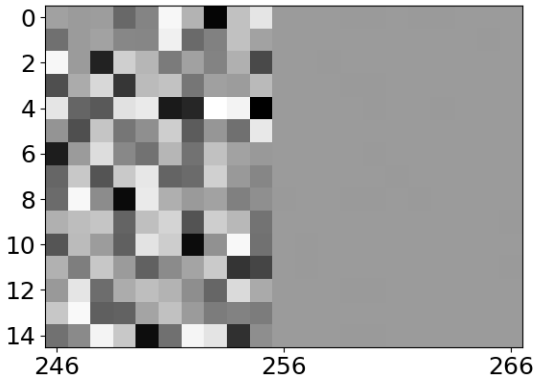


(c) Plot of sample columns of the matrix **G**

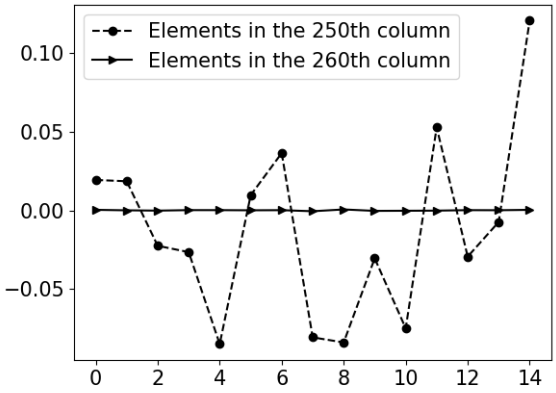
Figure C.1: A visual illustration of a random Gaussian matrix **G**



(a) Greyscale illustration of the learned matrix $\mathbf{\Phi}'_{sae}$

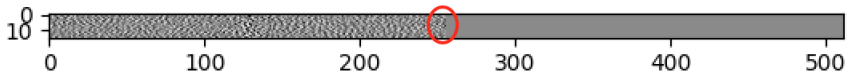


(b) Zoomed in region of the matrix $\mathbf{\Phi}'_{sae}$ which is denoted by the red circle in (a)

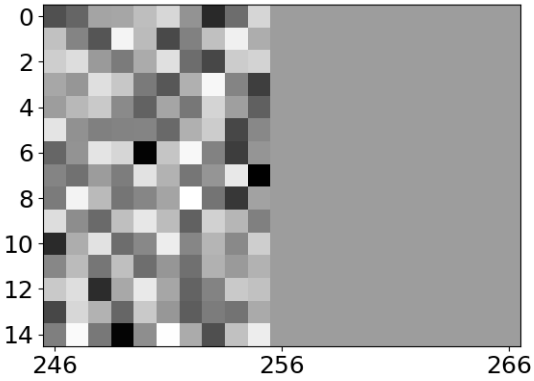


(c) Plot of sample columns of the matrix $\mathbf{\Phi}'_{sae}$

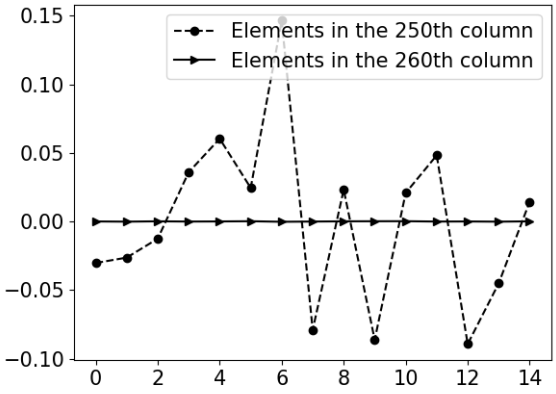
Figure C.2: A visual illustration of the data-driven matrix $\mathbf{\Phi}'_{sae}$ acquired via training *BP-sAE*



(a) Greyscale illustration of the learned matrix $\mathbf{\Phi}'_{gae}$



(b) Zoomed in region of the matrix $\mathbf{\Phi}'_{gae}$ which is denoted by the red circle in (a)



(c) Plot of sample columns of the matrix $\mathbf{\Phi}'_{gae}$

Figure C.3: A visual illustration of the data-driven matrix $\mathbf{\Phi}'_{gae}$ acquired via training *BP-gAE*

tern for the whole matrix, while the greyscales for the learned matrices $\mathbf{\Phi}'_{sae}$ and $\mathbf{\Phi}'_{gae}$ show distinct patterns for the first-half and second-half columns. To show the differences more clearly, we zoom in on the region between the 246th and 266th columns. The zoom-in images for the matrix $\mathbf{G}$, $\mathbf{\Phi}'_{sae}$ and $\mathbf{\Phi}'_{gae}$ are shown in Fig. C.1 (b), Fig. C.2 (b) and Fig. C.3 (b), respectively. Each small grey square represents an element in the matrix, and different grey-scaled colors indicate different element values with darker colors indicate larger values. In Fig. C.2 (b) and Fig. C.3 (b), the columns from the 246th to the 256th column have the elements with different values, while the columns from the 256th to the 256th column show a unique value. Then, we select the 250th column and the 260th column which are in first-half and second-half columns respectively, and plot their element values in Fig. C.1 (c), Fig. C.2 (c) and Fig. C.3 (c), respectively. The plot of Fig. C.1 (c) shows the element values for the initialized Gaussian matrix $\mathbf{G}$, while the plots in Fig. C.2 (c) and Fig. C.3 (c) show the element values for the learned matrices $\mathbf{\Phi}'_{sae}$ and $\mathbf{\Phi}'_{gae}$. The 260th columns for $\mathbf{\Phi}'_{sae}$ and $\mathbf{\Phi}'_{gae}$ are all zero-valued, which implies the second-half columns from the 256th to the 512th column are all zeros. This result indicates the learned matrices can adapt to our manually structured feature of the dataset. The all-zero columns of the learned measurement matrices are adaptive with the all-zero elements of the concatenating vectors $\mathbf{x}_c = [\mathbf{x}', \mathbf{0}]^T$.

We evaluate the reconstruction performance of the Gaussian matrix $\mathbf{G}$ and the learned matrices $\mathbf{\Phi}'_{sae}$ and $\mathbf{\Phi}'_{gae}$, and compare them in Table C.1. We can observe that the reconstruction accuracy improves from 12.4% for the Gaussian matrix $\mathbf{G}$ to 93.67% for the matrix $\mathbf{\Phi}'_{sae}$ and 96% for the matrix $\mathbf{\Phi}'_{gae}$. Furthermore, the NMSE decreases from 0.36 for the Gaussian matrix $\mathbf{G}$ to 0.08 for the matrix $\mathbf{\Phi}'_{sae}$ and 0.05 for the matrix $\mathbf{\Phi}'_{gae}$. When compared with the random Gaussian matrix $\mathbf{G}$, the reconstruction performance of the data-driven matrices $\mathbf{\Phi}'_{sae}$ and $\mathbf{\Phi}'_{gae}$ are significantly improved.

Table C.1: NMSE and Accurate Reconstruction Percentage Comparisons

Matrix	NMSE	Reconstruction percentage
Gaussian matrix $\mathbf{G}$	0.36	12.4%
Data-driven matrix $\mathbf{\Phi}'_{sae}$	0.08	93.67%
Data-driven matrix $\mathbf{\Phi}'_{gae}$	0.05	96%