SPARSE STRUCTURE LEARNING VIA INFORMATION-THEORETIC REGULARIZATION AND SELF-CONTAINED PROBABILISTIC ESTIMATION

by
Shuai Huang

A dissertation submitted to Johns Hopkins University in conformity with
the requirements for the degree of Doctor of Philosophy

Baltimore, Maryland
February, 2017

© 2017 Shuai Huang
All Rights Reserved

Abstract

Nowadays, there is an increasing amount of digital information constantly generated from every aspect of our life and data that we work with grow in both size and variety. Fortunately, most of the data have sparse structures. Compressive sensing offers us an efficient framework to not only collect data but also to process and analyze them in a timely fashion. Various compressive sensing tasks eventually boil down to the sparse signal recovery problem in an under-determined linear system. To better address the challenges of "big" data using compressive sensing, we focus on developing powerful sparse signal recovery approaches and providing theoretical analysis of their optimalities and convergences in this dissertation.

Specifically, we bring together insights from information theory and probabilistic graphical models to tackle the sparse signal recovery problem from the following two perspectives:

- *Sparsity-regularization approach*: we propose the Shannon entropy function and Rényi entropy function constructed from the sparse signal, and prove that minimizing them does promote sparsity in the recovered signal. Experiments on simulated and real data show that the two proposed entropy function minimization methods outperform state-of-the-art l_p-norm minimization and l_1-norm minimization methods.

- *Probabilistic approach*: we propose the generalized approximate message passing with built-in parameter estimation (PE-GAMP) framework, present its empirical convergence analysis and give detailed formulations to obtain the MMSE and MAP estimations of the sparse signal. Experiments on simulated and real data show that the proposed PE-GAMP is more robust, much simpler and has a wider applicability compared to the popular Expectation Maximization based parameter estimation method.

Dissertation Committee

Professor *Trac D. Tran* (Advisor/First reader)
Department of Electrical and Computer Engineering
Johns Hopkins Whiting School of Engineering

Professor *Mark A. Foster* (Second reader)
Department of Electrical and Computer Engineering
Johns Hopkins Whiting School of Engineering

Professor *Ralph Etienne-Cummings*
Department of Electrical and Computer Engineering
Johns Hopkins Whiting School of Engineering

Acknowledgments

Looking back at my Ph.D. life at Hopkins, there are so many people I'd like to thank for their kindness and help, and I am very fortunate to get to know all of you.

First of all, I'd like to offer my deepest gratitude to my advisor, Dr. Trac D. Tran. I could not have done this without his constant support, guidance and encouragement. I learned so much from him: from discovering and approaching research problems from different perspectives, to writing proposals and giving presentations. His enthusiasm and dedication to the research and students make me could not hope for a better mentor. Additionally, I'd like to thank Dr. Gerard G.L. Meyer, Dr. Glen A. Coppersmith, Dr. Damianos Karakos, Dr. Pablo A. Iglesias for their help and guidances, and my dissertation committee members Dr. Mark A. Foster, Dr. Ralph Etienne-Cummings for their time and efforts in proofreading my dissertation and providing valuable feedback.

I have also been blessed to meet and make friends with a lot of other great people at Hopkins. Ehsan Variani has been there should I needed some help or simply someone to chat with, and the discussions with him about research work and life are always insightful and inspiring. Yixin Gao and Yuan Cao have also been there to help me the whole time, they are both excellent researchers and dedicated to their work, I've always enjoyed those insightful and constructive discussions with them. I also appreciate Luoluo Liu's help all this time. She is also excellent and committed to her research, and I quite enjoy the insightful and refreshing discussions with her. I'd also like to thank every member of the Digital Signal Processing (DSP) laboratory for their help and support, it's been both an honor and a privilege for me to be part of our DSP lab.

Finally, I want to thank my parents for their love and unwavering support to me. They encouraged and instilled in me an eagerness and courage to learn and explore the unknown from an early age, and offered me any help they could get. Thank you, my dear parents!

Table of Contents

Table of Contents vi

List of Figures x

List of Algorithms xiv

I Introduction and background 1

1 Introduction 2

1.1 Sparse signal recovery 4

1.1.1 Sparsity-regularization approach 6

1.1.2 Probabilistic approach 7

1.2 Dissertation goal and contributions 8

1.3 Outline of the dissertation 9

2 The quest for sparsity 11

2.1 Restricted isometry property 11

2.1.1 Noiseless recovery 12

2.1.2 Noisy recovery 12

2.2 Incoherent matrix 13

2.3 Relaxation of the l_0 norm 14
2.4 Input and out channel distributions 15

II Sparsity-regularization approach **18**

3 Sparsity-regularization entropy function **19**
3.1 Introduction to entropy 19
3.2 Entropy function of the sparse signal 21
3.2.1 Shannon entropy function 21
3.2.2 Rényi entropy function 22
3.3 Sparsity promotion analysis 23

4 Sparse signal recovery via entropy function minimization **28**
4.1 Entropy function minimization 28
4.1.1 Generalized iterative shrinkage thresholding 33
4.2 Experimental results 35
4.2.1 Simulated sparse signal recovery 35
4.2.2 Real image recovery 37

5 RPCA via entropy function minimization **44**
5.1 Introduction to RPCA 44
5.2 Entropy function minimization 45
5.3 Experimental results 48

III Probabilistic approach 51

6 GAMP with built-in parameter estimation 52

6.1 Prior work 52

6.2 Message passings in PE-GAMP 53

6.2.1 Sum-product message passing 54

6.2.2 Max-sum message passing 56

6.2.3 The PE-GAMP algorithm 58

6.3 Sum-product PE-GAMP 64

6.3.1 Review on sum-product GAMP update 67

6.3.2 Sum-product parameter update 69

6.4 Max-sum PE-GAMP 70

6.4.1 Review on max-sum GAMP update 71

6.4.2 Max-sum parameter update 72

7 Empirical convergence analysis of the PE-GAMP 73

7.1 Review on the state evolution analysis of the GAMP 73

7.2 The state evolution analysis of the PE-GAMP 77

7.2.1 MAP parameter estimation state evolution 81

7.2.2 MMSE parameter estimation state evolution 82

7.3 Empirical convergence analysis 83

8 Probabilistic sparse signal recovery via PE-GAMP 90

8.1 Max-sum message passing 91

8.2 Sum-product message passing 93

8.2.1 Comparison with EM parameter estimation 100

8.3 Experimental results 102

8.3.1 Simulated sparse signal recovery 102

8.3.2 Real image recovery 105

8.3.3 Non-negative sparse coding for image classification 111

IV Conclusion and future work **114**

9 Conclusion and future work **115**

List of Figures

1.1 (a) The original image "Green Wheat Field with Cypress" by Van Gogh; (b) The magnitudes of the largest 20% wavelet coefficients from the red channel are between 16 $\sim$ 1938, the green and blue channels are similar; (c)The reconstructed image based on the largest 20% wavelet coefficients, with PSNR=32.98 dB. 3

1.2 A probabilistic view of the sparse signal recovery task [22]: The signal $\boldsymbol{x}$ is estimated given the output vector $\boldsymbol{y}$, the channel transition probability functions $p(x_j; \boldsymbol{\lambda})$, $p(y_i; \boldsymbol{\theta}|z_i)$ and the transformation matrix $\boldsymbol{A}$. $\{\boldsymbol{\lambda}, \boldsymbol{\theta}\}$ denote the parameters of the probability models and are usually unknown. 7

2.1 The 2D level plots: (a) the convex $\|\boldsymbol{x}\|_1$; (b) the nonconvex $\|\boldsymbol{x}\|_{0.5}^{0.5}$ 14

3.1 Rényi entropy defined on $\{p_1, 1 - p_1\}$ with different choices of α. . . . 20

3.2 Entropy function $h_p(\boldsymbol{x})$ with $p = 1$ in the 2-dimensional space. 22

3.3 The one-to-one mapping: $\boldsymbol{x} \longleftrightarrow \tilde{\boldsymbol{x}} \longleftrightarrow \ddot{\boldsymbol{x}}$ when $p = 0.5$. 24

3.4 The one-to-one mapping: $\mathcal{X}^{\epsilon} \longleftrightarrow \tilde{\boldsymbol{x}} \longleftrightarrow \ddot{\boldsymbol{x}}$ when $p = 0.5$. 27

4.1 (a) The phase transition curves (PTC) of different sparsity regularization approaches in the noiseless case; (b) The signal-to-noise-ratio (SNR) of the recovered signal $\hat{\boldsymbol{x}}$ using different sparsity regularization approaches in the noisy case. 36

4.2 The real images used in the recovery experiments: (a) Barbara; (b) Boat; (c) Lena; (d) Peppers. 37

4.3 The peak-signal-to-noise-ratio (PSNR) of the recovered images from "noiseless" measurements using different sparsity regularization approaches. (a) Barbara; (b) Boat; (c) Lena; (d) Peppers. 40

4.4 The peak-signal-to-noise-ratio (PSNR) of the recovered images from "noisy" measurements using different sparsity regularization approaches. (a) Barbara; (b) Boat; (c) Lena; (d) Peppers. 41

4.5 The recovered "Lena" image from noiseless measurements using different approaches with a sampling rate of 0.2: a) l_1 norm PSNR=27.81 dB; (b) l_p norm PSNR=29.63 dB; (c) Shannon entropy function $h_p(\boldsymbol{x})$ PSNR=30.79; (d) Rényi entropy function $h_{p,\alpha}(\boldsymbol{x})$ PSNR=30.75 dB. 42

4.6 The recovered "Lena" image from noisy measurements using different approaches with a sampling rate of 0.2: a) l_1 norm PSNR=27.58 dB; (b) l_p norm PSNR=29.21 dB; (c) Shannon entropy function $h_p(\boldsymbol{x})$ PSNR=30.19; (d) Rényi entropy function $h_{p,\alpha}(\boldsymbol{x})$ PSNR=30.15 dB. 43

5.1 The recovery of a $M \times M$ low rank matrix $\boldsymbol{L}$ corrupted by sparse noisy matrix $\boldsymbol{E}$ using RPCA. 50

6.1 The factor graph for the proposed PE-GAMP. "■" represents the factor node, and "◯" represents the variable node. Here, $\boldsymbol{\lambda} = \{\lambda_1, \cdots, \lambda_L\}$ and $\boldsymbol{\theta} = \{\theta_1, \cdots, \theta_K\}$ are the parameters whereas $\boldsymbol{x} = [x_1, \cdots, x_N]^{\mathrm{T}}$ is the sparse signal. 54

8.1 The phase transition curves (PTC) of different GAMP methods in the noiseless case. (a) Bernoulli-Gaussian (BG) sparse signal; (b) Bernoulli-Exponential (BE) sparse signal. 103

8.2 The signal-to-noise-ratio (SNR) of the recovered sparse signals using different GAMP methods in the noisy case. (a) Bernoulli-Gaussian (BG) sparse signal; (b) Bernoulli-Exponential (BE) sparse signal. 104

8.3 The peak-signal-to-noise-ratio (PSNR) of the recovered images from "noiseless" measurements using different GAMP methods. (a) Barbara; (b) Boat; (c) Lena; (d) Peppers. 107

8.4 The peak-signal-to-noise-ratio (PSNR) of the recovered images from "noisy" measurements using different GAMP methods. (a) Barbara; (b) Boat; (c) Lena; (d) Peppers. 108

8.5 The recovered lena image using different approaches with a sampling rate of 0.05 from noiseless measurements: a) EM-GAMP (BGm prior) 5.93 dB; (b) PE-GAMP (Laplace prior) 16.36 dB; (c) PE-GAMP (BGm prior) 20.50 dB. 109

8.6 The recovered lena image using different approaches with a sampling rate of 0.2 from noiseless measurements: a) EM-GAMP (BGm prior) 26.05 dB; (b) PE-GAMP (Laplace prior) 24.39 dB; (c) PE-GAMP (BGm prior) 26.06 dB. 109

8.7 The recovered lena image using different approaches with a sampling rate of 0.05 from noisy measurements: a) EM-GAMP (BGm prior) 5.92 dB; (b) PE-GAMP (Laplace prior) 16.17 dB; (c) PE-GAMP (BGm prior) 20.35 dB. 110

8.8 The recovered lena image using different approaches with a sampling rate of 0.2 from noisy measurements: a) EM-GAMP (BGm prior) 25.55 dB; (b) PE-GAMP (Laplace prior) 23.91 dB; (c) PE-GAMP (BGm prior) 25.55 dB. 110

8.9 Low-level SIFT features are densely sampled from local image patches. 112

8.10 The confusion matrix of the classification results on Caltech-101 dataset. 113

List of Algorithms

1 Sparse signal recovery via entropy function minimization 32

2 RPCA via entropy function minimization 49

3 The PE-GAMP algorithm 62

4 The PE-GAMP state evolution 78

5 MAP parameter estimation via line search method 94

Part I

Introduction and background

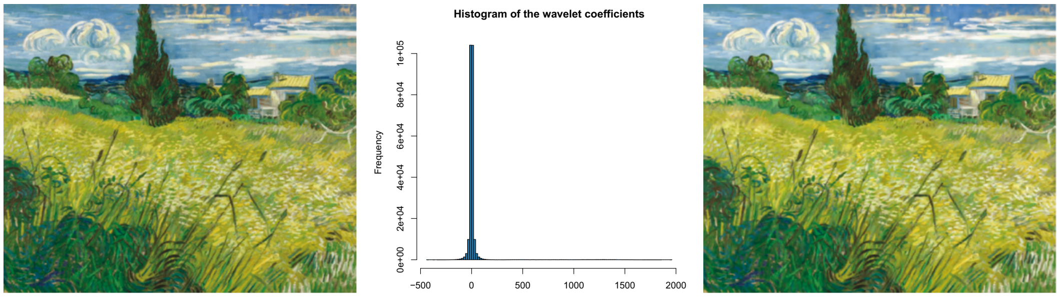
Chapter 1

Introduction

Technology advancement in data acquisition and sharing over the past decade has made it possible to collect vast amount of data from every aspect of modern society. Let it be the information passed across the Internet, the medical/health records gathered from wearable devices, or some specialized data collected in the lab, never before in history has data become so readily available and abundant. Yet this is both a blessing and a curse given the fact that the tools to *process* and *analyze* those vast amount of data are often in shortage, and the challenges we face nowadays eventually boil down to answering the following two questions:

- How to sample/acquire the data in a fast and efficient way?
- How to analyze the data so that we can make the best use of it?

Fortunately, a lot of the data have approximately *sparse* forms in the sense that they can be concisely represented by a few elements of a certain basis. Take one of the most common data, images, for example, at first glance, an image is a 2-dimensional or 3-dimensional dense matrix and it's hard to believe it could be sparse. However, if we inspect it in the wavelet domain, its wavelet coefficient vector is sparse: the "magnitudes" of its entries are mostly close to 0, as is shown in Figure 1.1, and the image can be reconstructed fairly well using just those



(a) (b) (c)

Figure 1.1: (a) The original image "Green Wheat Field with Cypress" by Van Gogh; (b) The magnitudes of the largest 20% wavelet coefficients from the red channel are between $16 \sim 1938$, the green and blue channels are similar; (c)The reconstructed image based on the largest 20% wavelet coefficients, with PSNR=32.98 dB.

large-magnitude coefficients.

For the data that have sparse structures, compressive sensing [1, 2] could provide answers for both questions. First of all, compressive sensing recovers the sparse structure of the data at a sampling rate much lower than the Nyquist rate, and thus enjoys much popularity in applications where a high sampling rate is expensive or impractical. For example, applying it to magnetic resonance imaging (MRI) could greatly reduce the scan time one needs to stay in the machine by acquiring much fewer samples [3, 4], which not only lowers the costs but also makes the whole process more comfortable for the patient. Besides, it plays an important role in designing and implementation of high-speed analog-to-digital converters (ADC) nowadays: reduced sampling rate makes acquiring higher frequency signals possible within the current capabilities of ADC device [5–7].

Secondly, compressive sensing also presents a novel perspective to analyze the data via its recovered sparse structure. Take the classification task for example, based on the observation that data samples can be *better* approximated through a sparse linear combinations of other samples in the same class compared to those from a different class, sparse representation based classification (SRC) has been shown to produce robust and accurate classification results in applications such as face recognition [8–10], hyperspectral image classification [11, 12], etc. Additionally, by enforcing the presence of sparse structure in the data, we can perform post-processing operations such as denoising, matrix completion, etc. on the data.

In this dissertation, we focus on developing powerful sparse signal recovery approaches in compressive sensing and providing theoretical analysis of their optimalities and convergences. In the effort to promote sparseness in the recovered signals, we bring together insights from information theory and probabilistic graphical models to tackle the recovery problem. Eventually we would like to provide feasible and sound solutions to address the challenges imposed by the explosion of data.

1.1 Sparse signal recovery

Compressive sensing revolves around the sparse signal recovery problem, which lays the foundation for applications such as dictionary learning, SRC, low rank matrix recovery, etc. A signal $\boldsymbol{s} \in \mathbb{R}^N$ is considered to be "sparse" if it satisfies either of the following two conditions:

- $\boldsymbol{s}$ contains only a few nonzero coefficients.
- $\boldsymbol{s}$ can be represented through a linear combination of a few components from some proper basis.

In real life, signals that have mostly small coefficients and only a few *relatively* large coefficients also fall into the category of "sparse" signals. Here the small or large coefficient is defined with regards to the *magnitude* of said coefficient. Specifically, let $\boldsymbol{V} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_N]$ denote a basis that spans the vector space where the signal $\boldsymbol{s}$ is sparse:

$$\boldsymbol{s} = \boldsymbol{V}\boldsymbol{x} = \textstyle\sum_{i=1}^{N} \boldsymbol{v}_i x_i \,, \tag{1.1}$$

where $\boldsymbol{x} \in \mathbb{R}^N$ contains mostly zero coefficients. Specifically, the sparsity level S of $\boldsymbol{x}$ is defined to be the number of non-zero coefficients in $\boldsymbol{x}$, i.e. the l_0 norm. We call $\boldsymbol{x}$ an S-sparse signal if $\|\boldsymbol{x}\|_0 \leq S$.

In the context of this dissertation, the linear sampling/measurement of the signal $\boldsymbol{s}$ is accomplished by left-multiplying $\boldsymbol{s}$ with a $M \times N$ sampling matrix $\boldsymbol{U}$. Let $\boldsymbol{w} \in \mathbb{R}^M$ denote the noise, we have the following measurement vector $\boldsymbol{y} \in \mathbb{R}^M$:

$$\boldsymbol{z} = \boldsymbol{U}\boldsymbol{s} = \boldsymbol{U}\boldsymbol{V}\boldsymbol{x} = \boldsymbol{A}\boldsymbol{x} \tag{1.2}$$

$$\boldsymbol{y} = \boldsymbol{z} + \boldsymbol{w} \,, \tag{1.3}$$

where $\boldsymbol{z} \in \mathbb{R}^M$ is the noiseless measurement vector; $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{V}$ is commonly known as the $M \times N$ sensing matrix, $\boldsymbol{A}$ is usually chosen by us and already known.

In compressive sensing, we try to recover the sparse signal $\boldsymbol{x}$ from limited measurements $\boldsymbol{y}$ where $M \ll N$. Here, (1.2)-(1.3) then describe a underdetermined linear system. Since the sensing matrix $\boldsymbol{A}$ contains more columns than rows, there are more than one solutions that would satisfy $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \leq \epsilon$, where $\epsilon \geq 0$ is the upper bound on the noise contribution. This makes the recovery of $\boldsymbol{x}$ an ill-posed problem.

1.1.1 Sparsity-regularization approach

Since the signal of interest itself is sparse, one naive, or perhaps the most straightforward, way to decide on a solution is to follow the well known Occam's razor and choose the sparsest (simplest) one:

$$P_0(\boldsymbol{x}) : \qquad \min_{\boldsymbol{x}} \quad \|\boldsymbol{x}\|_0 \qquad \text{subject to} \quad \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \leq \epsilon \,. \tag{1.4}$$

$P_0(\boldsymbol{x})$ is a nonconvex NP-hard problem whose solution requires an intractable combinatorial search [13]. In practice, two alternative approaches are usually employed to solve $P_0(\boldsymbol{x})$:

- Greedy search under the constrain $\|\boldsymbol{x}\|_0 \leq S$.
- Relaxation of the l_0 norm $\|\boldsymbol{x}\|_0$.

The greedy search approach leads to various matching pursuit methods [14–17]. While the relaxation approach leads to minimizing different objective functions that promote sparsity in the solution [18–21].

Here we focus on studying the "relaxation" approach that tries to solve the following unconstrained recovery problem:

$$P_g(\boldsymbol{x}) : \qquad \min_{\boldsymbol{x}} \quad \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda g(\boldsymbol{x}) \,, \tag{1.5}$$

where $\lambda > 0$ is the parameter that balances the trade-off between the data fidelity term $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2$ and the sparsity-regularization objective $g(\boldsymbol{x})$. A proper $g(\boldsymbol{x})$ is crucial to the success of the sparse signal recovery task: it should favor sparse solutions and make sure the problem $P_g(\boldsymbol{x})$ can be solved efficiently in the mean time.

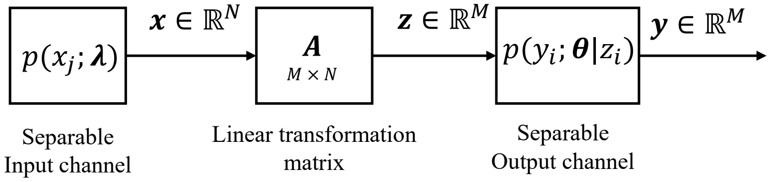

$p(x_j;\boldsymbol{\lambda})$
$\boldsymbol{x} \in \mathbb{R}^N$
$\boldsymbol{A}$
$M \times N$
$\boldsymbol{z} \in \mathbb{R}^M$
$p(y_i;\boldsymbol{\theta}|z_i)$
$\boldsymbol{y} \in \mathbb{R}^M$
Separable Input channel
Linear transformation matrix
Separable Output channel

Figure 1.2: A probabilistic view of the sparse signal recovery task [22]: The signal $\boldsymbol{x}$ is estimated given the output vector $\boldsymbol{y}$, the channel transition probability functions $p(x_j;\boldsymbol{\lambda})$, $p(y_i;\boldsymbol{\theta}|z_i)$ and the transformation matrix $\boldsymbol{A}$. $\{\boldsymbol{\lambda},\boldsymbol{\theta}\}$ denote the parameters of the probability models and are usually unknown.

1.1.2 Probabilistic approach

If we assume each entry x_j of the signal $\boldsymbol{x}$ is independently and identically distributed according to some distribution $p(x_j;\boldsymbol{\lambda})$. The under-determined linear system given in (1.2,1.3) can also be described using the probability model shown in Fig. 1.2. For the sparse signal recovery task, $p(x_j;\boldsymbol{\lambda})$ should reflect the "sparse" attribute of $\boldsymbol{x}$. Popular choices include Laplace distribution, Bernoulli-Gaussian mixture distribution, etc.

Under the Bayesian setting, we can use the approximate message passing (AMP) algorithm to compute either the maximum a posterior (MAP) or minimum mean square error (MMSE) estimate of the signal $\boldsymbol{x}$. Specifically, AMP uses approximated loopy belief propagation [22–24] to perform probabilistic inferences on the signal. The parameters of the probability model $\{\boldsymbol{\lambda},\boldsymbol{\theta}\}$ are usually unknown and need to be decided for the AMP algorithms. [25, 26] use the Expectation Maximization (EM) algorithm [27] to estimate the parameters. However, it is not widely applicable due to its high complexity, and thus can only handle simple distributions. Since real data come in all shapes and sizes, it is thus important to explore new approaches that can be easily adapted to a wide variety of distributions.

1.2 Dissertation goal and contributions

In this dissertation, we aim to tackle the sparse recovery problem using the aforementioned sparsity-regularization and probabilistic approaches, design efficient algorithms that scale to large datasets with ease, and provide detailed theoretical analyses of the proposed approaches. Specifically, the main contributions of the dissertation include:

- We propose the nonconvex "entropy function" of $\boldsymbol{x}$ as a new sparsity-regularization objective, and prove that minimizing it can produce sparser solutions than the popular convex l_1 norm-minimization approach.
- We design efficient iterative algorithm to recover the signal $\boldsymbol{x}$ by minimizing the proposed entropy function, and show that it outperforms both the convex l_1 norm-minimization and the nonconvex l_p norm-minimization approaches on simulated and real data.
- We adapt the proposed entropy function minimization approach to other compressive sensing applications, such as the low-rank matrix completion, robust Principle Component Analysis, etc.
- We propose a generalized approximate message passing framework with built-in parameter estimation (PE-GAMP), and provide detailed approximate loopy belief propagation procedures to perform joint probabilistic inferences on the signal and the parameters,
- We conduct the state evolution analysis on the proposed PE-GAMP, and prove its empirical convergence behavior in the large system limit under certain conditions.

- We provide detail formulations of the PE-GAMP for several input and out channel distributions, and show that it not only has a wider applicability but also is more robust than EM based parameter estimation approach when the measurements are limited.

1.3 Outline of the dissertation

This dissertation consists four parts. Part I introduces compressive sensing as a solution to address the challenges raised by the data explosion nowadays: Chapter 1 briefly explains the two approaches we employ in this dissertation to tackle the sparse signal recovery problem. Chapter 2 gives a survey of related work and provides necessary background to set up the work in subsequent chapters.

Part II explores the sparsity-regularization approach: Chapter 3 proposes the entropy functions $h_p(\boldsymbol{x})$, $h_{p,\alpha}(\boldsymbol{x})$ constructed from the signal $\boldsymbol{x}$ and investigates its property. Chapter 4 describes the iterative algorithm to recover the sparse signal by minimizing the entropy functions, and compares it with other sparsity-regularization objectives on simulated and real datasets. Chapter 5 applies the entropy function minimization approach to the robust principal component analysis.

Part III studies the probabilistic approach: Chapter 6 presents the proposed PE-GAMP framework and its approximate loopy belief propagation realizations. Chapter 7 conducts the state evolution analysis of the PE-GAMP and proves its empirical convergence behavior in the large system limit under certain conditions. Chapter 8 provides parameter estimation formulations of several input and output channels, followed by experiments comparing the PE-GAMP with the EM based parameter estimation approach.

Part IV concludes this dissertation with discussions and future work.

Chapter 2

The quest for sparsity

The success of signal recovery from the under-determined linear system in (1.2,1.3) relies on 1) the sparse attribute of the input signal and 2) the incoherence property of the sensing matrix. In order to enforce sparse prior information on the recovered solution, the sparsity-regularization approach directly searches for the sparsest solution that minimizes the objective function, while the probabilistic approach imposes a sparse prior distribution on the input signal. In this chapter, we explain the reasons behind the quest for sparsity in compressive sensing and pay tribute to the pioneering works in this field.

2.1 Restricted isometry property

Restricted isometry property [28, 29] is the key to understanding why sparsity matters, it is defined with respect to the sensing matrix $\boldsymbol{A}$.

Definition 1. *A matrix $\boldsymbol{A}$ satisfies the restricted isometry property (RIP) of order S if there exists a smallest bound $0 < \delta_S < 1$ such that*

$$(1-\delta_S)\|\boldsymbol{x}\|_2^2 \leq \|\boldsymbol{A}\boldsymbol{x}\|_2^2 \leq (1+\delta_S)\|\boldsymbol{x}\|_2^2, \tag{2.1}$$

holds for all S-sparse signal $\boldsymbol{x}$: $\|\boldsymbol{x}\|_0 \leq S$.

[29] uses the l_1 norm of $\boldsymbol{x}$ as the relaxed sparsity regularizer and solves the following problem to get the recovered signal $\hat{\boldsymbol{x}}_{l_1}$:

$$P_1(\boldsymbol{x}) : \qquad \min_{\boldsymbol{x}} \quad \|\boldsymbol{x}\|_1 \qquad \text{subject to} \quad \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \leq \epsilon \,. \tag{2.2}$$

2.1.1 Noiseless recovery

The l_1 and l_2 error of the recovered signal $\hat{\boldsymbol{x}}_{l_1}$ is bounded according to the following theorem by [29]:

Theorem 1. *(Noiseless recovery). If $\boldsymbol{A}$ satisfies the RIP of order $2S$ with $\delta_{2S} < \sqrt{2} - 1$, then $\hat{\boldsymbol{x}}_{l_1}$ obeys:*

$$\|\hat{\boldsymbol{x}}_{l_1} - \boldsymbol{x}\|_1 \leq C_0 \|\boldsymbol{x} - \tilde{\boldsymbol{x}}_S\|_1 \tag{2.3}$$

$$\|\hat{\boldsymbol{x}}_{l_1} - \boldsymbol{x}\|_2 \leq C_0 S^{-\frac{1}{2}} \|\boldsymbol{x} - \tilde{\boldsymbol{x}}_S\|_1 \,. \tag{2.4}$$

for some small constant C_0 given in [29]. In particular, if $\boldsymbol{x}$ is S-sparse, the recovery is exact.

Here, $\tilde{\boldsymbol{x}}_S$ denote the approximation of $\boldsymbol{x}$ by keeping only the S largest entries in magnitude. If $\delta_{2S} < 1$, the problem $P_0(\boldsymbol{x})$ in (1.4) has a unique S-sparse solution. We can then see that the solutions of $P_1(\boldsymbol{x})$ and $P_0(\boldsymbol{x})$ are actually the same when $\delta_{2S} < \sqrt{2} - 1$. This means the sparsest solution is indeed the true solution when $\boldsymbol{x}$ is S-sparse and $\boldsymbol{A}$ satisfies the corresponding RIP.

2.1.2 Noisy recovery

The l_2 error of $\hat{\boldsymbol{x}}_{l_1}$ is also bounded as stated in the following theorem [29]:

Theorem 2. *(Noisy recovery). If $\boldsymbol{A}$ satisfies the RIP of order $2S$ with $\delta_{2S} < \sqrt{2} - 1$,* *then* $\hat{\boldsymbol{x}}_{l_1}$ *obeys:*

$$\|\hat{\boldsymbol{x}}_{l_1} - \boldsymbol{x}\|_2 \leq C_0 S^{-\frac{1}{2}} \|\boldsymbol{x} - \tilde{\boldsymbol{x}}_S\|_1 + C_1 \epsilon \tag{2.5}$$

, for some small constants C_0, C_1 *given in [29].*

In this case, we can still recovery the signal robustly if it is S-sparse and $\boldsymbol{A}$ satisfies RIP with $\delta_{2S} < \sqrt{2} - 1$.

2.2 Incoherent matrix

Verifying the RIP property of the sensing matrix $\boldsymbol{A}$ is NP-hard in general [30]. In practice we can check the coherence of $\boldsymbol{A}$ instead [31, 32].

Definition 2. *The coherence* $\mu(\boldsymbol{A})$ *of a matrix* $\boldsymbol{A} \in \mathbb{R}^{M \times N}$*,* $M < N$ *is the largest absolute inner product between any two normalized columns* $\boldsymbol{a}_i$ *and* $\boldsymbol{a}_j$ *of* $\boldsymbol{A}$*.*

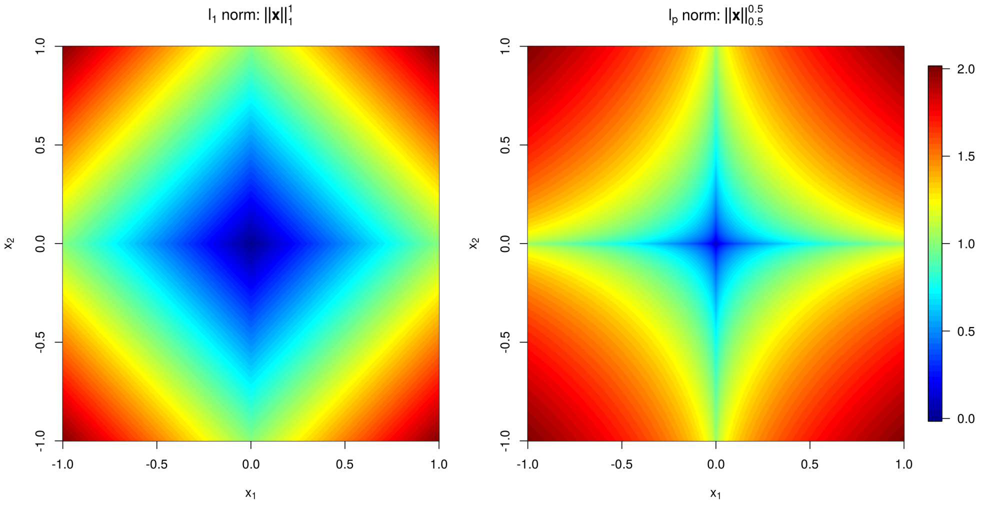
$$\mu(\boldsymbol{A}) = \max_{1 \leq i \neq j \leq N} | < \boldsymbol{a}_i, \boldsymbol{a}_j > | \,. \tag{2.6}$$

In fact, a matrix with low coherence is what makes RIP hold, as is given by the following theorem:

Theorem 3. *If* $\boldsymbol{A}$ *has coherence* $\mu(\boldsymbol{A})$*, then* $\boldsymbol{A}$ *satisfies RIP of order* S *with* $\delta_S = S \cdot \mu(\boldsymbol{A}), \forall S < \frac{1}{\mu(\boldsymbol{A})}$

This does make sense intuitively. Since we want to get as much information about the sparse signal $\boldsymbol{x}$ as possible out of the measurement $\boldsymbol{y}$, we would prefer a matrix $\boldsymbol{A}$ whose columns are diverse and has low correlation with one another, i.e. an *incoherent* matrix.

If $M \geq N$, a matrix $\boldsymbol{A}$ with orthonormal columns has the lowest coherence $\mu(\boldsymbol{A}) = 0$. In compressive sensing, however, M is much smaller than N: $M \ll N$,



(a) (b)

Figure 2.1: The 2D level plots: (a) the convex $\|\boldsymbol{x}\|_1$; (b) the nonconvex $\|\boldsymbol{x}\|_{0.5}^{0.5}$

and various approaches has been proposed to design efficient sensing matrix with low coherence [33–36]. In general, the presence of randomness in $\boldsymbol{A}$ is essential to constructing a working sensing matrix. As we shall see later, the state evolution analysis of the approximate message passing algorithms in the probabilistic approaches relies on the assumption that entries of $\boldsymbol{A}$ are i.i.d. according to $N(0, \frac{1}{M})$.

2.3 Relaxation of the l_0 norm

Various sparsity regularizers have been proposed as the relaxations of the l_0 norm [18–21]. Most popular among them are the convex l_1 norm and the nonconvex l_p norm to the p-th power:

- l_1 norm: $\|\boldsymbol{x}\|_1 = \sum_i |x_i|$.
- l_p norm to the p-th power: $\|\boldsymbol{x}\|_p^p = \sum_i |x_i|^p$, $0 < p < 1$.

The 2D level plots of the above two objectives are shown in Fig. 2.1. There exist numerous efficient iterative algorithms to find the minimizing solutions to the l_1 norm-minimization problem in (2.2) [37–40]. The minimization of the $\|\boldsymbol{x}\|_p^p$ can be done by following the reweighted-l_1 strategy [19–21].

2.4 Input and out channel distributions

The generative probabilistic model in Fig. 1.2 assumes the signal $\boldsymbol{x}$ and the measurement $\boldsymbol{y}$ both have i.i.d. entries: $x_j \sim p(x_j;\boldsymbol{\theta})$; $y_i \sim p(y_i;\boldsymbol{\theta}|z_i)$. The sparse attribute of the signal $\boldsymbol{x}$ is reflected by the prior distribution $p(x_j;\boldsymbol{\theta})$. Specifically, we will consider the following input channel distributions in this dissertation:

1. **Bernoulli-Gaussian mixture (BGm) Input Channel:** The sparse signal $\boldsymbol{x} \in \mathbb{R}^N$ can be modeled as a mixture of Bernoulli and Gaussian mixture distributions:

$$\begin{aligned} p(x_j;\boldsymbol{\lambda}) = {} & (1-\lambda_1)\delta(x_j) \\ & + \lambda_1 \sum_{c=1}^{C} \lambda_{c+1} \cdot \mathcal{N}(x_j;\lambda_{c+2},\lambda_{c+3}) \,, \end{aligned} \tag{2.7}$$

   where $x_j \in \mathbb{R}$; $\delta(\cdot)$ is Dirac delta function; $\lambda_1 \in [0,1]$ is the sparsity rate; for the c-th Gaussian mixture, $\lambda_{c+1} \in [0,1]$ is the mixture weight, $\lambda_{c+2} \in \mathbb{R}$ is the nonzero coefficient mean and $\lambda_{c+3} \in (0,\infty)$ is the nonzero coefficient variance; all the mixture weights should sum to 1: $\sum_{c=1}^{C} \lambda_{3c-1} = 1$.

2. **Bernoulli-Exponential mixture (BEm) Input Channel:** Nonnegative sparse signal $\boldsymbol{x} \in \mathbb{R}^N$ can be modeled as a mixture of Bernoulli and Exponential mixture distributions:

$$\begin{aligned} p(x_j;\boldsymbol{\lambda}) = {} & (1-\lambda_1)\delta(x_j) \\ & + \lambda_1 \sum_{c=1}^{C} \lambda_{c+1} \cdot \lambda_{c+2} \exp\left(-\lambda_{c+2} x_j\right) \,, \end{aligned} \tag{2.8}$$

where $x_j \in [0, \infty)$; $\lambda_1 \in [0, 1]$ is the sparsity rate; for the c-th Exponential mixture, $\lambda_{c+1} \in [0, 1]$ is the mixture weight and $\lambda_{c+2} \in (0, \infty)$; all the mixture weights should sum to 1: $\sum_{c=1}^{C} \lambda_{2c} = 1$.

3. **Laplace Input Channel:** The sparse signal $\boldsymbol{x} \in \mathbb{R}^N$ follows the following Laplace distribution:

$$p(x_j; \boldsymbol{\lambda}) = \frac{\lambda_1}{2} \exp\left(-\lambda_1 |x_j|\right) , \tag{2.9}$$

   where $x_j \in \mathbb{R}$; $\lambda_1 \in (0, \infty)$.

The probabilistic approach is closely related to the sparsity-regularization approaches. For instance, the solution of $P_1(\boldsymbol{x})$ in (2.2) is essentially the MAP estimation of $\boldsymbol{x}$ under the Laplace input channel and the following additive white Gaussian noise (AWGN) output channel:

- **Additive White Gaussian Noise (AWGN) Output Channel:** The noise $\boldsymbol{w} \in \mathbb{R}^M$ is assumed to be white Gaussian noise:

$$p(w_i; \boldsymbol{\theta}) = \mathcal{N}(w_i; 0, \theta_1) , \tag{2.10}$$

  where $w_i \in \mathbb{R}$ is the noise; $\theta_1 \in (0, \infty)$ is its variance.

The posterior distribution $p(x_j|\boldsymbol{y})$ can then obtained using the message passing algorithms [22–24]. We can then recover the sparse signal $\boldsymbol{x}$ as follows:

- **MMSE estimate:** $\hat{x}_j = \int x_j p(x_j; \boldsymbol{\lambda}, \boldsymbol{\theta}|\boldsymbol{y})\, dx_j$
- **MAP estimate:** $\hat{x}_j = \max_{x_j} p(x_j; \boldsymbol{\lambda}, \boldsymbol{\theta}|\boldsymbol{y})$

The parameters $\{\boldsymbol{\lambda}, \boldsymbol{\theta}\}$ are usually unknown in practice, and need to be estimated. EM based parameter estimation has been proposed in [25, 26] for the BGm input channel and AWGN output channel. However, its high complexity greatly limits its applicability for complex distributions.

Part II

Sparsity-regularization approach

Chapter 3

Sparsity-regularization entropy function

In this chapter we construct the nonconvex entropy function from the sparse signal $\boldsymbol{x}$ and use it as the sparsity regularizer. Specifically, two types of entropy functions are proposed: the Shannon entropy function $h_p(\boldsymbol{x})$ and the Rényi entropy function $h_{p,\alpha}(\boldsymbol{x})$. We show that minimizing them does favor sparse solutions and could improve upon the solutions from the l_1-norm minimization approach.

3.1 Introduction to entropy

We first introduce the entropy concepts in information theory [41, 42]. Both the Shannon entropy and Rényi entropy are defined with respect to the probability distribution $p(\mathcal{V})$ of some random variable $\mathcal{V}$. Here we give the following definitions in terms of discrete probability distribution[1]:

---

[1] For continuous distributions, the sum $\sum$ in (3.1,3.2) should be replaced with integration $\int$.

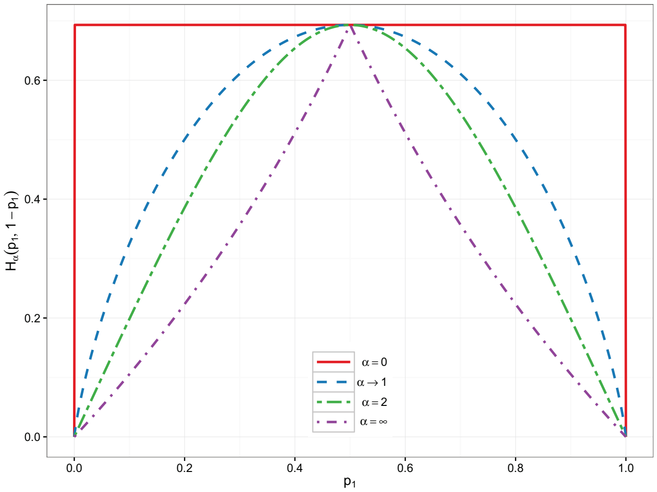


Figure 3.1: Rényi entropy defined on $\{p_1, 1 - p_1\}$ with different choices of α.

- Shannon entropy[2]:

$$\mathcal{H}(\mathcal{V}) = -\sum_{i=1}^{|\mathcal{V}|} p(v_i) \log p(v_i) \,. \tag{3.1}$$

  $\mathcal{H}(\mathcal{V})$ is strictly concave with respect to the probability distribution $\mathcal{P}_{\mathcal{V}} = \{p(v_1), \cdots, p(v_{|\mathcal{V}|})\}$.

- Rényi entropy:

$$\mathcal{H}_\alpha(\mathcal{V}) = \frac{1}{1-\alpha} \log \left(\sum_{i=1}^{|\mathcal{V}|} p(v_i)^\alpha \right) , \tag{3.2}$$

  where $\alpha \geq 0$ and $\alpha \neq 1$. When $\alpha \in (0, 1)$, $\mathcal{H}_\alpha(\mathcal{V})$ is strictly concave with respect to $\mathcal{P}_{\mathcal{V}}$ [43]; when $\alpha \in (1, \infty)$, $\mathcal{H}_\alpha(\mathcal{V})$ is strictly Schur concave with respect to $\mathcal{P}_{\mathcal{V}}$ [44].

We should make it clear that Shannon entropy $\mathcal{H}(\mathcal{V})$ is *not* a special case of the Rényi entropy, but the limiting value of the Rényi entropy $\mathcal{H}_\alpha(\mathcal{V})$ as $\alpha \to 1$ [45]. Hence we need to discuss them respectively in this dissertation. Take the the simple probability distribution $\mathcal{P} = \{p_1, 1 - p_1\}$ for example, the Shannon

---

[2] The "log" in this dissertation is by default natural logarithm, i.e. base e

entropy and the Rényi entropy defined on $\{p_1, 1 - p_1\}$ with different choices of α are shown in Fig. 3.1 adapted from [46].

3.2 Entropy function of the sparse signal

3.2.1 Shannon entropy function

Entropy measures uncertainty about the random variable $\mathcal{V}$. The lower the entropy is, the more predictable the variable $\mathcal{V}$ is, which corresponds to a more skewed distribution $p(\mathcal{V})$. The idea of a skewed distribution could translate naturally to the idea of a sparse distribution based on the fact that only a few probability values of $\mathcal{P}$ are significant. In other words, the entropy is also an indication how skewed/sparse the distribution $\mathcal{P}$ is. This observation motivates us to investigate the possibility of using it as a new sparsity regularizer in the dissertation.

Specifically, the Shannon entropy function $h_p(\boldsymbol{x})$ of the sparse signal $\boldsymbol{x}$ is defined as follows:

$$h_p(\boldsymbol{x}) = -\sum_{i=1}^{N} \frac{|x_i|^p}{\|\boldsymbol{x}\|_p^p} \log \frac{|x_i|^p}{\|\boldsymbol{x}\|_p^p}, \tag{3.3}$$

where $p > 0$. Here we essentially construct the following discrete probability distribution $\mathcal{P}$ out of $\boldsymbol{x}$ and then compute the Shannon entropy defined on $\mathcal{P}$:

$$\mathcal{P} := \left\{ \frac{|x_1|^p}{\|\boldsymbol{x}\|_p^p}, \frac{|x_2|^p}{\|\boldsymbol{x}\|_p^p}, \cdots, \frac{|x_N|^p}{\|\boldsymbol{x}\|_p^p} \right\}. \tag{3.4}$$

$h_p(\boldsymbol{x})$ is the nonconvex "Shannon entropy function" of $\boldsymbol{x}$, it should *not* be confused with the "Shannon entropy" of $\boldsymbol{x}$ in (3.1): $\mathcal{H}(\boldsymbol{x}) = -\int_{\boldsymbol{x}} p(\boldsymbol{x}) \log p(\boldsymbol{x})\, d\boldsymbol{x}$. The 2D level plot of $h_p(\boldsymbol{x})$ with $p = 0.5$ is shown in Fig. 3.2. We can see that the local minimums occur on the two axises in this case.

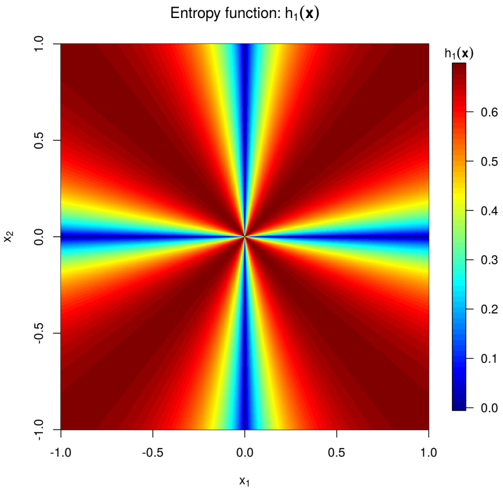


Figure 3.2: Entropy function $h_p(\boldsymbol{x})$ with $p = 1$ in the 2-dimensional space.

The sparse signal recovery problem in (1.5) based on the nonconvex Shannon entropy function (SEF) minimization then becomes:

$$P_{h_p}(\boldsymbol{x}): \qquad \min_{\boldsymbol{x}} \quad \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda h_p(\boldsymbol{x}), \tag{3.5}$$

where $\lambda > 0$.

3.2.2 Rényi entropy function

Using the discrete probability distribution $\mathcal{P}$ in (3.4), we define the Rényi entropy function of the sparse signal $\boldsymbol{x}$ as follows:

$$h_{p,\alpha}(\boldsymbol{x}) = \frac{1}{1-\alpha} \log\left(\sum_{i=1}^{|\mathcal{V}|}\left(\frac{|x_i|^p}{\|\boldsymbol{x}\|_p^p}\right)^{\alpha}\right), \tag{3.6}$$

where $p > 0$, $\alpha \geq 0$ and $\alpha \neq 1$. Again, this should not be confused with the Rényi entropy of $\boldsymbol{x}$ in (3.2). The sparse signal recovery problem in (1.5) based on the nonconvex Rényi entropy function (REF) minimization then becomes:

$$P_{h_{p,\alpha}}(\boldsymbol{x}): \qquad \min_{\boldsymbol{x}} \quad \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda h_{p,\alpha}(\boldsymbol{x}), \tag{3.7}$$

where $\lambda > 0$.

❖ **Discussion:** In previous sections we proposed the entropy functions $h_p(\boldsymbol{x})$, $h_{p,\alpha}(\boldsymbol{x})$ constructed from $\boldsymbol{x}$ as sparsity regularizers. However, they are not the entropy of the sparse signal $\boldsymbol{x}$. Its computation requires knowledge of the distribution $p(\boldsymbol{x})$ of $\boldsymbol{x}$. In practice, we usually assume the entries of $\boldsymbol{x}$ are i.i.d distributed according to some parameterized distribution $p(x|\boldsymbol{\lambda})$. The estimated parameters $\hat{\boldsymbol{\lambda}}$ can be obtained following the probabilistic approach presented in Part III. The entropy of the signal $\mathcal{H}(x), \mathcal{H}_\alpha(x)$ can then be computed as in (3.1,3.2) using $p(\boldsymbol{x}|\hat{\boldsymbol{\lambda}})$.

3.3 Sparsity promotion analysis

We next show that $h_p(\boldsymbol{x})$ and $h_{p,\alpha}(\boldsymbol{x})$ are sparsity regularizers in the following sense: minimizing them in an orthant $\mathbb{O}$ of the Euclidean space $\mathbb{R}^N$ leads us to solutions on the boundary of said orthant, i.e. sparser solutions.

❑ **Noiseless recovery:** In this case we are minimizing $h_p(\boldsymbol{x})$ or $h_{p,\alpha}(\boldsymbol{x})$ subject to the constrain $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$. We first show that there is a one to one mapping in each orthant between $\boldsymbol{x} = [x_1, \cdots, x_N]^\mathrm{T}$ and $\ddot{\boldsymbol{x}} = [\ddot{x}_1, \cdots, \ddot{x}_N]^\mathrm{T}$, where $\ddot{x}_i = \mathrm{sign}(x_i) \cdot \frac{|x_i|^p}{\|\boldsymbol{x}\|_p^p}$. This will be done in two steps: Lemma 1 and Lemma 2.

Lemma 1. *If $\boldsymbol{x}$ is the solution to $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$, $\boldsymbol{y} \neq 0$, then there is a one to one mapping in each orthant between $\boldsymbol{x}$ and $\tilde{\boldsymbol{x}} = \frac{\boldsymbol{x}}{\|\boldsymbol{x}\|_p}$.*

Proof. We just need to prove $\boldsymbol{x} \longleftrightarrow \tilde{\boldsymbol{x}}$:

- It is easy to verify that $\boldsymbol{x} \rightarrow \tilde{\boldsymbol{x}}$.

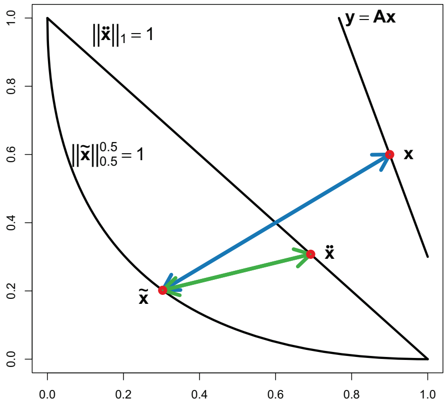


Figure 3.3: The one-to-one mapping: $x \longleftrightarrow \tilde{x} \longleftrightarrow \ddot{x}$ when $p = 0.5$.

- Suppose there are two solutions of $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$: $\boldsymbol{x}_{(1)}, \boldsymbol{x}_{(2)}$ in the same orthant, and they are both mapped to $\tilde{x}$. We then have:

$$\frac{\boldsymbol{x}_{(1)}}{\|\boldsymbol{x}_{(1)}\|_p} = \tilde{x} = \frac{\boldsymbol{x}_{(2)}}{\|\boldsymbol{x}_{(2)}\|_p} \tag{3.8}$$

$$\frac{\boldsymbol{y}}{\|\boldsymbol{x}_{(1)}\|_p} = \frac{\boldsymbol{A}\boldsymbol{x}_{(1)}}{\|\boldsymbol{x}_{(1)}\|_p} = \boldsymbol{A}\tilde{x} = \frac{\boldsymbol{A}\boldsymbol{x}_{(2)}}{\|\boldsymbol{x}_{(2)}\|_p} = \frac{\boldsymbol{y}}{\|\boldsymbol{x}_{(2)}\|_p}, \tag{3.9}$$

  which tells us $\frac{\boldsymbol{y}}{\|\boldsymbol{x}_{(1)}\|_p} = \frac{\boldsymbol{y}}{\|\boldsymbol{x}_{(2)}\|_p}$. Since $\boldsymbol{y} \neq 0$, we have $\|\boldsymbol{x}_{(1)}\|_p = \|\boldsymbol{x}_{(2)}\|_p$. Using (3.8), we get $\boldsymbol{x}_{(1)} = \boldsymbol{x}_{(2)}$. Hence $\boldsymbol{x} \leftarrow \tilde{x}$.

□

Lemma 2. *There is a one to one mapping in each orthant between $\tilde{x}$ and $\ddot{x}$*

Proof. We just need to prove $\tilde{x} \longleftrightarrow \ddot{x}$:

- We can rewrite $\ddot{x}$ in terms of $\tilde{x}$: $\ddot{x} = \text{sign}(\tilde{x}) \cdot |\tilde{x}|^p$. Hence $\tilde{x} \rightarrow \ddot{x}$.

- Suppose there are two points $\tilde{x}_{(1)}, \tilde{x}_{(2)}$ in the same orthant mapped to the same $\ddot{x}$. We then have:

$$\operatorname{sign}(\tilde{x}_{(1)}) \cdot |\tilde{x}_{(1)}|^p = \ddot{x} = \operatorname{sign}(\tilde{x}_{(2)}) \cdot |\tilde{x}_{(2)}|^p , \tag{3.10}$$

which tells us $|\tilde{x}_{(1)}| = |\tilde{x}_{(2)}|$. Since $\operatorname{sign}(\tilde{x}_{(1)}) = \operatorname{sign}(\tilde{x}_{(2)})$, we get $\tilde{x}_{(1)} = \tilde{x}_{(2)}$. Hence $\tilde{x} \leftarrow \ddot{x}$.

□

Combining Lemma 1 and Lemma 2, we have $\boldsymbol{x} \longleftrightarrow \ddot{x}$, as is shown in Fig. 3.3. Let $\mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots\}$ be the solutions of $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ in one of the orthants $\mathbb{O}$. Specifically, $\mathcal{X} = \mathcal{X}_1 \cup \mathcal{X}_2$ and $\mathcal{X}_1 \cap \mathcal{X}_2 = \emptyset$, where $\mathcal{X}_1$ contains solutions on the boundary of the orthant $\mathbb{O}$ and $\mathcal{X}_2$ contains the rest solutions that are not on the boundary. The solution $\boldsymbol{x}$ is then mapped to $\ddot{x}$ one by one, producing the corresponding mapped sets $\ddot{\mathcal{X}}_1, \ddot{\mathcal{X}}_2$. We can verify that the solutions in $\mathcal{X}_1$ are *sparser* than those in $\mathcal{X}_2$, and we have the following Lemma 3:

Lemma 3. *For every solution $\boldsymbol{x} \in \mathcal{X}_2$, there is a solution $\boldsymbol{x}^* \in \mathcal{X}_1$ on the boundary of the orthant $\mathbb{O}$ such that $h_p(\boldsymbol{x}^*) < h_p(\boldsymbol{x})$ and $h_{p,\alpha}(\boldsymbol{x}^*) < h_{p,\alpha}(\boldsymbol{x})$.*

Proof. By definition we have:

$$h_p(\boldsymbol{x}) = g(\ddot{x}) = -\textstyle\sum_{i=1}^{N} |\ddot{x}_i| \log |\ddot{x}_i| \tag{3.11}$$

$$h_{p,\alpha}(\boldsymbol{x}) = g_\alpha(\ddot{x}) = \frac{1}{1-\alpha} \log \left(\textstyle\sum_{i=1}^{N} |\ddot{x}_i|^\alpha \right) . \tag{3.12}$$

We first study the local minimums on the plane $\|\ddot{x}\|_1 = 1$. $g(\ddot{x})$ is *strictly concave* with respect to $\ddot{x}$, and the local minimums of $g(\ddot{x})$ are on the boundary of the orthant $\mathbb{O}$. Hence for every $\ddot{x} \in \ddot{\mathcal{X}}_2$, there is a $\ddot{x}^* \in \ddot{\mathcal{X}}_1$ such that $g(\ddot{x}^*) < g(\ddot{x})$.

When $\alpha \in (0, 1)$, $g_\alpha(\ddot{x})$ is *strictly concave* with respect to $\ddot{x}$, the local minimums of $g(\ddot{x})$ are on the boundary of the orthant $\mathbb{O}$. When $\alpha \in (1, \infty)$, $g_\alpha(\ddot{x})$ is *strictly*

Schur concave, since the boundary of the orthant $\mathbb{O}$ majorizes the $\ddot{\boldsymbol{x}}$ inside $\mathbb{O}$, the local minimums of $g(\ddot{\boldsymbol{x}})$ are also on the boundary of $\mathbb{O}$. Hence for every $\ddot{\boldsymbol{x}} \in \ddot{\mathcal{X}}_2$, there also exists a $\ddot{\boldsymbol{x}}^* \in \ddot{\mathcal{X}}_1$ such that $g_\alpha(\ddot{\boldsymbol{x}}^*) < g_\alpha(\ddot{\boldsymbol{x}})$ for $\alpha \in (0,1) \cup (1,\infty)$.

There is a one to one mapping in $\mathbb{O}$ between $\boldsymbol{x}$ and $\ddot{\boldsymbol{x}}$: $\boldsymbol{x} \longleftrightarrow \ddot{\boldsymbol{x}}$. Since $h_p(\boldsymbol{x}) = g(\ddot{\boldsymbol{x}})$ and $h_{p,\alpha}(\boldsymbol{x}) = g_\alpha(\ddot{\boldsymbol{x}})$, for every $\boldsymbol{x} \in \mathcal{X}_2$, there is a $\boldsymbol{x}^* \in \mathcal{X}_1$ such that $h_p(\boldsymbol{x}^*) < h_p(\boldsymbol{x})$ and $h_{p,\alpha}(\boldsymbol{x}^*) < h_{p,\alpha}(\boldsymbol{x})$. □

From Lemma 3 we can see that minimizing $h_p(\boldsymbol{x})$ or $h_{p,\alpha}(\boldsymbol{x})$ in the orthant $\mathbb{O}$ will lead us to the sparser solutions in $\mathcal{X}_1$.

❑ **Noisy recovery:** We can show similarly that minimizing $h_p(\boldsymbol{x})$ or $h_{p,\alpha}(\boldsymbol{x})$ subject to the constrain $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \le \epsilon$ in an orthant $\mathbb{O}$ of the Euclidean space $\in \mathbb{R}^N$ also produces sparse solutions. First, we have the following Lemma 4:

Lemma 4. *Let* $\mathcal{X}^\epsilon = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots\}$ *are the nonzero solutions satisfying the constrain* $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \le \epsilon, \boldsymbol{y} \neq 0$ *such that:* $\forall \boldsymbol{x}_i \neq \boldsymbol{x}_j$, $\boldsymbol{x}_i = \tau \boldsymbol{x}_j$ *for some* $\tau > 0$. *Pick any* $\boldsymbol{x}_i \in \mathcal{X}^\epsilon$, *there is a one to one mapping in each orthant between the set* $\mathcal{X}^\epsilon$ *and* $\tilde{\boldsymbol{x}}_i = \frac{\boldsymbol{x}_i}{\|\boldsymbol{x}_i\|_p}$.

Proof. We need to prove $\mathcal{X}_\epsilon \longleftrightarrow \tilde{\boldsymbol{x}}_i$

- $\forall \boldsymbol{x}_j \in \mathcal{X}^\epsilon \backslash \boldsymbol{x}_i$, we have $\tilde{\boldsymbol{x}}_i = \frac{\boldsymbol{x}_i}{\|\boldsymbol{x}_i\|_p} = \frac{\tau \boldsymbol{x}_j}{\|\tau \boldsymbol{x}_j\|_p} = \frac{\boldsymbol{x}_j}{\|\boldsymbol{x}_j\|_p} = \tilde{\boldsymbol{x}}_j$. It is easy to verify that $\boldsymbol{x}_j \to \tilde{\boldsymbol{x}}_j = \tilde{\boldsymbol{x}}_i$. Hence $\mathcal{X}^\epsilon \to \tilde{\boldsymbol{x}}_i$.

- Suppose that there are two sets $\mathcal{X}^\epsilon_{(1)}, \mathcal{X}^\epsilon_{(2)}$ in the same orthant being mapped to the same $\tilde{\boldsymbol{x}}_i$. Let $\boldsymbol{x}_1 \in \mathcal{X}^\epsilon_{(1)}$ and $\boldsymbol{x}_2 \in \mathcal{X}^\epsilon_{(2)}$, we have:

$$\frac{\boldsymbol{x}_1}{\|\boldsymbol{x}_1\|_p} = \tilde{\boldsymbol{x}}_1 = \tilde{\boldsymbol{x}}_i = \tilde{\boldsymbol{x}}_2 = \frac{\boldsymbol{x}_2}{\|\boldsymbol{x}_2\|_p}\,. \tag{3.13}$$

  We then have $\boldsymbol{x}_1 = \frac{\|\boldsymbol{x}_1\|_p}{\|\boldsymbol{x}_2\|_p}\boldsymbol{x}_2$, which means that $\boldsymbol{x}_1, \boldsymbol{x}_2$ belongs to the same set, i.e. $\mathcal{X}^\epsilon_{(1)} = \mathcal{X}^\epsilon_{(2)}$. Hence $\mathcal{X} \leftarrow \tilde{\boldsymbol{x}}_i$.

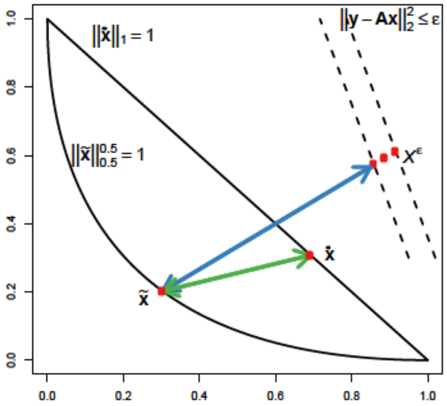


Figure 3.4: The one-to-one mapping: $\mathcal{X}^\epsilon \longleftrightarrow \tilde{x} \longleftrightarrow \ddot{x}$ when $p = 0.5$.

$\square$

Combining Lemma 4 and Lemma 2, we have $\mathcal{X}^\epsilon \longleftrightarrow \ddot{x}_i$ for any $\ddot{x}_i \in \mathcal{X}^\epsilon$, as is illustrated in Fig. 3.4. Let $\mathcal{X} = \{x_1, x_2, \cdots\}$ denote the nonzero solutions that satisfy $\|y - Ax\|_2^2 \leq \epsilon$ in one of the orthants $\mathbb{O}$. Specifically, $\mathcal{X} = \mathcal{X}_1 \cup \mathcal{X}_2$ and $\mathcal{X}_1 \cap \mathcal{X}_2 = \emptyset$, where $\mathcal{X}_1$ contains solutions on the boundary of the orthant $\mathbb{O}$ and $\mathcal{X}_2$ contains the rest solutions that are not on the boundary. Lemma 3 also applies in the noisy case. We can see that minimizing $h_p(x)$ or $h_{p,\alpha}(x)$ in the orthant $\mathbb{O}$ also leads to sparser solutions in $\mathcal{X}_1$ in the noisy case.

Chapter 4

Sparse signal recovery via entropy function minimization

In this chapter we propose the algorithms to perform the sparse signal recovery tasks in (3.5,3.7). Specifically, the proximal regularization [47, 48] of the data fidelity term $f(\boldsymbol{x}) = \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2$ and the first order approximations of the entropy functions $h_p(\boldsymbol{x}), h_{p,\alpha}(\boldsymbol{x})$ are minimized in alternation iteratively until convergence. Experiments on both simulated and real data show the proposed entropy function minimization approach outperforms the state-of-the-art l_1 norm-minimization and the l_p norm-minimization approaches.

4.1 Entropy function minimization

The proposed entropy functions $h_p(\boldsymbol{x})$, $h_{p,\alpha}(\boldsymbol{x})$ are nonconvex, a good initialization is needed to ensure good performance. Here we will use the solution from l_1 norm-minimization as the initialization to our proposed algorithm. The sparsity-promotion analysis in section 3.3 shows that we are able to obtain sparse solutions by minimizing the entropy functions. In order to solve the problems in (3.5,3.7), the following two steps are repeated in alternation until convergence.

1. In the first step, the data fidelity term $f(\boldsymbol{x}) = \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2$ is approximated:

For the $(t+1)$-th iteration to solve the problems $P_{h_p}(\boldsymbol{x})$ and $P_{h_{p,\alpha}}(\boldsymbol{x})$, we use its quadratic approximation, a.k.a. proximal regularization [47], at the previous t-th iteration's solution $\hat{\boldsymbol{x}}^{(t)}$ as is done in [38]:

$$\begin{aligned} f(\boldsymbol{x}) &= \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \\ &\leq f(\hat{\boldsymbol{x}}^{(t)}) + \left\langle \boldsymbol{x} - \hat{\boldsymbol{x}}^{(t)}, \nabla f(\hat{\boldsymbol{x}}^{(t)}) \right\rangle + \frac{\kappa}{2} \left\| \boldsymbol{x} - \hat{\boldsymbol{x}}^{(t)} \right\|_2^2 \\ &= f(\hat{\boldsymbol{x}}^{(t)}) - \frac{1}{2\kappa} \left\| \nabla f(\hat{\boldsymbol{x}}^{(t)}) \right\|_2^2 + \frac{\kappa}{2} \left\| \boldsymbol{x} - \left(\hat{\boldsymbol{x}}^{(t)} - \frac{1}{\kappa} \nabla f(\hat{\boldsymbol{x}}^{(t)}) \right) \right\|_2^2 \\ &= o(\hat{\boldsymbol{x}}^{(t)}) + \frac{\kappa}{2} \left\| \boldsymbol{x} - \left(\hat{\boldsymbol{x}}^{(t)} - \frac{1}{\kappa} \nabla f(\hat{\boldsymbol{x}}^{(t)}) \right) \right\|_2^2 , \end{aligned} \tag{4.1}$$

where $o(\hat{\boldsymbol{x}}^{(t)})$ is a relative constant depending on the previous solution $\hat{\boldsymbol{x}}^{(t)}$, $\nabla f(\hat{\boldsymbol{x}}^{(t)}) = 2(\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}\hat{\boldsymbol{x}}^{(t)} - \boldsymbol{A}^{\mathrm{T}}\boldsymbol{y})$, κ is the Lipschitz constant of the gradient ∇f [49]. The smallest value κ can take is twice the largest eigenvalue of $\boldsymbol{A}^{\mathrm{T}}\boldsymbol{A}$ to ensure that $f(\boldsymbol{x})$ is bounded by the proximal regularization. The problems in (3.5, 3.7) then becomes:

$$P_{h_p}^{(1)}(\boldsymbol{x}): \qquad \min_{\boldsymbol{x}} \quad \frac{\kappa}{2} \left\| \boldsymbol{x} - \left(\hat{\boldsymbol{x}}^{(t)} - \frac{1}{\kappa} \nabla f(\hat{\boldsymbol{x}}^{(t)}) \right) \right\|_2^2 + \lambda h_p(\boldsymbol{x}) \tag{4.2}$$

$$P_{h_{p,\alpha}}^{(1)}(\boldsymbol{x}): \qquad \min_{\boldsymbol{x}} \quad \frac{\kappa}{2} \left\| \boldsymbol{x} - \left(\hat{\boldsymbol{x}}^{(t)} - \frac{1}{\kappa} \nabla f(\hat{\boldsymbol{x}}^{(t)}) \right) \right\|_2^2 + \lambda h_{p,\alpha}(\boldsymbol{x}) . \tag{4.3}$$

2. In then second step, the problems $P_{h_p}^{(1)}(\boldsymbol{x})$ and $P_{h_{p,\alpha}}^{(1)}(\boldsymbol{x})$ are iteratively solved: In the "inner" $(r+1)$-th iteration to solve $P_{h_p}^{(1)}(\boldsymbol{x})$ and $P_{h_{p,\alpha}}^{(1)}(\boldsymbol{x})$, $h_p(\boldsymbol{x})$, $h_{p,\alpha}(\boldsymbol{x})$ are approximated with their first order approximations with respect to $|\hat{\boldsymbol{x}}^{(t+1,r)}|$ from the previous r-th iteration:

$$h_p(\boldsymbol{x}) \approx h_p(\hat{\boldsymbol{x}}^{(t+1,r)}) + \left\langle |\boldsymbol{x}| - |\hat{\boldsymbol{x}}^{(t+1,r)}|, \nabla h_p(\hat{\boldsymbol{x}}^{(t+1,r)}) \right\rangle \tag{4.4}$$

$$h_{p,\alpha}(\boldsymbol{x}) \approx h_{p,\alpha}(\hat{\boldsymbol{x}}^{(t+1,r)}) + \left\langle |\boldsymbol{x}| - |\hat{\boldsymbol{x}}^{(t+1,r)}|, \nabla h_{p,\alpha}(\hat{\boldsymbol{x}}^{(t+1,r)}) \right\rangle . \tag{4.5}$$

The first order derivatives $\nabla h_p(\boldsymbol{x})$, $\nabla h_{p,\alpha}(\boldsymbol{x})$ with respect to $|x_i|$ are as follows:

$$\frac{\partial h_p(\boldsymbol{x})}{\partial |x_i|} = -\frac{p|x_i|^{(p-1)} \log |x_i|^p}{\|\boldsymbol{x}\|_p^p} + \frac{p|x_i|^{(p-1)} \sum_l |x_l|^p \log |x_l|^p}{\|\boldsymbol{x}\|_p^{2p}} \tag{4.6}$$

$$\frac{\partial h_{p,\alpha}(\boldsymbol{x})}{\partial |x_i|} = \frac{1}{1-\alpha} \cdot \frac{1}{\sum_{l=1}^N \left(\frac{|x_l|}{\|\boldsymbol{x}\|_p}\right)^{p\alpha}} \cdot \frac{p\alpha}{\|\boldsymbol{x}\|_p^{p+p\alpha}} \left[|x_i|^{p\alpha-1}\|\boldsymbol{x}\|_p^p - |x_i|^{p-1}\|\boldsymbol{x}\|_{p\alpha}^{p\alpha}\right] . \tag{4.7}$$

Since $\log 0$ is $-\infty$, when computing $\nabla \hbar_p(|\hat{x}_i^{(t+1,r)}|)$, we add a small positive value $\epsilon = 1e^{-12}$ to $|\hat{x}_i^{(t+1,r)}|$ in case $|\hat{x}_i^{(t+1,r)}| = 0$. Ignoring the relative constant terms in (4.4,4.5) that depend on $\hat{\boldsymbol{x}}^{(r)}$, the problems $P_{h_p}^{(1)}(\boldsymbol{x})$ and $P_{h_{p,\alpha}}^{(1)}(\boldsymbol{x})$ then become:

$$P_{h_p}^{(2)}(\boldsymbol{x}): \qquad \min_{\boldsymbol{x}} \quad \frac{\kappa}{2}\left\|\boldsymbol{x} - \left(\hat{\boldsymbol{x}}^{(t)} - \frac{1}{\kappa}\nabla f(\hat{\boldsymbol{x}}^{(t)})\right)\right\|_2^2 + \lambda \left\langle |\boldsymbol{x}|, \nabla h_p(\hat{\boldsymbol{x}}^{(t+1,r)})\right\rangle \tag{4.8}$$

$$P_{h_{p,\alpha}}^{(2)}(\boldsymbol{x}): \qquad \min_{\boldsymbol{x}} \quad \frac{\kappa}{2}\left\|\boldsymbol{x} - \left(\hat{\boldsymbol{x}}^{(t)} - \frac{1}{\kappa}\nabla f(\hat{\boldsymbol{x}}^{(t)})\right)\right\|_2^2 + \lambda \left\langle |\boldsymbol{x}|, \nabla h_{p,\alpha}(\hat{\boldsymbol{x}}^{(t+1,r)})\right\rangle . \tag{4.9}$$

$P_{h_p}^{(2)}(\boldsymbol{x})$ and $P_{h_{p,\alpha}}^{(2)}(\boldsymbol{x})$ are simple reweighted l_1 norm-minimization problems that can be converted to a series of independent one-dimensional problems. The solutions $\hat{x}_i^{(t+1,r+1)}$ to the above problems can be obtained using the iterative shrinkage thresholding algorithm (ISTA):

$$\hat{x}_i^{(t+1,r+1)} = \mathcal{T}_{\frac{\lambda}{\kappa}\nabla h_p(|\hat{x}_i^{(t+1,r)}|)}\left(\hat{x}_i^{(t)} - \frac{1}{\kappa}\nabla f(\hat{x}_i^{(t)})\right) \tag{4.10}$$

$$\hat{x}_i^{(t+1,r+1)} = \mathcal{T}_{\frac{\lambda}{\kappa}\nabla h_{p,\alpha}(|\hat{x}_i^{(t+1,r)}|)}\left(\hat{x}_i^{(t)} - \frac{1}{\kappa}\nabla f(\hat{x}_i^{(t)})\right) , \tag{4.11}$$

where $\mathcal{T}_\tau(\cdot)$ is the soft thresholding function, a.k.a. shrinkage operator, defined as follows:

$$\mathcal{T}_\tau(x) = \begin{cases} 0 & \text{if } |x| \leq \tau \\ (|x| - \tau) \cdot \text{sign}(x) & \text{if } |x| > \tau\,. \end{cases} \tag{4.12}$$

The detailed derivation of (4.10,4.11) can be found in section 4.1.1. ISTA usually converges slowly in practice, [38] proposes a fast iterative shrinkage thresholding algorithm (FISTA) to address this issue. Here we choose the FISTA and propose the Algorithm 1 to compute the recovered signal $\hat{\boldsymbol{x}}$. The Lipschitz constant κ can be viewed as a suitable step size to ensure the upper bound on $f(\boldsymbol{x})$ in (4.1). When κ is unknown or difficult to compute, we can use the backtracking strategy proposed in [38] to find it.

Naturally, choosing a proper λ is the key to the success of sparse signal recovery. For noiseless signals, we will use the fixed-point continuation (FPC) method [50] to solve a series of problems: Starting with a relatively large λ_0, FPC decreases $\lambda^{(t+1)} = \rho\lambda^{(t)}$ in the $(t+1)$-th iteration until $\lambda^{(t+1)}$ is close to 0 and initializes $P_{h_p}^{(1)}(\boldsymbol{x})$, $P_{h_{p,\alpha}}^{(1)}(\boldsymbol{x})$ with the previous solution $\hat{\boldsymbol{x}}^{(t)}$ obtained with $\lambda^{(t)}$. To ensure the best performance, ρ is chosen to be $0.9 \leq \rho < 1$. For noisy signals, usually a fixed λ performs better and is thus preferred. The optimal λ can be tuned on some development set.

Algorithm 1 Sparse signal recovery via entropy function minimization

```
Require: $\{y, A\}, \lambda, \kappa, \{p, \alpha\}$
Initialize $\{\hat{x}^{(t)}, t = 0\}$ by solving $(P_1)$ in (2.2), $c_0 = 1$;
for $t = \{0, 1, \cdots\}$ do
    Compute $\hat{x}^{(t)} - \frac{1}{\kappa}\nabla f(\hat{x}^{(t)})$ in (4.1);
    Initialize $\{\hat{x}^{(t+1,r)}, r = 0\}$ with $\hat{x}^{(t)}$;
    for $r = \{0, 1, \cdots\}$ do
        Compute $\nabla h_p(x)$ or $\nabla h_{p,\alpha}(x)$ in (4.6, 4.7);
        Obtain $\hat{x}^{(t+1,r+1)}$ by solving $P^{(2)}_{h_p}(x)$ or $P^{(2)}_{h_{p,\alpha}}(x)$ in (4.10, 4.11);
        if $\hat{x}^{(t+1,r+1)}$ reaches convergence or the objective functions in (4.2, 4.3) increase then
            $\hat{x}^{(t+1)} = \hat{x}^{(t+1,r+1)}$;
            break;
        end if
    end for
    $c_{t+1} = \frac{1+\sqrt{1+4c_t^2}}{2}$;
    $\hat{x}^{(t+1)} = \hat{x}^{(t+1)} + \left(\frac{c_t - 1}{c_{t+1}}\right)(\hat{x}^{(t+1)} - \hat{x}^{(t)})$;
    if $\hat{x}^{(t+1)}$ reaches convergence then
        $\hat{x} = \hat{x}^{(t+1)}$;
        break;
    end if
end for
Return Output $\hat{x}$;
```

4.1.1 Generalized iterative shrinkage thresholding

Conventional iterative shrinkage thresholding method solves a convex problem and requires the threshold to be positive. However, the derivatives $\nabla h_p(\hat{\boldsymbol{x}}^{(t+1,r)}$ and $\nabla h_{p,\alpha}(\hat{\boldsymbol{x}}^{(t+1,r)}$ in (4.8,4.9) could be negative. Here we show that the optimal solution can still be obtained using the soft shrinkage operator given in (4.12), yet with a completely different derivation process.

Take $P_{h_p}^{(2)}(\boldsymbol{x})$ for example, let $\tau_i^{(t+1,r)} = \frac{2\lambda}{\kappa}\nabla h_p(|\hat{x}_i^{(t+1,r)}|)$, $\tilde{x}_i^{(t)} = \hat{x}_i^{(t)} - \frac{1}{\kappa}\nabla f(\hat{x}_i^{(t)})$, we have the following problem for x_i:

$$\min_{x_i} \quad \left\| x_i - \tilde{x}_i^{(t)} \right\|_2^2 + \tau_i^{(t+1,r)} |x_i| \,. \tag{4.13}$$

When $\tau_i^{(t+1,r)} \geq 0$, (4.13) is a convex problem. Its solution is given by applying the shrinkage operator given in (4.12) on $\tilde{x}_i^{(t)}$ with the threshold $\frac{\tau_i^{(t+1,r)}}{2}$. When $\tau_i^{(t+1,r)} < 0$, (4.13) is a not necessarily a convex problem. Luckily this is a simple one dimensional problem, its global optimal solution can be still found as follows:

1. When $\tilde{x}_i^{(t)} < \frac{\tau_i^{(t+1,r)}}{2}$:

   For $x_i \geq 0$, we have:

   $$\min_{x_i} \quad \left(x_i + \frac{\tau_i^{(t+1,r)} - 2\tilde{x}_i^{(t)}}{2} \right)^2 - \frac{\left(\tau_i^{(t+1,r)} - 2\tilde{x}_i^{(t)}\right)^2}{4} + (\tilde{x}_i^{(t)})^2 \,. \tag{4.14}$$

   Since $\tau_i^{(t+1,r)} - 2\tilde{x}_i^{(t)} > 0$, the x_i that minimizes (4.14) is 0.

   For $x_i < 0$, we have:

   $$\min_{x_i} \quad \left(x_i + \frac{-\tau_i^{(t+1,r)} - 2\tilde{x}_i^{(t)}}{2} \right)^2 - \frac{\left(\tau_i^{(t+1,r)} + 2\tilde{x}_i^{(t)}\right)^2}{4} + (\tilde{x}_i^{(t)})^2 \,. \tag{4.15}$$

   Since $-\tau_i^{(t+1,r)} - 2\tilde{x}_i^{(t)} > 0$, the x_i that minimizes (4.15) is $\frac{\tau_i^{(t+1,r)} + 2\tilde{x}_i^{(t)}}{2}$.

(4.13) is continuous at the point $x_i = 0$. Hence the global minimum of (4.13) is obtained by $x_i = \frac{\tau_i^{(t+1,r)}+2\tilde{x}_i^{(t)}}{2}$.

2. When $\tilde{x}_i^{(t)} \geq \frac{-\tau_i^{(t+1,r)}}{2}$:

   For $x_i \geq 0$, we have (4.14). Since $\tau_i^{(t+1,r)} - 2\tilde{x}_i^{(t)} < 0$, the x_i that minimizes (4.14) is $\frac{-\tau_i^{(t+1,r)}+2\tilde{x}_i^{(t)}}{2}$.

   For $x_i < 0$, we have (4.15). Since $-\tau_i^{(t+1,r)} - 2\tilde{x}_i^{(t)} < 0$, the x_i that minimizes (4.15) is 0.

   (4.13) is continuous at the point $x_i = 0$. Hence the global minimum of (4.13) is obtained by $x_i = \frac{-\tau_i^{(t+1,r)}+2\tilde{x}_i^{(t)}}{2}$.

3. When $0 > \tilde{x}_i^{(t)} \geq \frac{\tau_i^{(t+1,r)}}{2}$:

   For $x_i \geq 0$, we have (4.14). Since $\tau_i^{(t+1,r)} - 2\tilde{x}_i^{(t)} < 0$, the x_i that minimizes (4.14) is $\frac{-\tau_i^{(t+1,r)}+2\tilde{x}_i^{(t)}}{2}$.

   For $x_i < 0$, we have (4.15). Since $-\tau_i^{(t+1,r)} - 2\tilde{x}_i^{(t)} > 0$, the x_i that minimizes (4.15) is $\frac{\tau_i^{(t+1,r)}+2\tilde{x}_i^{(t)}}{2}$.

   It's easy to verify that the minimum of (4.15) is *smaller* than the minimum of (4.14). Hence the global minimum of (4.13) is obtained by $\frac{\tau_i^{(t+1,r)}+2\tilde{x}_i^{(t)}}{2}$.

4. When $0 \leq \tilde{x}_i^{(t)} < \frac{-\tau_i^{(t+1,r)}}{2}$:

   For $x_i \geq 0$, we have (4.14). Since $\tau_i^{(t+1,r)} - 2\tilde{x}_i^{(t)} < 0$, the x_i that minimizes (4.14) is $\frac{-\tau_i^{(t+1,r)}+2\tilde{x}_i^{(t)}}{2}$.

   For $x_i < 0$, we have (4.15). Since $-\tau_i^{(t+1,r)} - 2\tilde{x}_i^{(t)} > 0$, the x_i that minimizes (4.15) is $\frac{\tau_i^{(t+1,r)}+2\tilde{x}_i^{(t)}}{2}$.

   It's easy to verify that the minimum of (4.15) is *larger* than the minimum of (4.14). Hence the global minimum of (4.13) is obtained by $\frac{-\tau_i^{(t+1,r)}+2\tilde{x}_i^{(t)}}{2}$.

Combining the above 4 different scenarios, we have the following results:

1. When $\tilde{x}_i^{(t)} \geq 0$, the solution to (4.13) is $\tilde{x}_i^{(t)} - \frac{\tau_i^{(t+1,r)}}{2}$.
2. When $\tilde{x}_i^{(t)} < 0$, the solution to (4.13) is $\tilde{x}_i^{(t)} + \frac{\tau_i^{(t+1,r)}}{2}$.

This is exactly the shrinkage operator given in (4.12) on $\tilde{x}_i^{(t)}$ with the threshold $\frac{\tau_i^{(t+1,r)}}{2}$.

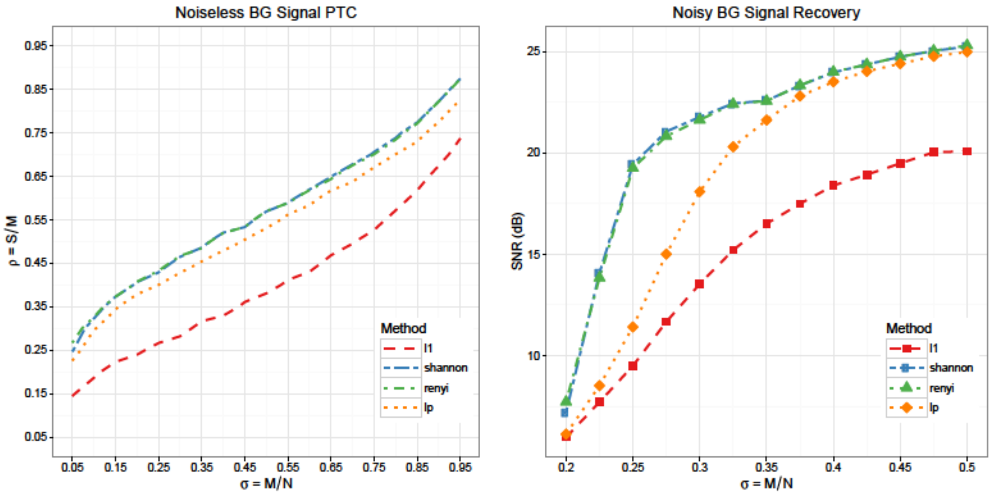
4.2 Experimental results

We compare the proposed Shannon entropy function (SEF) minimization and Rényi entropy function (REF) minimization approaches with the state-of-the-art l_1 norm (L1) minimization and l_p norm (Lp) minimization approaches on simulated and real datasets.

4.2.1 Simulated sparse signal recovery

For the noiseless sparse signal recovery experiments, we fix $N = 1000$ and vary the sampling ratio $\sigma = \frac{M}{N} \in [0.05, 0.1, 0.15, \cdots, 0.95]$ and the sparsity ratio $\rho = \frac{S}{M} \in [0.05, 0.1, 0.15, \cdots, 0.95]$, where S is the sparsity of the signal, i.e. the number of nonzero coefficients. For each combination of σ and ρ, we randomly generate 100 pairs of $\{\boldsymbol{x}, \boldsymbol{A}\}$: A is a $M \times N$ random Gaussian matrix with normalized and centralized rows; the *nonzero* entries of the sparse signal $\boldsymbol{x} \in \mathbb{R}^N$ are i.i.d. generated according to the Gaussian distribution $\mathcal{N}(0, 1)$.

Given the measurement vector $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ and the sensing matrix $\boldsymbol{A}$, we try to recover the signal $\boldsymbol{x}$. If $\epsilon = \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2 / \|\boldsymbol{x}\|_2 < 10^{-3}$, the recovery is considered to be a success. The parameters are selected to obtain best performance for each method: for the SEF minimization approach, $p = 1.1$; for the REF minimization approach, $p = 1.1$, $\alpha = 1.1$; for the Lp minimization approach, $p = 0.5$. FPC



(a) (b)

Figure 4.1: (a) The phase transition curves (PTC) of different sparsity regularization approaches in the noiseless case; (b) The signal-to-noise-ratio (SNR) of the recovered signal $\hat{x}$ using different sparsity regularization approaches in the noisy case.

method [50] is used to approach the optimal $\lambda = 0$. Based on the 100 trials, we compute the success recovery rate for each combination of σ and ρ and plot the PTCs in Fig. 4.1(a).

The PTC is the contour that corresponds to the 0.5 success rate in the domain $(\sigma, \rho) \in (0, 1)^2$, it divides the domain into a "success" phase (lower right) and a "failure" phase (upper left). We can see that the proposed SEF minimization and REF minimization approaches generally perform equally well, and they both perform better than the L1 and Lp minimization approaches.

We next try to recover the sparse signal $\boldsymbol{x}$ from a noisy measurement vector $\boldsymbol{y}$. Specifically, we fix $S = 100$, $N = 1000$ and increase the number of measurement M. $\boldsymbol{y} \in \mathbb{R}^M$ is generated as follows:

$$\text{Noisy measurements:} \qquad \boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \nu \boldsymbol{w}\,, \tag{4.16}$$

(a) (b) (c) (d)

Figure 4.2: The real images used in the recovery experiments: (a) Barbara; (b) Boat; (c) Lena; (d) Peppers.

where $\nu > 0$ controls the amount of noise added to $\boldsymbol{y}$, the entries of $\boldsymbol{w}$ are i.i.d Gaussian $\mathcal{N}(0,1)$. We choose $\nu = 0.1$, this creates a measurement $\boldsymbol{y}$ with signal to noises ratio (SNR) around 20 dB. We randomly generate 100 triples of $\{\boldsymbol{x}, \boldsymbol{A}, \boldsymbol{w}\}$. The average SNRs of the recovered signals $\hat{\boldsymbol{x}}$ are shown in Fig. 4.1(b). We can see that the proposed SEF/REF minimization approaches and the Lp minimization approach perform better than the L1 minimization approach. When $\sigma < 0.5$), the SEF and REF minimization approaches outperform the Lp minimization approach.

4.2.2 Real image recovery

Real images are considered to be approximately sparse under some proper basis, such as the DCT basis, wavelet basis, etc. Here we compare the recovery performances of the aforementioned sparsity regularization approaches based on varying noiseless and noisy measurements of the 4 real images in Fig. 4.2: Barbara, Boat, Lena, Peppers. Specifically, in order to reveal the sparse coefficients $\boldsymbol{x}$ of the real images $\boldsymbol{s}$, we use the sparsity averaging method by [51] to construct an over-complete wavelet basis by concatenating Db1-Db4 [52] as follows:

$$\boldsymbol{V} = \frac{1}{2} \times \left[\boldsymbol{V}_{Db1} \quad \boldsymbol{V}_{Db2} \quad \boldsymbol{V}_{Db3} \quad \boldsymbol{V}_{Db4}\right] . \tag{4.17}$$

It is easy to verify that $\boldsymbol{s} = \boldsymbol{V}\boldsymbol{x}$, and $\boldsymbol{x} = \boldsymbol{V}^{\mathrm{T}}\boldsymbol{s}$. The sampling matrix $\boldsymbol{U}$ is constructed using the structurally random matrix approach by [35]:

$$\boldsymbol{U} = \boldsymbol{D}\boldsymbol{F}\boldsymbol{R}\,, \tag{4.18}$$

where $\boldsymbol{R}$ is a uniform random permutation matrix that scrambles the signal's sample locations globally while a diagonal matrix of Bernoulli random variables flips the signal's sample signs locally, $\boldsymbol{F}$ is an orthonormal DCT matrix that computes fast transforms, $\boldsymbol{D}$ is a sub-sampling matrix that randomly selects a subset of the rows of the matrix $\boldsymbol{F}\boldsymbol{R}$.

The noiseless measurements $\boldsymbol{y}$ of the image $\boldsymbol{s}$ are obtained as follows:

$$\text{Noiseless measurements:} \qquad \boldsymbol{y} = \boldsymbol{D}\boldsymbol{F}\boldsymbol{R}\boldsymbol{V}\boldsymbol{x} = \boldsymbol{U}\boldsymbol{V}\boldsymbol{x} = \boldsymbol{U}\boldsymbol{s}\,. \tag{4.19}$$

The noisy measurements $\boldsymbol{y}$ are obtained as follows:

$$\text{Noisy measurements:} \qquad \boldsymbol{y} = \boldsymbol{U}\boldsymbol{s} + \nu\boldsymbol{w}\,. \tag{4.20}$$

The entries of the noise $\boldsymbol{w}$ are generated using i.i.d. Gaussian distribution $\mathcal{N}(0,1)$, ν is chosen to be 0.02 so that the SNR of the measurement vector $\boldsymbol{y}$ is around 30 dB.

Take the SEF minimization for example, we have the following recovery problem:

$$\min_{\boldsymbol{s}} \qquad \|\boldsymbol{y} - \boldsymbol{U}\boldsymbol{s}\|_2^2 + \lambda\, h_p(\boldsymbol{V}^{\mathrm{T}}\boldsymbol{s})\,. \tag{4.21}$$

Since the recovery problem is with respect to $\boldsymbol{x}$, we need to modified Algorithm 1: we also use the proximal regularization of the data fidelity term $\|\boldsymbol{y} - \boldsymbol{U}\boldsymbol{s}\|_2^2$, the optimization problem in the $(t+1)$-th iteration then becomes:

$$\min_{s} \qquad \frac{\kappa}{2}\left\|s-\left(\hat{s}^{(t)}-\frac{1}{\kappa}\cdot 2U^{\mathrm{T}}(U\hat{s}^{(t)}-y)\right)\right\|_2^2+\lambda\, h(V^{\mathrm{T}}s)\,, \tag{4.22}$$

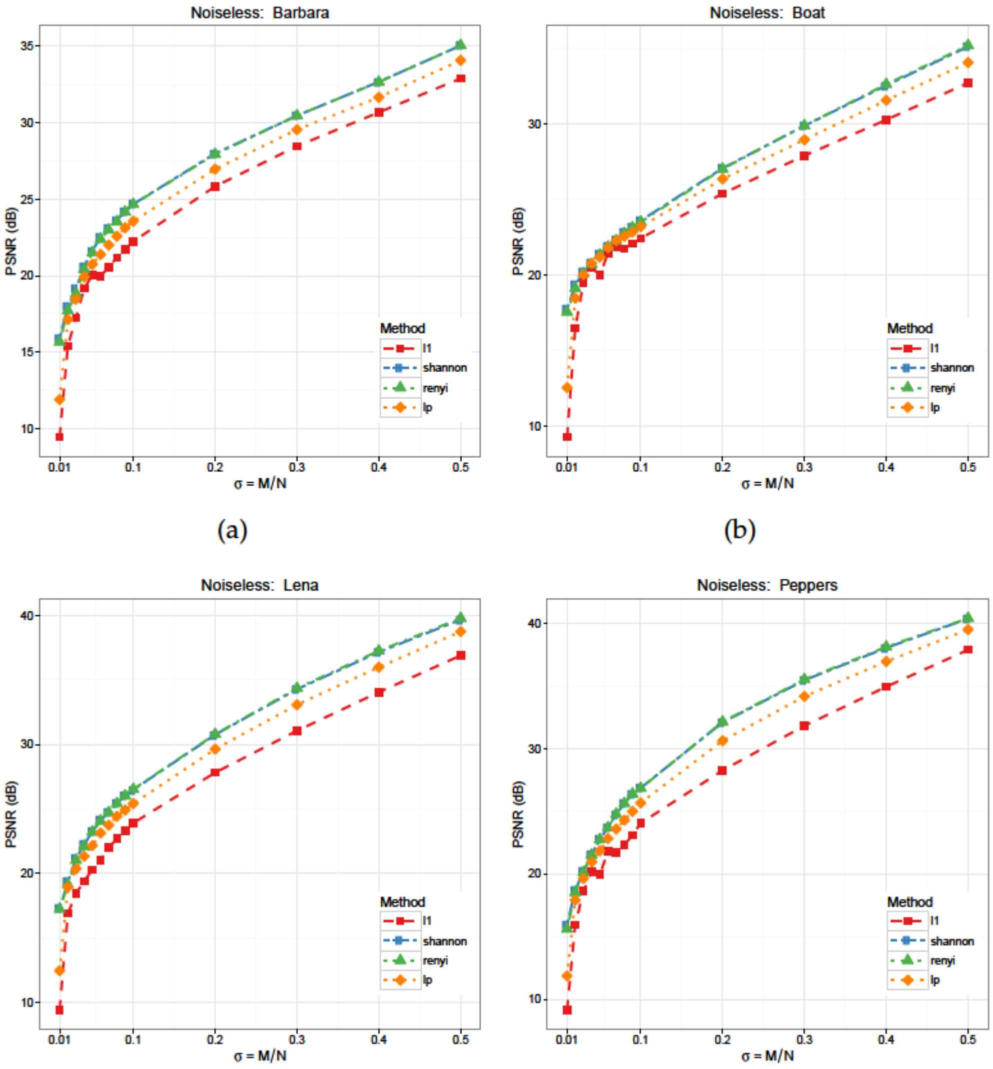
where $\kappa = 2$ for the chosen U in (4.18). In the $(r+1)$-th iteration to minimize (4.22), let $Q^{(t+1,r)}$ be a diagonal matrix whose diagonal entries are the partial derivative of $h_p(V^{\mathrm{T}}s)$ with respect to $\left|V^{\mathrm{T}}s\right|$ at the solution $\hat{s}^{(t+1,r)}$, the optimization problem is as follows:

$$\left\|s-\left(\hat{s}^{(t)}-U^{\mathrm{T}}(U\hat{s}^{(t)}-y)\right)\right\|_2^2+\lambda Q^{(t+1,r)}\left|V^{\mathrm{T}}s\right|\,. \tag{4.23}$$

(4.23) can be efficiently solved using the alternating split bregman shrinkage algorithm by [53].

Since the real images are only approximately sparse, both the noiseless and noisy recovery experiments are done using a fixed λ. The parameters are tuned to obtain best performance for each approach. For the L1 minimization approach, $\lambda = 0.1$; for the SEF minimization approach, $p = 1$, $\lambda = 5000$; for the REF minimization approach, $p = 0.9$, $\alpha = 1.1$, $\lambda = 10000$; for the Lp minimization approach, $p = 0.8$, $\lambda = 0.01$. The peak signal to noise ratios (PSNR) of the noiseless and noisy recovery experiments are shown in Fig. 4.3 and 4.4 respectively. We can see that the proposed SEF and REF entropy function minimization approaches perform equally well, and they give the best performances in terms of PSNR (dB).

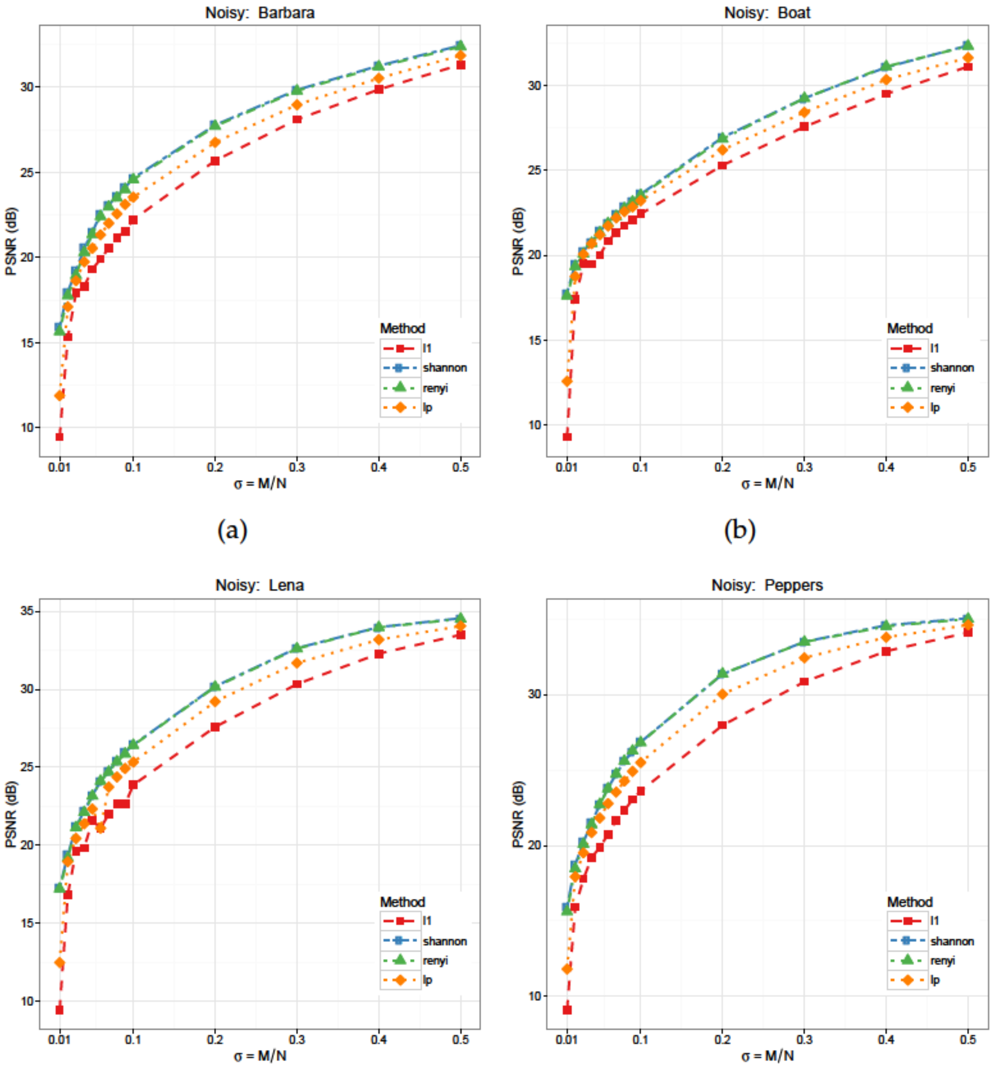
Take the “Lena” image for example, we show the recovered images from noiseless and noisy measurements when the sampling rate $\sigma = 0.2$ in Figure 4.5 and Figure 4.6 respectively. We can see that the images recovered by the proposed SEF and REF minimization methods are cleaner and smoother compared to the other two methods.



(a) (b)

(c) (d)

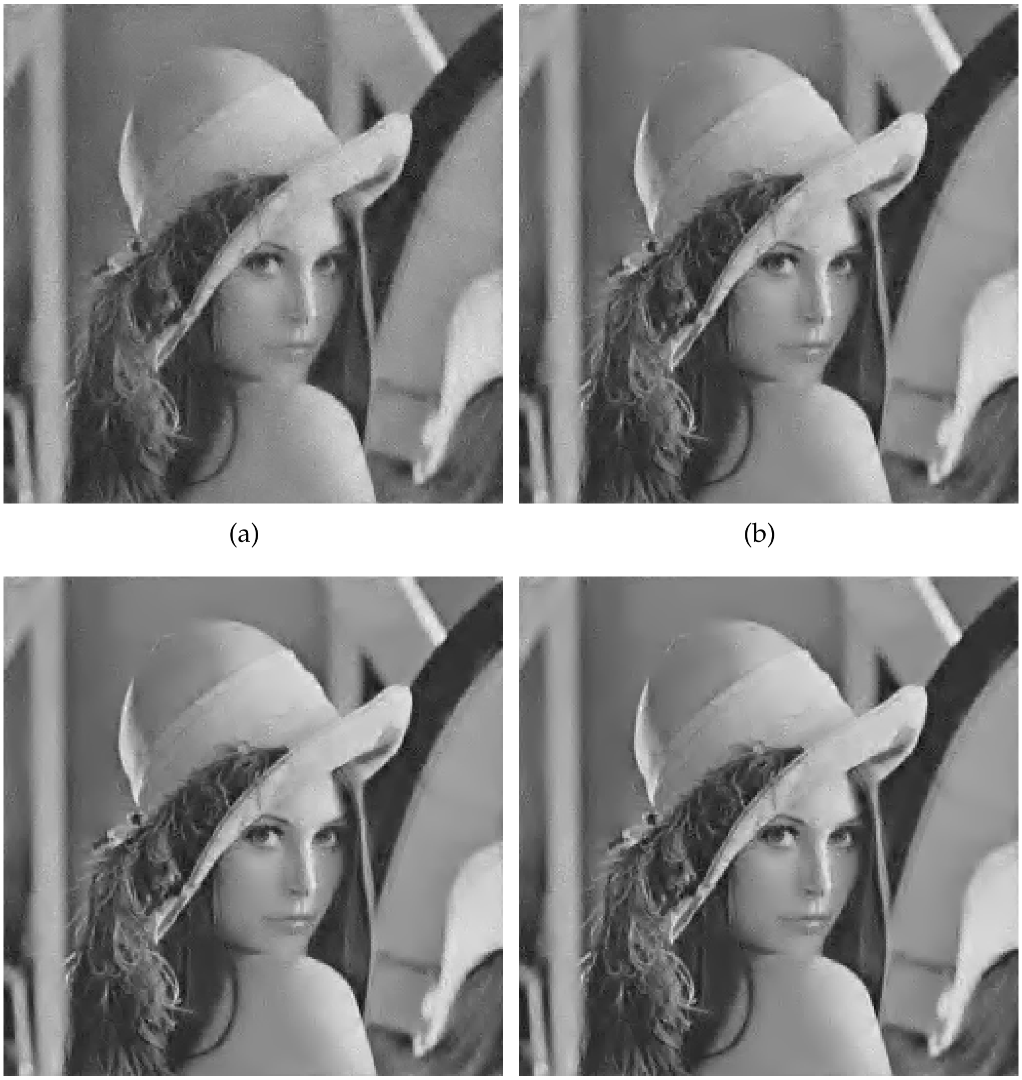
Figure 4.3: The peak-signal-to-noise-ratio (PSNR) of the recovered images from "noiseless" measurements using different sparsity regularization approaches. (a) Barbara; (b) Boat; (c) Lena; (d) Peppers.



(a) (b)

(c) (d)

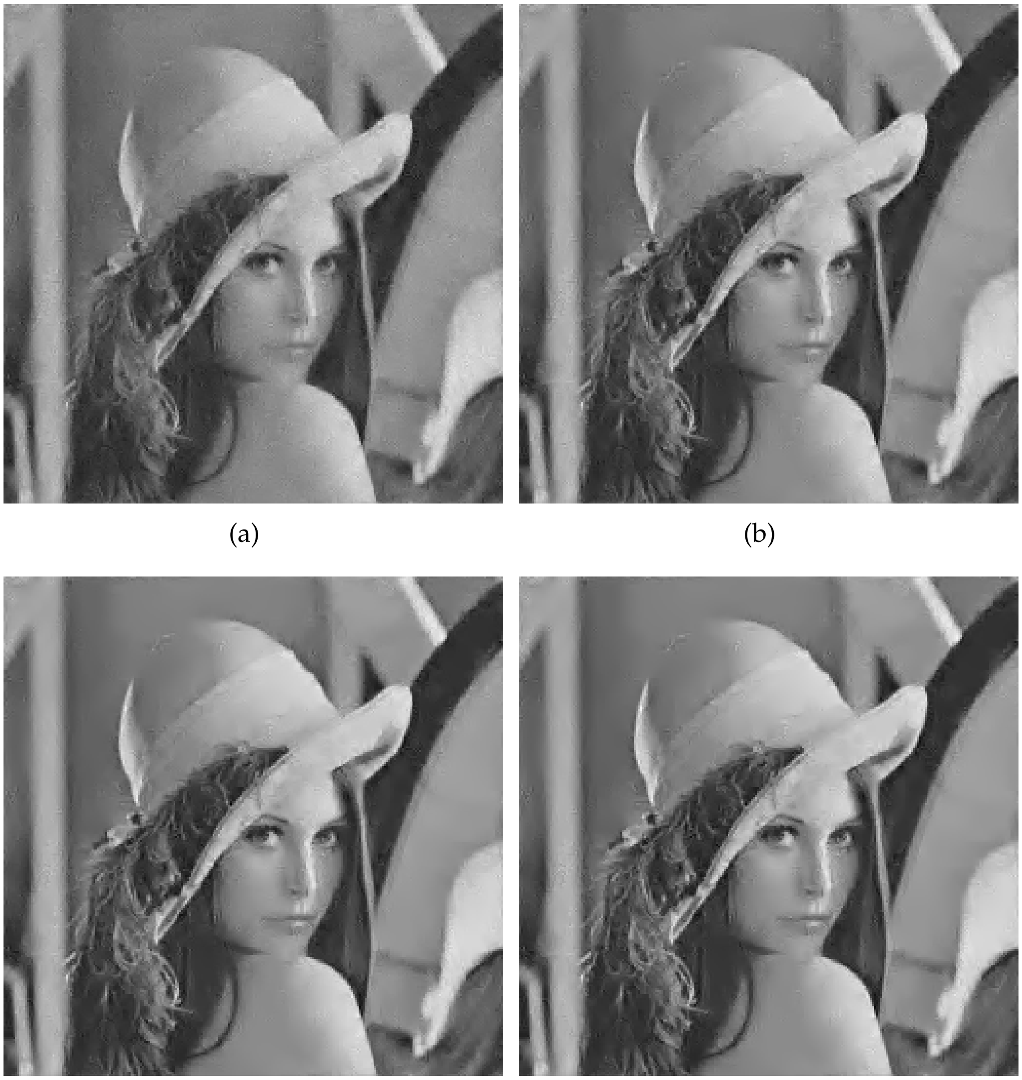
Figure 4.4: The peak-signal-to-noise-ratio (PSNR) of the recovered images from "noisy" measurements using different sparsity regularization approaches. (a) Barbara; (b) Boat; (c) Lena; (d) Peppers.



(a) (b)

(c) (d)

Figure 4.5: The recovered "Lena" image from noiseless measurements using different approaches with a sampling rate of 0.2: a) l_1 norm PSNR=27.81 dB; (b) l_p norm PSNR=29.63 dB; (c) Shannon entropy function $h_p(\boldsymbol{x})$ PSNR=30.79; (d) Rényi entropy function $h_{p,\alpha}(\boldsymbol{x})$ PSNR=30.75 dB.



(a)

(b)

(c)

(d)

Figure 4.6: The recovered "Lena" image from noisy measurements using different approaches with a sampling rate of 0.2: a) l_1 norm PSNR=27.58 dB; (b) l_p norm PSNR=29.21 dB; (c) Shannon entropy function $h_p(\boldsymbol{x})$ PSNR=30.19; (d) Rényi entropy function $h_{p,\alpha}(\boldsymbol{x})$ PSNR=30.15 dB.

Chapter 5

RPCA via entropy function minimization

In this chapter we apply the propose entropy function minimization approach to recover a low rank matrix $\boldsymbol{L}$ corrupted by some sparse noise matrix $\boldsymbol{E}$. In this case the singular values σ of the low rank matrix $\boldsymbol{L}$ and the noise $\boldsymbol{E}$ are both sparse, and we will recover $\boldsymbol{L}$ by minimizing the two entropy functions defined respectively on σ and $\boldsymbol{E}$.

5.1 Introduction to RPCA

Principal component analysis (PCA) enjoys much popularity as the tool to analyze high-dimensional data that lie in a much lower-dimensional linear subspace, i.e. the data is of low-rank. It does so by searching for the subspace that maximizes the variances of said data, however, this procedure would fail when the noise is large enough such that the accurate estimation of the variance is impossible. In the case that the noise is sparsely distributed, it has been shown in [54, 55] that under rather broad conditions, the low rank matrix $\boldsymbol{L}$ can be efficiently and accurately recovered from the corrupted observation $\boldsymbol{D} = \boldsymbol{L} + \boldsymbol{E}$

by solving the following constrained l_1 norm minimization problem:

$$\min_{\boldsymbol{L},\boldsymbol{E}} \quad \|\boldsymbol{L}\|_* + \lambda\|\boldsymbol{E}\|_1 \qquad \text{subject to } \boldsymbol{D} = \boldsymbol{L} + \boldsymbol{E}\,, \tag{5.1}$$

where $\|\boldsymbol{L}\|_* = |\boldsymbol{\sigma}|_1$ is the nuclear norm of $\boldsymbol{L}$, $\lambda > 0$. $\boldsymbol{L}$ can also be obtained by solving the unconstrained problem:

$$\min_{\boldsymbol{L},\boldsymbol{E}} \quad \|\boldsymbol{D} - \boldsymbol{L} - \boldsymbol{E}\|_F^2 + \mu\|\boldsymbol{L}\|_* + \mu\lambda\|\boldsymbol{E}\|_1\,, \tag{5.2}$$

where $\{\mu, \lambda\} > 0$, $\|\cdot\|_F$ is the Frobenius norm of a matrix. Various methods can be used to solve the recovery problem [56–60]. The recovered $\boldsymbol{L}$ is rid of the sparse noise, which makes the PCA on $\boldsymbol{L}$ more robust. This approach is thus called robust principal component analysis (RPCA).

5.2 Entropy function minimization

Here we give the formulations of RPCA via entropy function minimization. Specifically, the following recovery problems need to be solved:

$$P_{h_p}(\boldsymbol{L},\boldsymbol{E}): \qquad \min_{\boldsymbol{L},\boldsymbol{E}} \quad \|\boldsymbol{D} - \boldsymbol{L} - \boldsymbol{E}\|_F^2 + \mu \cdot h_p(\boldsymbol{\sigma}) + \mu\lambda \cdot h_p(\boldsymbol{E}) \tag{5.3}$$

$$P_{h_{p,\alpha}}(\boldsymbol{L},\boldsymbol{E}): \qquad \min_{\boldsymbol{L},\boldsymbol{E}} \quad \|\boldsymbol{D} - \boldsymbol{L} - \boldsymbol{E}\|_F^2 + \mu \cdot h_{p,\alpha}(\boldsymbol{\sigma}) + \mu\lambda \cdot h_{p,\alpha}(\boldsymbol{E})\,. \tag{5.4}$$

Searching for the minimizing $\boldsymbol{L}, \boldsymbol{E}$ together is often quite difficult. In practice, we just keep one of them fixed while searching for the other. As is done in section 4.1, we also use the proximal regularization of the data fidelity term $f(\boldsymbol{L},\boldsymbol{E}) = \|\boldsymbol{D} - \boldsymbol{L} - \boldsymbol{E}\|_F^2$ due to its efficiency and adaptability, and search for $\boldsymbol{L}, \boldsymbol{E}$ iteratively until convergence.

- **Low rank matrix recovery:** In the $(t+1)$-th iteration, the proximal regularization of $f(\boldsymbol{L}, \hat{\boldsymbol{E}}^{(t)})$ at the solution $\hat{\boldsymbol{L}}^{(t)}$ from the previous t-th iteration is as follows:

$$\begin{aligned} f(\boldsymbol{L},\hat{\boldsymbol{E}}^{(t)}) &= \|\boldsymbol{D}-\boldsymbol{L}-\hat{\boldsymbol{E}}^{(t)}\|_F^2 \\ &\le o(\hat{\boldsymbol{L}}^{(t)},\hat{\boldsymbol{E}}^{(t)}) + \frac{\kappa_{\boldsymbol{L}}}{2}\left\|\boldsymbol{L}-\widetilde{\boldsymbol{L}}^{(t)}\right\|_F^2 , \end{aligned} \tag{5.5}$$

where $\kappa_{\boldsymbol{L}} \ge 2$ is the Lipschitz constant, $\widetilde{\boldsymbol{L}}^{(t)} = \hat{\boldsymbol{L}}^{(t)} - \frac{2}{\kappa_{\boldsymbol{L}}}\left(\hat{\boldsymbol{L}}^{(t)} + \hat{\boldsymbol{E}}^{(t)} - \boldsymbol{D}\right)$. Ignoring the constant term $o(\hat{\boldsymbol{L}}^{(t)},\hat{\boldsymbol{E}}^{(t)})$, the first step of the low rank matrix recovery problems are then:

$$P^{(1)}_{h_p}(\boldsymbol{L}): \qquad \min_{\boldsymbol{L}} \quad \frac{\kappa_{\boldsymbol{L}}}{2}\|\boldsymbol{L}-\widetilde{\boldsymbol{L}}^{(t)}\|_F^2 + \mu\cdot h_p(\boldsymbol{\sigma}) \tag{5.6}$$

$$P^{(1)}_{h_{p,\alpha}}(\boldsymbol{L}): \qquad \min_{\boldsymbol{L}} \quad \frac{\kappa_{\boldsymbol{L}}}{2}\|\boldsymbol{L}-\widetilde{\boldsymbol{L}}^{(t)}\|_F^2 + \mu\cdot h_{p,\alpha}(\boldsymbol{\sigma})\,. \tag{5.7}$$

In the second step of the recovery problems, the entropy functions $h_p(\boldsymbol{\sigma})$, $h_{p,\alpha}(\boldsymbol{\sigma})$ are approximated using their first-order approximation as before and iteratively minimized until convergence. For the $(r+1)$-th iteration to solve (5.6,5.7), we have:

$$P^{(2)}_{h_p}(\boldsymbol{L}): \qquad \min_{\boldsymbol{L}} \quad \frac{\kappa_{\boldsymbol{L}}}{2}\|\boldsymbol{L}-\widetilde{\boldsymbol{L}}^{(t)}\|_F^2 + \mu\cdot\left\langle \boldsymbol{\sigma}, \nabla h_p(\hat{\boldsymbol{\sigma}}^{(t+1,r)})\right\rangle \tag{5.8}$$

$$P^{(2)}_{h_{p,\alpha}}(\boldsymbol{L}): \qquad \min_{\boldsymbol{L}} \quad \frac{\kappa_{\boldsymbol{L}}}{2}\|\boldsymbol{L}-\widetilde{\boldsymbol{L}}^{(t)}\|_F^2 + \mu\cdot\left\langle \boldsymbol{\sigma}, \nabla h_{p,\alpha}(\hat{\boldsymbol{\sigma}}^{(t+1,r)})\right\rangle , \tag{5.9}$$

where $\hat{\boldsymbol{\sigma}}^{(t+1,r)}$ are the singular values of the low rank matrix $\hat{\boldsymbol{L}}^{(t+1,r)}$ in the previous r-th iteration. Suppose the singular value decomposition (SVD) of the matrix $\widetilde{\boldsymbol{L}}^{(t)}$ is as follows:

$$\widetilde{\boldsymbol{L}}^{(t)} = \widetilde{\boldsymbol{U}}^{(t)}\cdot\widetilde{\boldsymbol{S}}^{(t)}\cdot\widetilde{\boldsymbol{V}}^{(t)^{\mathrm{T}}} , \tag{5.10}$$

where $\boldsymbol{S}$ is a diagonal matrix whose diagonal entries are the singular values $\widetilde{\boldsymbol{\sigma}}^{(t)}$ of $\widetilde{\boldsymbol{L}}^{(t)}$. The solutions to problems like (5.8,5.9) are known to have the close forms [58]:

$$\hat{\boldsymbol{L}}^{(t+1,r+1)} = \widetilde{\boldsymbol{U}}^{(t)} \cdot \hat{\boldsymbol{S}}^{(t+1,r+1)} \cdot \widetilde{\boldsymbol{V}}^{(t)^{\mathrm{T}}}, \tag{5.11}$$

where $\hat{\boldsymbol{S}}^{(t+1,r+1)}$ is a diagonal matrix whose entries $\hat{\boldsymbol{\sigma}}^{(t+1,r+1)}$ are obtained using the following thresholding operator in (4.12) on $\widetilde{\boldsymbol{\sigma}}^{(t)}$:

$$\hat{\sigma}_i^{(t+1,r+1)} = \mathcal{T}_{\frac{\mu \nabla h_p\left(\hat{\sigma}_i^{(t+1,r)}\right)}{\kappa_{\boldsymbol{L}}}}(\widetilde{\sigma}_i^{(t)}) \tag{5.12}$$

$$\hat{\sigma}_i^{(t+1,r+1)} = \mathcal{T}_{\frac{\mu \nabla h_{p,\alpha}\left(\hat{\sigma}_i^{(t+1,r)}\right)}{\kappa_{\boldsymbol{L}}}}(\widetilde{\sigma}_i^{(t)}) \,. \tag{5.13}$$

In [57], $\kappa_{\boldsymbol{L}}$ is chosen to be 4, which is larger than the smallest Lipschitz constant. To maintain consistency, in this dissertation we also set $\kappa_{\boldsymbol{L}} = 4$.

- **Sparse matrix recovery:** In the $(t+1)$-th iteration, the proximal regularization of $f(\hat{\boldsymbol{L}}^{(t)}, \boldsymbol{E})$ at the solution $\hat{\boldsymbol{E}}^{(t)}$ from the previous t-th iteration is as follows:

$$\begin{aligned} f(\hat{\boldsymbol{L}}^{(t)}, \boldsymbol{E}) &= \|\boldsymbol{D} - \hat{\boldsymbol{L}}^{(t)} - \boldsymbol{E}\|_F^2 \\ &\leq o(\hat{\boldsymbol{L}}^{(t)}, \hat{\boldsymbol{E}}^{(t)}) + \frac{\kappa_{\boldsymbol{E}}}{2} \left\|\boldsymbol{E} - \widetilde{\boldsymbol{E}}^{(t)}\right\|_F^2, \end{aligned} \tag{5.14}$$

where $\kappa_{\boldsymbol{E}} \geq 2$ is the Lipschitz constant, and $\widetilde{\boldsymbol{E}}^{(t)} = \hat{\boldsymbol{E}}^{(t)} - \frac{2}{\kappa_{\boldsymbol{E}}} \left(\hat{\boldsymbol{L}}^{(t)} + \hat{\boldsymbol{E}}^{(t)} - \boldsymbol{D}\right)$. Ignoring the constant term $o(\hat{\boldsymbol{L}}^{(t)}, \hat{\boldsymbol{E}}^{(t)})$, the first step of the low rank matrix recovery problems are then:

$$P_{h_p}^{(1)}(\boldsymbol{E}): \qquad \min_{\boldsymbol{E}} \quad \frac{\kappa_{\boldsymbol{E}}}{2}\|\boldsymbol{E} - \widetilde{\boldsymbol{E}}^{(t)}\|_F^2 + \mu\lambda \cdot h_p(\boldsymbol{E}) \tag{5.15}$$

$$P_{h_{p,\alpha}}^{(1)}(\boldsymbol{E}): \qquad \min_{\boldsymbol{E}} \quad \frac{\kappa_{\boldsymbol{E}}}{2}\|\boldsymbol{E} - \widetilde{\boldsymbol{E}}^{(t)}\|_F^2 + \mu\lambda \cdot h_{p,\alpha}(\boldsymbol{E}) \,. \tag{5.16}$$

In the second step of the recovery problems, the entropy functions $h_p(\boldsymbol{E})$,

$h_{p,\alpha}(\boldsymbol{E})$ are approximated using their first-order approximation and iteratively minimized until convergence as is done in section 4.1. In the $(r+1)$-th iteration to solve (5.15,5.16), we have:

$$P^{(2)}_{h_p}(\boldsymbol{E}): \qquad \min_{\boldsymbol{E}} \quad \frac{\kappa_{\boldsymbol{E}}}{2}\|\boldsymbol{E}-\widetilde{\boldsymbol{E}}^{(t)}\|_F^2 + \mu\lambda \cdot \left\langle \boldsymbol{E}, \nabla h_p(\hat{\boldsymbol{E}}^{(t+1,r)}) \right\rangle \tag{5.17}$$

$$P^{(2)}_{h_{p,\alpha}}(\boldsymbol{E}): \qquad \min_{\boldsymbol{E}} \quad \frac{\kappa_{\boldsymbol{E}}}{2}\|\boldsymbol{E}-\widetilde{\boldsymbol{E}}^{(t)}\|_F^2 + \mu\lambda \cdot \left\langle \boldsymbol{E}, \nabla h_{p,\alpha}(\hat{\boldsymbol{E}}^{(t+1,r)}) \right\rangle, \tag{5.18}$$

where $\hat{\boldsymbol{E}}^{(t+1,r)}$ are the sparse error matrix in the previous r-th iteration. The entries of $\hat{\boldsymbol{E}}^{(t+1,r+1)}$ are obtained using the following thresholding operator on $\widetilde{\boldsymbol{E}}^{(t)}$:

$$\hat{E}^{(t+1,r+1)}_{ij} = \mathcal{T}_{\frac{\mu\lambda\nabla h_p\left(\hat{E}^{(t+1,r)}_{ij}\right)}{\kappa_{\boldsymbol{E}}}}(\widetilde{E}^{(t)}_{ij}) \tag{5.19}$$

$$\hat{E}^{(t+1,r+1)}_{ij} = \mathcal{T}_{\frac{\mu\lambda\nabla h_{p,\alpha}\left(\hat{E}^{(t+1,r)}_{ij}\right)}{\kappa_{\boldsymbol{E}}}}(\widetilde{E}^{(t)}_{ij}). \tag{5.20}$$

In practice, the parameter λ is usually chosen to be fixed, while the parameter μ starts with a relatively large value and is reduced by a rate $\eta < 1$ after each iteration. Using FISTA [38] to speed up the convergence, we summarize the proposed approach in Algorithm 2.

5.3 Experimental results

For the experiments on the recovery of the low rank matrix $\boldsymbol{L}$ corrupted by sparse noise $\boldsymbol{E}$, the $M \times M$ matrix $\boldsymbol{L}$ is generated as a product of two independent $M \times R$ matrices whose entries follow independently identically distributed Gaussian distribution $\mathcal{N}(0,1)$; the nonzero entries of $M \times M$ matrix $\boldsymbol{E}$ are independently identically generated following the uniform distribution in the range $[-500, 500]$ and their positions are chosen randomly as well. We then have the corrupted

Algorithm 2 RPCA via entropy function minimization

```
Require: $\boldsymbol{D}, \{\mu, \lambda, \eta\}, \{\kappa_{\boldsymbol{L}}, \kappa_{\boldsymbol{E}}\}, \{p, \alpha\}$
Initialize $\{\hat{\boldsymbol{L}}^{(t)}, \hat{\boldsymbol{E}}^{(t)}, t = 0\}$ and $c_0 = 1$;
for $t = \{0, 1, \cdots\}$ do
    Compute $\widetilde{\boldsymbol{L}}^{(t)}, \widetilde{\boldsymbol{E}}^{(t)}$ in (5.5, 5.14);
    Initialize $\{\hat{\sigma}^{(t+1,r)}, r = 0\}$ with $\hat{\sigma}^{(t)}$;
    for $r = \{0, 1, \cdots\}$ do
        Obtain $\hat{\sigma}^{(t+1,r+1)}$ by solving $P^{(2)}_{h_p}(\boldsymbol{L})$ or $P^{(2)}_{h_{p,\alpha}}(\boldsymbol{L})$ in (5.8, 5.9);
        if $\hat{\sigma}^{(t+1,r+1)}$ reaches convergence or the objective functions in (5.6, 5.9) increase then
            $\hat{\sigma}^{(t+1)} = \hat{\sigma}^{(t+1,r+1)}$;
            break;
        end if
    end for
    Initialize $\{\hat{\boldsymbol{E}}^{(t+1,r)} r = 0\}$ with $\hat{\boldsymbol{E}}^{(t)}$ respectively;
    for $r = \{0, 1, \cdots\}$ do
        Obtain $\hat{\boldsymbol{E}}^{(t+1,r+1)}$ by solving $P^{(2)}_{h_p}(\boldsymbol{E})$ or $P^{(2)}_{h_{p,\alpha}}(\boldsymbol{E})$ in (5.17, 5.18);
        if $\hat{\boldsymbol{E}}^{(t+1,r+1)}$ reaches convergence or the objective functions in (5.15, 5.16) increase then
            $\hat{\boldsymbol{E}}^{(t+1)} = \hat{\boldsymbol{E}}^{(t+1,r+1)}$;
            break;
        end if
    end for
    $c_{t+1} = \frac{1+\sqrt{1+4c_t^2}}{2}$;
    $\hat{\boldsymbol{L}}^{(t+1)} = \hat{\boldsymbol{L}}^{(t+1)} + \left(\frac{c_t-1}{c_{t+1}}\right)(\hat{\boldsymbol{L}}^{(t+1)} - \hat{\boldsymbol{L}}^{(t)})$;
    $\hat{\boldsymbol{E}}^{(t+1)} = \hat{\boldsymbol{E}}^{(t+1)} + \left(\frac{c_t-1}{c_{t+1}}\right)(\hat{\boldsymbol{E}}^{(t+1)} - \hat{\boldsymbol{E}}^{(t)})$;
    Reduce the value of $\mu$: $\mu = \eta\mu$;
    if $\hat{\boldsymbol{L}}^{(t+1)}, \hat{\boldsymbol{E}}^{(t+1)}$ reach convergence then
        $\hat{\boldsymbol{L}} = \hat{\boldsymbol{L}}^{(t+1)}, \hat{\boldsymbol{E}} = \hat{\boldsymbol{E}}^{(t+1)}$;
        break;
    end if
end for
Return Output $\hat{\boldsymbol{L}}, \hat{\boldsymbol{E}}$;
```

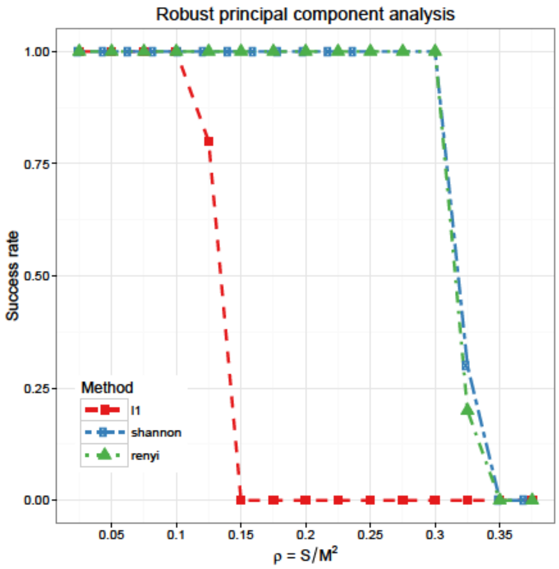


Figure 5.1: The recovery of a $M \times M$ low rank matrix $\boldsymbol{L}$ corrupted by sparse noisy matrix $\boldsymbol{E}$ using RPCA.

observations $\boldsymbol{D} = \boldsymbol{L} + \boldsymbol{E}$. Using the same experimental settings as in [55], we fix the size of L to be $M = 200$ rank of $\boldsymbol{L}$ to be $R = 40$, and increase the number of nonzero entries of $\boldsymbol{E}$ gradually $S = 1000, 2000, \cdots, 15000$. If the recovered low rank matrix $\hat{\boldsymbol{L}}$ satisfies $\frac{\|\hat{\boldsymbol{L}}-\boldsymbol{L}\|_F}{\|\boldsymbol{L}\|_F} < 1e^{-2}$, the recovery is considered to be a success. We randomly generated 100 pairs of $\boldsymbol{L}, \boldsymbol{E}$ for different S.

The parameters are tuned for each method to obtain best performance: for L1 minimization, $\lambda = M^{-0.5}$; for SEF minimization, $\lambda = 10$, $p = 1$; for REF minimization, $\lambda = 10$, $p = 11$, $\alpha = 9$. The μ sequence starts with 10^4, and decreases at a rate of $\eta = 0.99$. The results in terms of success rate are shown in Fig. 5.1. We can see that the proposed SEF and REF minimization approaches perform much better than the conventional L1 regularized RPCA by [54, 55].

Part III

Probabilistic approach

Chapter 6

GAMP with built-in parameter estimation

In this chapter we propose the generalized approximate message passing (GAMP) framework with built-in parameter estimation and give its approximate loopy belief propagation realizations. Specifically, the parameters $\{\boldsymbol{\theta}, \boldsymbol{\lambda}\}$ are treated as unknown random variables following some simple prior distributions. and estimated jointly with the signal $\boldsymbol{x}$.

6.1 Prior work

Following the probabilistic interpretation of the under-determined linear system (1.2,1.3) in Fig. 1.2, the entries of $\boldsymbol{x}$ and $\boldsymbol{w}$ are assumed to be i.i.d distributed according to some distributions $p(x_j; \boldsymbol{\lambda})$, $p(w_i; \boldsymbol{\theta})$ respectively. Probabilistic inferences can then be performed on the corresponding factor graph to recover $\boldsymbol{x}$ using Gaussian/quadratic approximations of loopy belief propagation, a.k.a. message passing [24]. Based on different inference tasks, loopy BP has the following two variants:

- *Sum-product* message passing for the MMSE estimation of $\boldsymbol{x}$: The "marginal" posterior distributions $\{p(x_j; \boldsymbol{\lambda}|\boldsymbol{y}),\ p(w_i; \boldsymbol{\theta}|\boldsymbol{y})\}$ can be obtained.

- *Max-sum* message passing for the MAP estimation of $\boldsymbol{x}$: The $\widetilde{\boldsymbol{x}}, \widetilde{\boldsymbol{w}}$ that maximize the "joint" posterior distribution $p(\widetilde{\boldsymbol{x}}, \widetilde{\boldsymbol{w}}; \boldsymbol{\lambda}, \boldsymbol{\theta} | \boldsymbol{y})$ can be obtained.

In [23, 61, 62] the *approximate* message passing (AMP) algorithm based on a quadratic approximation of *max-sum* message passing is proposed, it has low complexity and can be used to find solutions of the l_1 norm minimization problem $P_{l_1}(\boldsymbol{x})$ accurately.

Various methods based on the AMP framework have been proposed over the years [22, 63, 64]. In this dissertation we focus on the generalized version of the AMP algorithm (GAMP) by [22]. Compared with other AMP algorithms, GAMP can work with essentially arbitrary input and output channel distributions, and approximate both the *sum-product* and *max-sum* message passings using only scalar estimations and linear transforms. The parameters $\{\boldsymbol{\lambda}, \boldsymbol{\theta}\}$ in the input and output channels are usually unknown, and need to be decided for the AMP/GAMP algorithm.

In [25, 26, 65, 66], Expectation Maximization (EM) [27] algorithm is used to perform parameter estimation for the GAMP. They treat $\boldsymbol{x}$ as the hidden variable and tries to find the parameters that maximize the likelihood $p(\boldsymbol{y}; \boldsymbol{\lambda}, \boldsymbol{\theta})$. However, EM based parameter estimation is not widely applicable, it has high complexity and can only be used with *sum-product* message passing.

6.2 Message passings in PE-GAMP

The factor graph of the proposed GAMP framework with built-in parameter estimation (PE-GAMP) that treats the parameters as random variables is shown in Fig. 6.1. Inference tasks performed on the factor graph rely on the "messages" passed among connected nodes of the graph. Here we adopt the same notations

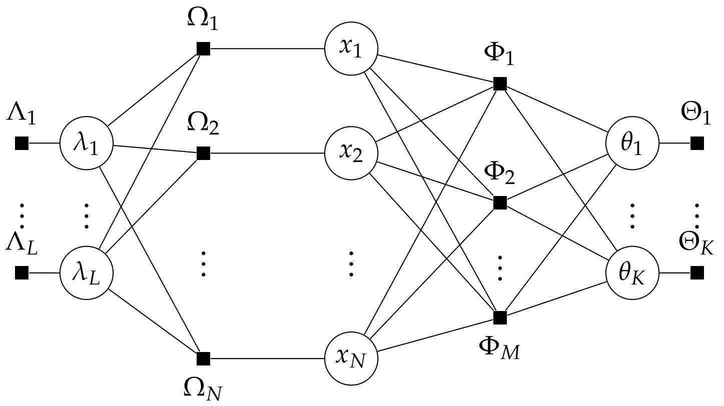


Figure 6.1: The factor graph for the proposed PE-GAMP. "■" represents the factor node, and "◯" represents the variable node. Here, $\boldsymbol{\lambda} = \{\lambda_1, \cdots, \lambda_L\}$ and $\boldsymbol{\theta} = \{\theta_1, \cdots, \theta_K\}$ are the parameters whereas $\boldsymbol{x} = [x_1, \cdots, x_N]^{\mathrm{T}}$ is the sparse signal.

used by [22]. Take the messages being passed between the factor node Φ_m and the variable node x_n for example, $\Delta_{\Phi_m \rightarrow x_n}$ is the message from Φ_m to x_n, and $\Delta_{\Phi_m \leftarrow x_n}$ is the message from x_n to Φ_m. Both $\Delta_{\Phi_m \rightarrow x_n}$ and $\Delta_{\Phi_m \leftarrow x_n}$ can be viewed as functions of x_n. In the following section 6.2.1 and 6.2.2, we give the messages being passed on the generalized factor graph in log domain for the *sum-product* message passing algorithm and the *max-sum* message passing algorithm respectively.

6.2.1 Sum-product message passing

Sum-product message passing is used to compute the marginal distributions of the random variables in the graph: $p(\boldsymbol{x}|\boldsymbol{y}), p(\boldsymbol{\lambda}|\boldsymbol{y}), p(\boldsymbol{\theta}|\boldsymbol{y})$. In the following, we first present the *sum-product* message updates equations in the $(t+1)$-th iteration.

$$\begin{aligned}\Delta^{(t+1)}_{\Phi_m \to x_n} &= \text{const} + \log \int_{\boldsymbol{x} \backslash x_n, \boldsymbol{\theta}} \Big[\Phi_m\left(y_m, \boldsymbol{x}, \boldsymbol{\theta}\right) \\ &\qquad \times \exp\left(\textstyle\sum_{j \neq n} \Delta^{(t)}_{\Phi_m \leftarrow x_j} + \sum_v \Delta^{(t)}_{\Phi_m \leftarrow \theta_v}\right)\Big]\end{aligned} \tag{6.1a}$$

$$\Delta^{(t+1)}_{\Phi_m \leftarrow x_n} = \text{const} + \Delta^{(t+1)}_{\Omega_n \to x_n} + \textstyle\sum_{i \neq m} \Delta^{(t+1)}_{\Phi_i \to x_n} \tag{6.1b}$$

$$\Delta^{(t+1)}_{\Omega_n \to x_n} = \text{const} + \log \int_{\boldsymbol{\lambda}} \Omega_n(x_n, \boldsymbol{\lambda}) \cdot \exp\left(\textstyle\sum_u \Delta^{(t)}_{\Omega_n \leftarrow \lambda_u}\right) \tag{6.1c}$$

$$\Delta^{(t+1)}_{\Omega_n \leftarrow x_n} = \text{const} + \textstyle\sum_i \Delta^{(t+1)}_{\Phi_i \to x_n}\,, \tag{6.1d}$$

where $\boldsymbol{x} \backslash x_n$ denotes the sequence obtained by removing x_n from $\boldsymbol{x}$, $\Phi_m(y_m, \boldsymbol{x}, \boldsymbol{\theta}) = p(y_m | \boldsymbol{x}, \boldsymbol{\theta})$ and $\Omega_n(x_n, \boldsymbol{\lambda}) = p(x_n | \boldsymbol{\lambda})$. Similarly, we can write the message updates involving the variable nodes λ_l, θ_k as follows:

$$\begin{aligned}\Delta^{(t+1)}_{\Omega_n \to \lambda_l} &= \text{const} + \log \int_{x_n, \boldsymbol{\lambda} \backslash \lambda_l} \Big[\Omega_n(x_n, \boldsymbol{\lambda}) \\ &\qquad \times \exp\left(\Delta^{(t+1)}_{\Omega_n \leftarrow x_n} + \textstyle\sum_{u \neq l} \Delta^{(t)}_{\Omega_n \leftarrow \lambda_u}\right)\Big]\end{aligned} \tag{6.2a}$$

$$\Delta^{(t+1)}_{\Omega_n \leftarrow \lambda_l} = \text{const} + \textstyle\sum_{j \neq n} \Delta^{(t+1)}_{\Omega_j \to \lambda_l} + \log p(\lambda_l) \tag{6.2b}$$

$$\begin{aligned}\Delta^{(t+1)}_{\Phi_m \to \theta_k} &= \text{const} + \log \int_{\boldsymbol{\theta} \backslash \theta_k, \boldsymbol{x}} \Big[\Phi_m\left(y_m, \boldsymbol{x}, \boldsymbol{\theta}\right) \\ &\qquad \times \exp\left(\textstyle\sum_j \Delta^{(t)}_{\Phi_m \leftarrow x_j} + \sum_{v \neq k} \Delta^{(t)}_{\Phi_m \leftarrow \theta_v}\right)\Big]\end{aligned} \tag{6.2c}$$

$$\Delta^{(t+1)}_{\Phi_m \leftarrow \theta_k} = \text{const} + \textstyle\sum_{i \neq m} \Delta^{(t+1)}_{\Phi_i \to \theta_k} + \log p(\theta_k)\,, \tag{6.2d}$$

where $p(\lambda_l), p(\theta_k)$ are the pre-specified priors of the parameters. Let $\Gamma(x_n)$, $\Gamma(\lambda_l)$, $\Gamma(\theta_k)$ denote the factor nodes in the neighborhood of the variable nodes

x_n, λ_l, θ_k respectively, we have the following posterior marginals:

$$p(x_n|\boldsymbol{y}) \propto \exp \Delta_{x_n}^{(t+1)} = \exp \left(\Delta_{\Omega_n \to x_n}^{(t+1)} + \textstyle\sum_{\Phi_m \in \Gamma(x_n)} \Delta_{\Phi_m \to x_n}^{(t+1)} \right) \tag{6.3a}$$

$$p(\lambda_l|\boldsymbol{y}) \propto \exp \Delta_{\lambda_l}^{(t+1)} = \exp \left(\log p(\lambda_l) + \textstyle\sum_{\Omega_n \in \Gamma(\lambda_l)} \Delta_{\Omega_n \to \lambda_l}^{(t+1)} \right) \tag{6.3b}$$

$$p(\theta_k|\boldsymbol{y}) \propto \exp \Delta_{\theta_k}^{(t+1)} = \exp \left(\log p(\theta_k) + \textstyle\sum_{\Phi_m \in \Gamma(\theta_k)} \Delta_{\Phi_m \to \theta_k}^{(t+1)} \right) . \tag{6.3c}$$

Using $p(x_n|\boldsymbol{y})$, the MMSE estimate of $\boldsymbol{x}$ can then be computed:

$$\hat{x}_n = \mathbb{E}\left[x_n|\boldsymbol{y}\right] = \int_{x_n} x_n p(x_n|\boldsymbol{y})\,. \tag{6.4}$$

6.2.2 Max-sum message passing

Max-sum message passing is used to compute the "joint" MAP estimates of the random variables in the graph:

$$(\hat{\boldsymbol{x}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\theta}}) = \arg\max_{\boldsymbol{x},\boldsymbol{\lambda},\boldsymbol{\theta}} p(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\theta}|\boldsymbol{y})\,. \tag{6.5}$$

For the *max-sum* message passing, the message updates from the variable nodes to the factor nodes are the same as the aforementioned *sum-product* message updates, i.e. (6.6b, 6.6d, 6.7b, 6.7d). We only need to change the message updates from the factor nodes to the variable nodes by replacing $\int$ with max. Specifically, we have the following message updates between the variable node x_n and the factor nodes Φ_m, Ω_n in the $(t+1)$-th iteration:

$$\Delta^{(t+1)}_{\Phi_m \to x_n} = \text{const} + \max_{\boldsymbol{x} \backslash x_n, \boldsymbol{\theta}} \Big[\log \Phi_m \left(y_m, \boldsymbol{x}, \boldsymbol{\theta} \right) \tag{6.6a}$$

$$+ \textstyle\sum_{j \neq n} \Delta^{(t)}_{\Phi_m \leftarrow x_j} + \sum_v \Delta^{(t)}_{\Phi_m \leftarrow \theta_v} \Big]$$

$$\Delta^{(t+1)}_{\Phi_m \leftarrow x_n} = \text{const} + \Delta^{(t+1)}_{\Omega_n \to x_n} + \textstyle\sum_{i \neq m} \Delta^{(t+1)}_{\Phi_i \to x_n} \tag{6.6b}$$

$$\Delta^{(t+1)}_{\Omega_n \to x_n} = \text{const} + \max_{\boldsymbol{\lambda}} \Big[\log \Omega_n (x_n, \boldsymbol{\lambda}) + \textstyle\sum_u \Delta^{(t)}_{\Omega_n \leftarrow \lambda_u} \Big] \tag{6.6c}$$

$$\Delta^{(t+1)}_{\Omega_n \leftarrow x_n} = \text{const} + \textstyle\sum_i \Delta^{(t+1)}_{\Phi_i \to x_n} . \tag{6.6d}$$

The message updates involving the variable nodes λ_l, θ_k are then:

$$\Delta^{(t+1)}_{\Omega_n \to \lambda_l} = \text{const} + \max_{x_n, \boldsymbol{\lambda} \backslash \lambda_l} \Big[\log \Omega_n (x_n, \boldsymbol{\lambda}) \tag{6.7a}$$

$$+ \Delta^{(t+1)}_{\Omega_n \leftarrow x_n} + \textstyle\sum_{u \neq l} \Delta^{(t)}_{\Omega_n \leftarrow \lambda_u} \Big]$$

$$\Delta^{(t+1)}_{\Omega_n \leftarrow \lambda_l} = \text{const} + \textstyle\sum_{j \neq n} \Delta^{(t+1)}_{\Omega_j \to \lambda_l} + \log p(\lambda_l) \tag{6.7b}$$

$$\Delta^{(t+1)}_{\Phi_m \to \theta_k} = \text{const} + \max_{\boldsymbol{\theta} \backslash \theta_k, \boldsymbol{x}} \Big[\log \Phi_m \left(y_m, \boldsymbol{x}, \boldsymbol{\theta} \right) \tag{6.7c}$$

$$+ \textstyle\sum_j \Delta^{(t)}_{\Phi_m \leftarrow x_j} + \sum_{v \neq k} \Delta^{(t)}_{\Phi_m \leftarrow \theta_v} \Big]$$

$$\Delta^{(t+1)}_{\Phi_m \leftarrow \theta_k} = \text{const} + \textstyle\sum_{i \neq m} \Delta^{(t+1)}_{\Phi_i \to \theta_k} + \log p(\theta_k) . \tag{6.7d}$$

Similarly, we have the following posterior distributions that are different from those in (6.3):

$$\begin{aligned} p(x_n, \hat{\boldsymbol{x}}^{(t+1)} \backslash \hat{x}_n^{(t+1)}, \hat{\boldsymbol{\lambda}}^{(t+1)}, \hat{\boldsymbol{\theta}}^{(t+1)} | \boldsymbol{y}) &\propto \exp \Delta_{x_n}^{(t+1)} \\ &= \exp \left(\Delta_{\Omega_n \to x_n}^{(t+1)} + \textstyle\sum_{\Phi_m \in \Gamma(x_n)} \Delta_{\Phi_m \to x_n}^{(t+1)} \right) \end{aligned} \tag{6.8a}$$

$$\begin{aligned} p(\hat{\boldsymbol{x}}^{(t+1)}, \lambda_l, \hat{\boldsymbol{\lambda}}^{(t+1)} \backslash \hat{\lambda}_l^{(t+1)} | \boldsymbol{y}) &\propto \exp \Delta_{\lambda_l}^{(t+1)} \\ &= \exp \left(\log p(\lambda_l) + \textstyle\sum_{\Omega_n \in \Gamma(\lambda_l)} \Delta_{\Omega_n \to \lambda_l}^{(t+1)} \right) \end{aligned} \tag{6.8b}$$

$$\begin{aligned} p(\hat{\boldsymbol{x}}^{(t+1)}, \theta_k, \hat{\boldsymbol{\theta}}^{(t+1)} \backslash \hat{\theta}_k^{(t+1)} | \boldsymbol{y}) &\propto \exp \Delta_{\theta_k}^{(t+1)} \\ &= \exp \left(\log p(\theta_k) + \textstyle\sum_{\Phi_m \in \Gamma(\theta_k)} \Delta_{\Phi_m \to \theta_k}^{(t+1)} \right) , \end{aligned} \tag{6.8c}$$

where $\hat{\boldsymbol{x}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\theta}}$ are the maximizing values computed from (6.6a,6.6c,6.7a,6.7c) accordingly. The "joint" MAP estimates of the signal $\boldsymbol{x}$ and the parameters $\boldsymbol{\lambda}, \boldsymbol{\theta}$ are then:

$$\hat{x}_n = \arg\max_{x_n} \; p(x_n, \hat{\boldsymbol{x}}^{(t+1)} \backslash \hat{x}_n^{(t+1)}, \hat{\boldsymbol{\lambda}}^{(t+1)}, \hat{\boldsymbol{\theta}}^{(t+1)} | \boldsymbol{y}) \tag{6.9a}$$

$$\hat{\lambda}_l = \arg\max_{\lambda_l} \; p(\hat{\boldsymbol{x}}^{(t+1)}, \lambda_l, \hat{\boldsymbol{\lambda}}^{(t+1)} \backslash \hat{\lambda}_l^{(t+1)} | \boldsymbol{y}) \tag{6.9b}$$

$$\hat{\theta}_k = \arg\max_{\theta_k} \; p(\hat{\boldsymbol{x}}^{(t+1)}, \theta_k, \hat{\boldsymbol{\theta}}^{(t+1)} \backslash \hat{\theta}_k^{(t+1)} | \boldsymbol{y}) \,. \tag{6.9c}$$

6.2.3 The PE-GAMP algorithm

The priors $p(\lambda_l), p(\theta_k)$ on the parameters are usually chosen to be some simple distributions. If we do not have any knowledge on how $\boldsymbol{\lambda}, \boldsymbol{\theta}$ are distributed, we can fairly assume a uniform prior and treat $p(\lambda_l), p(\theta_k)$ as constants. Since λ_l, θ_k are treated as random variables in the PE-GAMP framework, they will be jointly estimated along with the signal $\boldsymbol{x}$ in the message-updating process.

❑ **Sum-product message passing:** Take λ_l for example, in the PE-GAMP, we propose to approximate the underlying distribution $p^{(t+1)}_{\Omega_n \leftarrow \lambda_l}(\lambda_l|\boldsymbol{y}) \propto \exp(\Delta^{(t+1)}_{\Omega_n \leftarrow \lambda_l})$ using Dirac delta function:

$$p^{(t+1)}_{\Omega_n \leftarrow \lambda_l}(\lambda_l|\boldsymbol{y}) \approx \delta\left(\lambda_l - \hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}\right) , \tag{6.10}$$

where $\delta(\cdot)$ is the Dirac delta function, $\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}$ can be computed using either the MAP or MMSE estimation:

$$\text{MAP estimation of } \lambda_l\text{: } \hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l} := \arg\max_{\lambda_l} \Delta^{(t+1)}_{\Omega_n \leftarrow \lambda_l} \tag{6.11a}$$

$$\text{MMSE estimation of } \lambda_l\text{: } \hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l} := \mathbb{E}[\lambda_l|\Delta^{(t+1)}_{\Omega_n \leftarrow \lambda_l}] , \tag{6.11b}$$

where $\mathbb{E}[\lambda_l|\Delta^{(t+1)}_{\Omega_n \leftarrow \lambda_l}]$ is the mean of the distribution $\frac{1}{\mathcal{C}} \exp(\Delta^{(t+1)}_{\Omega_n \leftarrow \lambda_l})$, $\mathcal{C}$ is a normalizing constant.

The formulations for the rest parameters can be derived similarly. The reason behind the choice of Dirac delta approximation of $p^{(t+1)}_{\Omega_n \leftarrow \lambda_l}(\lambda_l|\boldsymbol{y})$ is its simplicity, it amounts to the scalar MAP or MMSE estimation of λ_l from the posterior distribution $p^{(t+1)}_{\Omega_n \leftarrow \lambda_l}(\lambda_l|\boldsymbol{y})$. Other approximations often make it quite difficult to compute the message $\Delta^{(t+1)}_{\Omega_n \rightarrow \lambda_l}$ in (6.2a) due to the lack of closed-form solutions.

The updated messages from the factor nodes to the variable nodes are then:

$$\begin{aligned}\Delta^{(t+1)}_{\Phi_m\to x_n} &= \text{const} + \log \int_{\boldsymbol{x}\backslash x_n} \Big[\Phi_m\left(y_m, \boldsymbol{x}, \hat{\boldsymbol{\theta}}^{(t)}_{\Phi_m}\right) \\ &\qquad \times \exp\left(\textstyle\sum_{j\neq n} \Delta^{(t)}_{\Phi_m\leftarrow x_j}\right)\Big]\end{aligned} \tag{6.12a}$$

$$\Delta^{(t+1)}_{\Omega_n\to x_n} = \text{const} + \log \Omega_n(x_n, \hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n}) \tag{6.12b}$$

$$\begin{aligned}\Delta^{(t+1)}_{\Omega_n\to \lambda_l} &= \text{const} + \log \int_{x_n} \Big[\Omega_n\left(x_n, \lambda_l, \hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n}\backslash \hat{\lambda}^{(t)}_{\Omega_n\leftarrow\lambda_l}\right) \\ &\qquad \times \exp\left(\Delta^{(t+1)}_{\Omega_n\leftarrow x_n}\right)\Big]\end{aligned} \tag{6.12c}$$

$$\begin{aligned}\Delta^{(t+1)}_{\Phi_m\to \theta_k} &= \text{const} + \log \int_{\boldsymbol{x}} \Big[\Phi_m\left(y_m, \boldsymbol{x}, \theta_k, \hat{\boldsymbol{\theta}}^{(t)}_{\Phi_m}\backslash \hat{\theta}^{(t)}_{\Phi_m\leftarrow\theta_k}\right) \\ &\qquad \times \exp\left(\textstyle\sum_{j} \Delta^{(t)}_{\Phi_m\leftarrow x_j}\right)\Big] ,\end{aligned} \tag{6.12d}$$

where $\hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n}$, $\hat{\boldsymbol{\theta}}^{(t)}_{\Phi_m}$ are scalar estimates from the previous t-th iteration at nodes Ω_n and Φ_m respectively.

$$\hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n} = \left\{ \hat{\lambda}^{(t)}_{\Omega_n\leftarrow\lambda_u} \middle| u = 1, \cdots, L \right\} \tag{6.13a}$$

$$\hat{\boldsymbol{\theta}}^{(t)}_{\Phi_m} = \left\{ \hat{\theta}^{(t)}_{\Phi_m\leftarrow\theta_v} \middle| v = 1, \cdots, K \right\} . \tag{6.13b}$$

❑ **Max-sum message passing:** Take λ_l for example, a straightforward way to solve the problems in (6.6c, 6.7a) is to iteratively maximize each variable in $\{x_n, \boldsymbol{\lambda}\backslash\lambda_l\}$ while keeping the rest fixed until convergence. However, it is inefficient and quite unnecessary. In practice one iteration would suffice. Hence we propose to use the following solutions as the approximate maximizing parameters:

$$\hat{\lambda}^{(t+1)}_{\Omega_n\leftarrow\lambda_l} = \arg\max_{\lambda_l} \log \Omega_n\left(\hat{x}^{(t)}_n, \lambda_l, \hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n}\backslash \hat{\lambda}^{(t)}_{\Omega_n\leftarrow\lambda_l}\right) + \Delta^{(t)}_{\Omega_n\leftarrow\lambda_l}. \tag{6.14}$$

The updated messages from the factor nodes to the variable nodes can be obtained by replacing "$\int$" in (6.12) with "max" like before.

For the rest of the paper, parameter estimation operations like those in (6.11, 6.14) will be abbreviated by the two functions $f_{\Omega_n \leftarrow \lambda_l}(\cdot), f_{\Phi_m \leftarrow \theta_k}(\cdot)$.

$$\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l} = f_{\Omega_n \leftarrow \lambda_l}(\cdot) \quad \text{and} \quad \hat{\theta}^{(t+1)}_{\Phi_m \leftarrow \theta_k} = f_{\Phi_m \leftarrow \theta_k}(\cdot)\,. \tag{6.15}$$

They are different from the input and output channels estimation functions $g_{\text{in}}(\cdot), g_{\text{out}}(\cdot)$ defined in [22].

The proposed GAMP algorithm with built-in parameter estimation (PE-GAMP) can be summarized in Algorithm 3, where $\boldsymbol{q}_\Phi \in \mathbb{R}^M, \boldsymbol{r}_\Omega \in \mathbb{R}^N$ can be viewed as some new random variables created inside the original GAMP framework [22], and $\boldsymbol{\tau}^q_\Phi \in \mathbb{R}^M, \boldsymbol{\tau}^s_\Phi \in \mathbb{R}^M, \boldsymbol{\tau}^r_\Omega \in \mathbb{R}^N$ are their corresponding variances. As is done in [22], further simplification will be made by replacing the variance vectors with scalars when performing asymptotic analysis of Algorithm 3:

$$\boldsymbol{\tau}^q_\Phi, \boldsymbol{\tau}^r_\Omega \xRightarrow{\text{Replace}} \tau^q_\Phi, \tau^r_\Omega\,. \tag{6.16}$$

For the *sum-product* message passing, PE-GAMP naturally produces MMSE estimation of $\boldsymbol{x}$ in (6.20a). After the convergence is reached, we can also compute the MAP estimation of $\boldsymbol{x}$ using $p(x_n|\boldsymbol{y})$: $\hat{x}_n = \arg\max_{x_n} p(x_n|\boldsymbol{y})$. For the *max-sum* message passing, PE-GAMP naturally produces the "joint" MAP estimation of $\boldsymbol{x}$ in (6.20a). However, there isn't any meaningful MMSE estimation of $\boldsymbol{x}$ in this case.

Algorithm 3 The PE-GAMP algorithm

```
Require: The matrix $A \in \mathbb{R}^{M\times N}$; the observation $y \in \mathbb{R}^{M}$; the input and output channels estimation functions $g_{\text{in}}(\cdot), g_{\text{out}}(\cdot)$; the parameter estimation functions $f_{\Omega_n}(\cdot), f_{\Phi_m}(\cdot)$.
Set $\boldsymbol{s}^{(-1)} = 0$ and initialize $\hat{\boldsymbol{x}}^{(0)}, \boldsymbol{\tau}^{x}_{\Omega}(0), \hat{\boldsymbol{\lambda}}^{(0)}_{\Omega_n}, \hat{\boldsymbol{\theta}}^{(0)}_{\Phi_m}$.
for $t = \{0, 1, \cdots\}$ do
    Output channel *linear* update: For each $m = 1, \cdots, M$
        $\tau^{q}_{\Phi_m}(t) = \sum_n |A_{mn}|^2 \tau^{x}_{\Omega_n}(t)$    (6.17a)
        $q^{(t)}_{\Phi_m} = \sum_n A_{mn}\hat{x}^{(t)}_n - \tau^{q}_{\Phi_m}(t) s^{(t-1)}_m$    (6.17b)
        $\hat{z}^{(t)}_m = \sum_n A_{mn}\hat{x}^{(t)}_n \, .$    (6.17c)
    Output channel *nonlinear* update: For each $m = 1, \cdots, M$
        $s^{(t)}_{\Phi_m} = g_{\text{out}}\left(t, q^{(t)}_{\Phi_m}, \tau^{q}_{\Phi_m}(t), y_m, \hat{\boldsymbol{\theta}}^{(t)}_{\Phi_m}\right)$    (6.18a)
        $\tau^{s}_{\Phi_m}(t) = -\frac{\partial}{\partial q} g_{\text{out}}\left(t, q^{(t)}_{\Phi_m}, \tau^{q}_{\Phi_m}(t), y_m, \hat{\boldsymbol{\theta}}^{(t)}_{\Phi_m}\right) \, .$    (6.18b)
    Input channel *linear* update: For each $n = 1, \cdots, N$
        $\tau^{r}_{\Omega_n}(t) = \left[\sum_m |A_{mn}|^2 \tau^{s}_{\Phi_m}(t)\right]^{-1}$    (6.19a)
        $r^{(t)}_{\Omega_n} = x^{(t)}_n + \tau^{r}_{\Omega_n}(t) \sum_m A_{mn} s^{(t)}_m \, .$    (6.19b)
    Input *nonlinear* update: For each $n = 1, \cdots, N$
        $\hat{x}^{(t+1)}_n = g_{\text{in}}\left(t, r^{(t)}_{\Omega_n}, \tau^{r}_{\Omega_n}(t), \hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n}\right)$    (6.20a)
        $\tau^{x}_{\Omega_n}(t+1) = \tau^{r}_{\Omega_n}(t) \frac{\partial}{\partial r} g_{\text{in}}\left(t, r^{(t)}_{\Omega_n}, \tau^{r}_{\Omega_n}(t), \hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n}\right) \, .$    (6.20b)
```

Algorithm 3 The PE-GAMP algorithm (continued)

```
    *Sum-product* message passing parameters update: For each $k = 1, \cdots, K$ and $l = 1, \cdots, L$.
        $\hat{\lambda}_{\Omega_n \leftarrow \lambda_l}^{(t+1)} = f_{\Omega_n \leftarrow \lambda_l}\left(t, \boldsymbol{r}_{\Omega}^{(t)}, \boldsymbol{\tau}_{\Omega}^{r}(t), \lambda_l, \hat{\boldsymbol{\lambda}}_{\Omega_n}^{(t)} \backslash \hat{\lambda}_{\Omega_n \leftarrow \lambda_l}^{(t)}\right) \qquad (6.21a)$
        $\hat{\theta}_{\Phi_m \leftarrow \theta_k}^{(t+1)} = f_{\Phi_m \leftarrow \theta_k}\left(t, \boldsymbol{q}_{\Phi}^{(t)}, \boldsymbol{y}, \boldsymbol{\tau}_{\Phi}^{q}(t), \theta_k, \hat{\boldsymbol{\theta}}_{\Phi_m}^{(t)} \backslash \hat{\theta}_{\Phi_m \leftarrow \theta_k}^{(t)}\right) . \qquad (6.21b)$
    *Max-sum* message passing parameters update: For each $k = 1, \cdots, K$ and $l = 1, \cdots, L$.
        $\hat{\lambda}_{\Omega_n \leftarrow \lambda_l}^{(t+1)} = f_{\Omega_n \leftarrow \lambda_l}\left(t, \hat{x}_n^{(t)}, \boldsymbol{r}_{\Omega}^{(t)}, \boldsymbol{\tau}_{\Omega}^{r}(t), \lambda_l, \hat{\boldsymbol{\lambda}}_{\Omega_n}^{(t)} \backslash \hat{\lambda}_{\Omega_n \leftarrow \lambda_l}^{(t)}\right) \qquad (6.22a)$
        $\hat{\theta}_{\Phi_m \leftarrow \theta_k}^{(t+1)} = f_{\Phi_m \leftarrow \theta_k}\left(t, \hat{\boldsymbol{z}}^{(t)}, \boldsymbol{q}_{\Phi}^{(t)}, \boldsymbol{y}, \boldsymbol{\tau}_{\Phi}^{q}(t), \theta_k, \hat{\boldsymbol{\theta}}_{\Phi_m}^{(t)} \backslash \hat{\theta}_{\Phi_m \leftarrow \theta_k}^{(t)}\right) . \qquad (6.22b)$
    **if** $\hat{\boldsymbol{x}}^{(t+1)}$ reaches convergence **then**
        $\hat{\boldsymbol{x}} = \hat{\boldsymbol{x}}^{(t+1)}$;
        break;
    **end if**
**end for**
**Return** Output $\hat{\boldsymbol{x}}$;
```

6.3 Sum-product PE-GAMP

Approximate message passing uses quadratic/Gaussian approximations of the messages from the variable nodes to the factor nodes to perform loopy belief propagation. To maintain consistency with [22], we use the same notations for the quadratic approximations of messages involving $\boldsymbol{x}$. Specifically, $\Delta_{x_n}^{(t)}, \Delta_{\Phi_m \leftarrow x_n}^{(t)}$ in the t-th iteration can be used to construct the following distributions about x_n:

$$p(x_n|\boldsymbol{y}) \propto \exp(\Delta_{x_n}^{(t)}) \tag{6.23a}$$

$$p_{\Phi_m \leftarrow x_n}^{(t)}(x_n|\boldsymbol{y}) \propto \exp(\Delta_{\Phi_m \leftarrow x_n}^{(t)}) \,. \tag{6.23b}$$

We then have the following expectations and variances definitions:

$$\hat{x}_n^{(t)} := \mathbb{E}[x_n|\Delta_{x_n}^{(t)}] \tag{6.24a}$$

$$\tau_{\Omega_n}^x(t) := \text{var}[x_n|\Delta_{x_n}^{(t)}] \tag{6.24b}$$

$$\hat{x}_{\Phi_m \leftarrow x_n}^{(t)} := \mathbb{E}[x_n|\Delta_{\Phi_m \leftarrow x_n}^{(t)}] \tag{6.24c}$$

$$\tau_{\Phi_m \leftarrow x_n}^x(t) := \text{var}[x_n|\Delta_{\Phi_m \leftarrow x_n}^{(t)}] \,. \tag{6.24d}$$

If the entries a_{mn} of the sensing matrix A is small, $\tau_{\Omega_n}^x(t) \approx \tau_{\Phi_m \leftarrow x_n}^x(t)$. The message $\Delta_{\Phi_m \leftarrow x_n}^{(t)}$ in the t-th iteration will be approximated quadratically:

$$\begin{aligned} \Delta_{\Phi_m \leftarrow x_n}^{(t)} &\approx \text{const} - \frac{1}{2\tau_{\Phi_m \leftarrow x_n}^x(t)} \left(x_n - \hat{x}_{\Phi_m \leftarrow x_n}^{(t)} \right)^2 \\ &\approx \text{const} - \frac{1}{2\tau_{\Omega_n}^x(t)} \left(x_n - \hat{x}_{\Phi_m \leftarrow x_n}^{(t)} \right)^2 , \end{aligned} \tag{6.25}$$

which makes the approximation of $p^{(t)}_{\Phi_m \leftarrow x_n}(x_n|\boldsymbol{y})$ a Gaussian distribution. Similarly we have the following approximations for $\boldsymbol{x}$ involving the node Ω_n:

$$\hat{x}^{(t)}_{\Omega_n \leftarrow x_n} := \mathbb{E}[x_n | \Delta^{(t)}_{\Omega_n \leftarrow x_n}] \tag{6.26a}$$

$$\tau^{x}_{\Omega_n \leftarrow x_n}(t) := \mathrm{var}[x_n | \Delta^{(t)}_{\Omega_n \leftarrow x_n}] \tag{6.26b}$$

$$\Delta^{(t)}_{\Omega_n \leftarrow x_n} \approx \mathrm{const} - \frac{1}{2\tau^{x}_{\Omega_n \leftarrow x_n}(t)} \left(x_n - \hat{x}^{(t)}_{\Omega_n \leftarrow x_n} \right)^2 . \tag{6.26c}$$

In the proposed PE-GAMP, we use Dirac delta approximation of the messages involving the parameters $\boldsymbol{\lambda}, \boldsymbol{\theta}$. Specifically, the parameters are estimated using MAP or MMSE estimations:

1. MAP estimation:

$$\hat{\lambda}^{(t)}_{\Omega_n \leftarrow \lambda_l} := \arg \max_{\lambda_l} \Delta^{(t)}_{\Omega_n \leftarrow \lambda_l} \tag{6.27a}$$

$$\hat{\theta}^{(t)}_{\Phi_m \leftarrow \theta_k} := \arg \max_{\theta_k} \Delta^{(t)}_{\Phi_m \leftarrow \theta_k} . \tag{6.27b}$$

2. MMSE estimation:

$$\hat{\lambda}^{(t)}_{\Omega_n \leftarrow \lambda_l} := \mathbb{E}[\lambda_l | \Delta^{(t)}_{\Omega_n \leftarrow \lambda_l}] \tag{6.28a}$$

$$\hat{\theta}^{(t)}_{\Phi_m \leftarrow \theta_k} := \mathbb{E}[\theta_k | \Delta^{(t)}_{\Phi_m \leftarrow \theta_k}] . \tag{6.28b}$$

The corresponding messages involving the parameters $\boldsymbol{\lambda}, \boldsymbol{\theta}$ in the (t)-th iteration can then be approximated as follows:

$$\exp\left(\Delta^{(t)}_{\Omega_n \leftarrow \lambda_l} \right) \approx \delta \left(\lambda_l - \hat{\lambda}^{(t)}_{\Omega_n \leftarrow \lambda_l} \right) \tag{6.29a}$$

$$\exp\left(\Delta^{(t)}_{\Phi_m \leftarrow \theta_k} \right) \approx \delta \left(\theta_k - \hat{\theta}^{(t)}_{\Phi_m \leftarrow \theta_k} \right) . \tag{6.29b}$$

Using approximated messages from the variable node to factor node in (6.29), $\Delta^{(t+1)}_{\Phi_m \to x_n}$ can then be computed:

$$\Delta^{(t+1)}_{\Phi_m \to x_n} = \text{const} + \log \int_{\boldsymbol{x} \backslash x_n} \Phi_m \left(y_m, \boldsymbol{x}, \hat{\boldsymbol{\theta}}^{(t)}_{\Phi_m} \right) \cdot \exp \left(\sum_{j \neq n} \Delta^{(t)}_{\Phi_m \leftarrow x_j} \right) . \tag{6.30}$$

Direct integration with respect to $\boldsymbol{x} \backslash x_n$ in (6.30) is quite difficult. If we go back to the original belief propagation, we can see that the message $\Delta^{(t+1)}_{\Phi_m \to x_n}$ essentially performs the following computation:

$$\begin{aligned} \log p(y_m, x_n) &= \log \int_{\boldsymbol{x} \backslash x_n, \boldsymbol{\theta}} p(y_m, \boldsymbol{x}, \theta) \\ &= \log \int_{\boldsymbol{x} \backslash x_n, \boldsymbol{\theta}} p(y_m | \boldsymbol{x}, \boldsymbol{\theta}) p(\boldsymbol{x}) p(\boldsymbol{\theta}) . \end{aligned} \tag{6.31}$$

Let $z'_m = z_m - a_{mn} x_n = \sum_{j \neq n} a_{mj} x_j$, $\log p(y_m, x_n)$ can also be written as:

$$\begin{aligned} \log p(y_m, x_n) &= \log \int_{z'_m, \boldsymbol{\theta}} p(y_m, x_n, z'_m, \boldsymbol{\theta}) \\ &= \log \int_{z'_m, \boldsymbol{\theta}} p(y_m | x_n, z'_m, \theta) p(z'_m) p(\boldsymbol{\theta}) . \end{aligned} \tag{6.32}$$

Translating (6.32) back to the message gives us:

$$\begin{aligned} \Delta^{(t+1)}_{\Phi_m \to x_n} = \text{const} + \log \int_{z'_m} \Bigg[& \Phi \left(y_m, x_n, z'_m, \hat{\boldsymbol{\theta}}^{(t)}_{\Phi_m} \right) \\ & \times \exp \Bigg(- \frac{1}{2 \left(\tau^q_{\Phi_m}(t) - a^2_{mn} \tau^x_{\Omega_n}(t) \right)} \\ & \times \left(z'_m - \left(q^{(t)}_{\Phi_m} - a_{mn} \hat{x}^{(t)}_{\Phi_m \leftarrow x_n} \right) \right)^2 \Bigg) \Bigg] , \end{aligned} \tag{6.33}$$

where $\tau^q_{\Phi_m}(t), q^{(t)}_{\Phi_m}$ are as follows:

$$\tau^q_{\Phi_m}(t) = \sum_j a^2_{mj}\tau^x_{\Omega_j}(t) \tag{6.34a}$$

$$q^{(t)}_{\Phi_m} = \sum_j a_{mj}\hat{x}^{(t)}_{\Phi_m \leftarrow x_j} . \tag{6.34b}$$

If a_{mn} is small, $a^2_{mn}\tau^x_{\Omega_n}(t)$ can be neglected. Since the integration of z'_m is from $-\infty$ to ∞, we replace z'_m with $z_m = z'_m + a_{mn}x_n$. (6.33) then becomes:

$$\begin{aligned}\Delta^{(t+1)}_{\Phi_m \to x_n} = \text{const} + \log \int_{z_m} \Bigg[& \Phi\left(y_m, z_m, \hat{\boldsymbol{\theta}}^{(t)}_{\Phi_m}\right) \times \\ & \exp\left(\frac{-1}{2\tau^q_{\Phi_m}(t)}\left(z_m - \left(q^{(t)}_{\Phi_m} + a_{mn}\left(x_n - \hat{x}^{(t)}_{\Phi_m \leftarrow x_n}\right)\right)\right)^2\right)\Bigg] .\end{aligned} \tag{6.35}$$

6.3.1 Review on sum-product GAMP update

For completeness, we include the GAMP update from [22] to compute $\Delta^{(t+1)}_{\Phi_m \to x_n}$, $\Delta^{(t+1)}_{\Phi_m \leftarrow x_n}$. The following function $H(q, \tau^q, y, \boldsymbol{\theta})$ is defined:

$$H(q, \tau^q, y, \boldsymbol{\theta}) = \log \int_z \Phi(y, z, \boldsymbol{\theta}) \cdot \exp\left(-\frac{1}{2\tau^q}(z-q)^2\right) . \tag{6.36}$$

$\Delta^{(t+1)}_{\Phi_m \to x_n}$ in (6.35) can then be written as:

$$\Delta^{(t+1)}_{\Phi_m \to x_n} = \text{const} + H\left(q^{(t)}_{\Phi_m} + a_{mn}\left(x_n - \hat{x}^{(t)}_{\Phi_m \leftarrow x_n}\right), \tau^q_{\Phi_m}(t), y_m, \hat{\boldsymbol{\theta}}^{(t)}_{\Phi_m}\right) . \tag{6.37}$$

Next, we try to approximate the message $\Delta^{(t+1)}_{\Phi_m \to x_n}$ up to second order Taylor series at $q^{(t)}_{\Phi_m}$. We define the following:

$$g_{\text{out}}(q, \tau^q, y, \boldsymbol{\theta}) := \frac{\partial}{\partial q} H(q, \tau^q, y, \boldsymbol{\theta}) . \tag{6.38}$$

Let $s^{(t)}_{\Phi_m}, \tau^s_{\Phi_m}(t)$ be the first and second order of $H(\cdot)$ at $q^{(t)}_{\Phi_m}$:

$$s^{(t)}_{\Phi_m} = g_{\text{out}}\left(t, q^{(t)}_{\Phi_m}, \tau^q_{\Phi_m}(t), y_m, \hat{\boldsymbol{\theta}}^{(t)}_{\Phi_m}\right) \tag{6.39}$$

$$\tau^s_{\Phi_m}(t) = -\frac{\partial}{\partial q} g_{\text{out}}\left(t, q^{(t)}_{\Phi_m}, \tau^q_{\Phi_m}(t), y_m, \hat{\boldsymbol{\theta}}^{(t)}_{\Phi_m}\right) . \tag{6.40}$$

$\Delta^{(t+1)}_{\Phi_m \to x_n}$ can then be approximated by:

$$\begin{aligned} \Delta^{(t+1)}_{\Phi_m \to x_n} \approx \text{const} &+ s^{(t)}_{\Phi_m} a_{mn}\left(x_n - \hat{x}^{(t)}_{\Phi_m \leftarrow x_n}\right) \\ &- \frac{\tau^s_{\Phi_m}(t)}{2} a^2_{mn}\left(x_n - \hat{x}^{(t)}_{\Phi_m \leftarrow x_n}\right)^2 . \end{aligned} \tag{6.41}$$

$\Delta^{(t+1)}_{\Phi_m \leftarrow x_n}$ will then be computed as is done in [22]:

$$\begin{aligned} \Delta^{(t+1)}_{\Phi_m \leftarrow x_n} \approx \text{const} &+ \Delta^{(t+1)}_{\Omega_n \to x_n} \\ &- \frac{1}{2\tau^r_{\Phi_m \leftarrow x_n}(t)}\left(r^{(t)}_{\Phi_m \leftarrow x_n} - x_n\right)^2 \end{aligned} \tag{6.42a}$$

$$\tau^r_{\Phi_m \leftarrow x_n}(t) = \left(\sum_{i \neq m} a^2_{in} \tau^s_{\Phi_i}(t)\right)^{-1} \tag{6.42b}$$

$$r^{(t)}_{\Phi_m \leftarrow x_n} = \hat{x}^{(t)}_{\Phi_m \leftarrow x_n} + \tau^r_{\Phi_m \leftarrow x_n}(t) \sum_{i \neq m} s^{(t)}_{\Phi_i} a_{in} . \tag{6.42c}$$

(6.42b, 6.42c) are approximated:

$$\tau^r_{\Phi_m \leftarrow x_n}(t) \approx \tau^r_{\Omega_n}(t) = \left(\sum_i a^2_{in} \tau^s_{\Phi_i}(t)\right)^{-1} \tag{6.43a}$$

$$\begin{aligned} r^{(t)}_{\Phi_m \leftarrow x_n} &\approx \left(\hat{x}^{(t)}_{\Phi_m} + \tau^r_{\Omega_n}(t) \sum_i s^{(t)}_{\Phi_i} a_{in}\right) - \tau^r_{\Omega_n}(t) a_{mn} s^{(t)}_{\Phi_m} \\ &= r^{(t)}_{\Omega_n} - \tau^r_{\Omega_n}(t) a_{mn} s^{(t)}_{\Phi_m} . \end{aligned} \tag{6.43b}$$

The following definition is also made in [22]:

$$g_{\text{in}}\left(r_{\Omega_n}^{(t)}, \tau_{\Omega_n}^{r}(t), \hat{\boldsymbol{\lambda}}_{\Omega_n}^{(t)}\right) := \frac{\int_{x_n} x_n \exp\left(\Delta_{\Omega_n \to x_n}^{(t)} - \frac{1}{2\tau_{\Omega_n}^{r}(t)}\left(r_{\Omega_n}^{(t)} - x_n\right)^2\right)}{\int_{x_n} \exp\left(\Delta_{\Omega_n \to x_n}^{(t)} - \frac{1}{2\tau_{\Omega_n}^{r}(t)}\left(r_{\Omega_n}^{(t)} - x_n\right)^2\right)}. \tag{6.44}$$

$\hat{x}_{\Phi_m \leftarrow x_n}^{(t+1)}, \hat{x}_n^{(t+1)}, q_{\Phi_m}^{(t+1)}$ are then [22]:

$$\hat{x}_{\Phi_m \leftarrow x_n}^{(t+1)} = g_{\text{in}}\left(r_{\Phi_m \leftarrow x_n}^{(t)}, \tau_{\Phi_m \leftarrow x_n}^{r}(t), \hat{\boldsymbol{\lambda}}_{\Omega_n}^{(t)}\right) \tag{6.45a}$$

$$\hat{x}_n^{(t+1)} = g_{\text{in}}\left(r_{\Omega_n}^{(t)}, \tau_{\Omega_n}^{r}(t), \hat{\boldsymbol{\lambda}}_{\Omega_n}^{(t)}\right) \tag{6.45b}$$

$$q_{\Phi_m}^{(t+1)} \approx \sum_j a_{mj} \hat{x}_j^{(t+1)} - \tau_{\Phi_m}^{q}(t) s_{\Phi_m}^{(t)}. \tag{6.45c}$$

6.3.2 Sum-product parameter update

Similarly we can compute the rest messages from the factor nodes to variable nodes in the proposed PE-GAMP using Dirac delta approximation of the messages involving the parameters:

$$\Delta_{\Omega_n \to x_n}^{(t+1)} = \text{const} + \log \Omega_n\left(x_n, \hat{\boldsymbol{\lambda}}_{\Omega_n}^{(t)}\right) \tag{6.46a}$$

$$\begin{aligned}\Delta_{\Omega_n \to \lambda_l}^{(t+1)} = \text{const} + \log \int_{x_n} \Bigg[& \Omega_n\left(x_n, \lambda_l, \hat{\boldsymbol{\lambda}}_{\Omega_n}^{(t)} \backslash \hat{\lambda}_{\Omega_n \leftarrow \lambda_l}^{(t)}\right) \\ & \times \exp\left(-\frac{1}{2\tau_{\Omega_n \leftarrow x_n}^{x}(t+1)}\left(x_n - \hat{x}_{\Omega_n \leftarrow x_n}^{(t+1)}\right)^2\right)\Bigg]\end{aligned} \tag{6.46b}$$

$$\begin{aligned}\Delta_{\Phi_m \to \theta_k}^{(t+1)} = \text{const} + \log \int_{z_m} \Bigg[& \Phi\left(y_m, z_m, \theta_k, \hat{\boldsymbol{\theta}}_{\Phi_m}^{(t)} \backslash \hat{\theta}_{\Phi_m \leftarrow \theta_k}^{(t)}\right) \\ & \times \exp\left(-\frac{1}{2\tau_{\Phi_m}^{q}(t)}\left(z_m - q_{\Phi_m}^{(t)}\right)^2\right)\Bigg].\end{aligned} \tag{6.46c}$$

We next compute (6.26) in the $(t+1)$-th iteration. Using (6.41) we can get $\Delta^{(t+1)}_{\Omega_n \leftarrow x_n}$ in (6.26c) first:

$$\Delta^{(t+1)}_{\Omega_n \leftarrow x_n} \approx -\frac{1}{2\tau^r_{\Omega_n}(t)} \left(x_n - r^{(t)}_{\Omega_n} \right)^2 . \tag{6.47}$$

(6.26a,6.26b) in the $(t+1)$-th iteration are then:

$$\hat{x}^{(t+1)}_{\Omega_n \leftarrow x_n} = r^{(t)}_{\Omega_n}, \qquad \tau^x_{\Omega_n \leftarrow x_n}(t+1) = \tau^r_{\Omega_n}(t) . \tag{6.48}$$

The parameters $\hat{\boldsymbol{\lambda}}^{(t+1)}_{\Omega_n}, \hat{\boldsymbol{\theta}}^{(t+1)}_{\Phi_m}$ in the $(t+1)$-th iteration can then be computed using (6.27) or (6.28).

6.4 Max-sum PE-GAMP

The approximated *max-sum* message passing also uses quadratic approximation of the messages. It is in many ways similar to the *sum-product* message passing presented previously in section 6.3. A few differences in do exists though. Specifically, the definitions in (6.24) are changed into:

$$\hat{x}^{(t)}_n := \arg\max_{x_n} \Delta^{(t)}_{x_n} \tag{6.49a}$$

$$\tau^x_{\Omega_n}(t) := -\left(\frac{\partial^2 \Delta^{(t)}_{x_n}}{\partial x_n^2} \Big|_{x_n = \hat{x}^{(t)}_n} \right)^{-1} \tag{6.49b}$$

$$\hat{x}^{(t)}_{\Phi_m \leftarrow x_n} := \arg\max_{x_n} \Delta^{(t)}_{\Phi_m \leftarrow x_n} \tag{6.49c}$$

$$\tau^x_{\Phi_m \leftarrow x_n}(t) := -\left(\frac{\partial^2 \Delta^{(t)}_{\Phi_m \leftarrow x_n}}{\partial x_n^2} \Big|_{x_n = \hat{x}^{(t)}_{\Phi_m \leftarrow x_n}} \right)^{-1} . \tag{6.49d}$$

In the proposed PE-GAMP, the parameters are computed as follows:

$$\hat{\lambda}^{(t)}_{\Omega_n\leftarrow\lambda_l} = \arg\max_{\lambda_l} \log\Omega_n\left(\hat{x}_n^{(t-1)}, \lambda_l, \hat{\boldsymbol{\lambda}}^{(t-1)}_{\Omega_n}\backslash\hat{\lambda}^{(t-1)}_{\Omega_n\leftarrow\lambda_l}\right) + \Delta^{(t-1)}_{\Omega_n\leftarrow\lambda_l} \tag{6.50a}$$

$$\hat{\theta}^{(t)}_{\Phi_m\leftarrow\theta_k} = \arg\max_{\theta_k} \log\Phi_m\left(y_m, \hat{\boldsymbol{x}}^{(t-1)}, \theta_k, \hat{\boldsymbol{\theta}}^{(t-1)}_{\Phi_m}\backslash\hat{\theta}^{(t-1)}_{\Phi_m\leftarrow\theta_k}\right) + \Delta^{(t-1)}_{\Phi_m\leftarrow\theta_k}. \tag{6.50b}$$

6.4.1 Review on max-sum GAMP update

The definitions of $H(q,\tau^q,y,\boldsymbol{\theta})$ and the input function $g_{\text{in}}(\cdot)$ are also different from *sum-product* message passing. [22] has the following definitions:

$$H(q,\tau^q,y) = \max_z\left[\log\Phi(y,z,\boldsymbol{\theta}) - \frac{1}{2\tau^q}(z-q)^2\right] \tag{6.51}$$

$$g_{\text{out}}\left(r^{(t)}_{\Omega_n}, \tau^r_{\Omega_n}(t), \hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n}\right) = \arg\max_{x_n}\left[\Delta^{(t)}_{\Omega_n\to x_n} - \frac{1}{2\tau^r_{\Omega_n}(t)}\left(r^{(t)}_{\Omega_n} - x_n\right)^2\right]. \tag{6.52}$$

The message $\Delta^{(t+1)}_{\Phi_m\to x_n}$ is also different from (6.35) in section 6.3. In [22], it is given as follows:

$$\begin{aligned}\Delta^{(t+1)}_{\Phi_m\to x_n} \approx \max_{z_m}\Bigg[&\log\Phi\left(z_m, y_m, \hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n}\right)\\ &-\frac{1}{2\tau^q_{\Phi_m}(t)}\left(z_m - \left(q^{(t)}_{\Phi_m} + a_{mn}\left(x_n - \hat{x}^{(t)}_{\Phi_m\leftarrow x_n}\right)\right)\right)^2\Bigg].\end{aligned} \tag{6.53}$$

6.4.2 Max-sum parameter update

The messages in (6.46) are also updated:

$$\Delta^{(t+1)}_{\Omega_n \to x_n} = \log \Omega_n(x_n, \hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n}) \tag{6.54a}$$

$$\begin{aligned} \Delta^{(t+1)}_{\Omega_n \to \lambda_l} = \max_{x_n} \Bigg[& \log \Omega_n \left(x_n, \lambda_l, \hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n} \backslash \hat{\lambda}^{(t)}_{\Omega_n \leftarrow \lambda_l} \right) \\ & - \frac{1}{2\tau^r_{\Omega_n}(t)} \left(x_n - r^{(t)}_{\Omega_n} \right)^2 \Bigg] \end{aligned} \tag{6.54b}$$

$$\begin{aligned} \Delta^{(t+1)}_{\Phi_m \to \theta_k} = \max_{z_m} \Bigg[& \log \Phi \left(y_m, z_m, \theta_k \hat{\boldsymbol{\theta}}^{(t)}_{\Phi_m} \backslash \hat{\theta}^{(t)}_{\Phi_m \leftarrow \theta_k} \right) \\ & - \frac{1}{2\tau^q_{\Phi_m}(t)} \left(z_m - q^{(t)}_{\Phi_m} \right)^2 \Bigg] . \end{aligned} \tag{6.54c}$$

The parameters $\hat{\boldsymbol{\lambda}}^{(t+1)}_{\Omega_n}, \hat{\boldsymbol{\theta}}^{(t+1)}_{\Phi_m}$ in the $(t+1)$-th iteration can then be computed using (6.50).

Chapter 7

Empirical convergence analysis of the PE-GAMP

In this chapter we study the empirical convergence behavior of various variables introduced in the proposed PE-GAMP algorithm through its state evolution analysis in the large system limit as $N \to \infty$. The following analysis is built upon prior work in [22, 62, 66]. We show that the estimated parameters $\{\hat{\boldsymbol{\lambda}}_{\Omega_n}, \hat{\boldsymbol{\theta}}_{\Phi_m}\}$ eventually converge to the scalars $\{\overline{\boldsymbol{\lambda}}_{\Omega_n}, \overline{\boldsymbol{\theta}}_{\Phi_m}\}$ in (7.18), and the entries of the estimated signal $\hat{\boldsymbol{x}}$ are able to achieve the empirical convergence defined in Definition 4.

7.1 Review on the state evolution analysis of the GAMP

We first introduce the definitions as well as assumptions used in the state evolution (SE) analysis [22] that studies the empirical convergence behavior of the variables in the large system limit. It is a minor modification of the work from [62].

Definition 3. *A function $g(\cdot) : \mathbb{R}^r \to \mathbb{R}^s$ is* pseudo-Lipschitz *of order $k > 1$, if there* *exists an* $L > 0$ *such that* $\forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^r$,

$$\|g(\boldsymbol{x}) - g(\boldsymbol{y})\| \leq L(1 + \|\boldsymbol{x}\|^{k-1} + \|\boldsymbol{y}\|^{k-1})\|\boldsymbol{x} - \boldsymbol{y}\| \,. \tag{7.1}$$

Definition 4. *Suppose* $\{\boldsymbol{v}^{[N]} \in \mathbb{R}^{sl_N}, N = 1, 2, \cdots\}$ *is a sequence of vectors, and each* $\boldsymbol{v}^{[N]}$ *contains* l_N *blocks of vector components* $\{\boldsymbol{v}_i^{[N]} \in \mathbb{R}^s, i = 1, \cdots, l_N\}$. *The components of* $\boldsymbol{v}^{[N]}$ empirically converges with bounded moments of order k *to a random vector* $\boldsymbol{v} \in \mathbb{R}^s$ *as* $N \to \infty$ *if: For all pseudo-Lipschitz continuous functions* $g(\cdot)$ *of order k,*

$$\lim_{N\to\infty} \frac{1}{l_N} \sum_{i=1}^{l_N} g\left(\boldsymbol{v}_i^{[N]}\right) = \mathbb{E}\left[g(\boldsymbol{v})\right] < \infty \,. \tag{7.2}$$

When the nature of convergence is clear, it can be simply written as follows:

$$\lim_{N\to\infty} \boldsymbol{v}_i^{[N]} \overset{\mathrm{PL}(k)}{=} \boldsymbol{v} \,. \tag{7.3}$$

Based on the above pseudo-Lipschitz continuity and empirical convergence definitions, GAMP also makes the following assumptions about the estimation of $\boldsymbol{x} \in \mathbb{R}^N$ [22, 62].

Assumption 1. *The GAMP solves a series of estimation problems indexed by the input signal dimension N:*

a) The output dimension M is deterministic and scales linearly with the input dimension N: $\lim_{N\to\infty} \frac{N}{M} = \beta$ *for some* $\beta > 0$.

b) The matrix $\boldsymbol{A} \in \mathbb{R}^{M\times N}$ *has i.i.d Gaussian entries* $A_{ij} \sim \mathcal{N}(0, \frac{1}{M})$.

c) The components of initial condition $\hat{\boldsymbol{x}}^{(0)}, \tau_\Omega^x(0)$ *and the input signal* $\boldsymbol{x}$ *empirically* *converge with bounded moments of order* $2k-2$ *as follows:*

$$\lim_{N\to\infty}(\hat{x}_n^{(0)}, x_n) \overset{\mathrm{PL}(2k-2)}{=} (\hat{\mathcal{X}}^{(0)}, \mathcal{X}) \tag{7.4a}$$

$$\lim_{N\to\infty}\tau_{\Omega_n}^x(0) = \overline{\tau}_{\Omega}^x(0)\,. \tag{7.4b}$$

d) The output vector $\boldsymbol{y}\in\mathbb{R}^M$ *depends on the transform output* $\boldsymbol{z}=\boldsymbol{A}\boldsymbol{x}\in\mathbb{R}^M$ *and the noise vector* $\boldsymbol{w}\in\mathbb{R}^M$ *through some function* $g(\cdot)$*. For* $\forall m=1,\cdots,M$*,*

$$y_m = g(z_m, w_m)\,. \tag{7.5}$$

w_m *empirically converges with bounded moments of order* $2k-2$ *to some random variable* $\mathcal{W}\in\mathbb{R}$ *with distribution* $p(w)$*. The conditional distribution of* $\mathcal{Y}$ *given* $\mathcal{Z}$ *is given by* $p(y|z)$*.*

e) The channel estimation functions $g_{in}(\cdot)$*,* $g_{out}(\cdot)$ *and their partial derivatives with respect to* r,q,z *exist almost everywhere and are pseudo-Lipschitz continuous of order* k*.*

The SE equations of the GAMP describe the limiting behavior of the following scalar random variables and scalar variances as $N\to\infty$:

$$\boldsymbol{\psi}_{\mathrm{in}} := \{(x_n, \hat{x}_n^{(t+1)}, r_{\Omega_n}^{(t)}),\ n=1,\cdots,N\} \tag{7.6a}$$

$$\boldsymbol{\psi}_{\mathrm{out}} := \{(z_m, \hat{z}_m^{(t)}, y_m, q_{\Phi_m}^{(t)}),\ m=1,\cdots,M\} \tag{7.6b}$$

$$\boldsymbol{\psi}_{\tau} := (\tau_{\Phi}^q, \tau_{\Omega}^r)\,. \tag{7.6c}$$

[22] showed that (7.6a-7.6b) empirically converge with bounded moments of

order k to the following random vectors:

$$\lim_{N\to\infty} \boldsymbol{\psi}_{\text{in}} \overset{\text{PL}(k)}{=} \overline{\boldsymbol{\psi}}_{\text{in}} := (\mathcal{X}, \hat{\mathcal{X}}^{(t+1)}, \mathcal{R}_{\Omega}^{(t)}) \tag{7.7a}$$

$$\lim_{N\to\infty} \boldsymbol{\psi}_{\text{out}} \overset{\text{PL}(k)}{=} \overline{\boldsymbol{\psi}}_{\text{out}} := (\mathcal{Z}, \hat{\mathcal{Z}}^{(t)}, \mathcal{Y}, \mathcal{Q}_{\Phi}^{(t)})\,, \tag{7.7b}$$

where $\mathcal{R}_{\Omega}^{(t)}, \mathcal{Z}, \mathcal{Q}_{\Phi}^{(t)}$ are as follows for some computed $\alpha^r \in \mathbb{R}$, $\xi^r \in \mathbb{R}$, $\boldsymbol{K}^q \in \mathbb{R}^{2\times 2}$:

$$\mathcal{R}_{\Omega}^{(t)} = \alpha^r \mathcal{X} + \mathcal{V}, \qquad \mathcal{V} \sim \mathcal{N}(0, \xi^r) \tag{7.8a}$$

$$(\mathcal{Z}, \mathcal{Q}_{\Phi}^{(t)}) \sim \mathcal{N}(0, \boldsymbol{K}^q)\,. \tag{7.8b}$$

Additionally, for $\boldsymbol{\psi}_{\tau}$, the following convergence holds:

$$\lim_{N\to\infty} \boldsymbol{\psi}_{\tau} = \overline{\boldsymbol{\psi}}_{\tau} := (\overline{\tau}_{\Phi}^{q}, \overline{\tau}_{\Omega}^{r})\,. \tag{7.9}$$

In [66], the adaptive-GAMP is proposed to perform state evolution analysis of the EM based parameter estimation methods. Specifically, the following Assumption 2 can be established on the estimation problem:

Assumption 2. *The adaptive-GAMP with parameter estimation solves a series of estimation problem indexed by the input signal dimension N:*

a) *Assumptions 1(a) to 1(d) with $k = 2$.*
b) *Assumption 5(b).*
c) *For every t, the estimation (adaptation) function $f_{\lambda}(t, \boldsymbol{r}_{\Omega}^{(t)}, \tau_{\Omega}^{r}(t))$ can be considered as a function of $\boldsymbol{r}_{\Omega}^{(t)}$ that satisfies the weak pseudo-Lipschitz continuity property: If the sequence of vector $\boldsymbol{r}_{\Omega}^{(t)}$ indexed by N empirically converges with bounded moments of order $k = 2$ and the sequence of scalars $\tau_{\Omega}^{r}(t)$ converge as*

follows:

$$\lim_{N\to\infty} \boldsymbol{r}_{\Omega}^{(t)} \overset{\mathrm{PL}(k)}{=} \mathcal{R}_{\Omega}^{(t)}, \quad \lim_{M\to\infty} \tau_{\Omega}^{r}(t) = \overline{\tau}_{\Omega}^{r}(t)\,. \tag{7.10}$$

Then,

$$\lim_{N\to\infty} f_{\boldsymbol{\lambda}}(t, \boldsymbol{r}_{\Omega}^{(t)}, \tau_{\Omega}^{r}(t)) = f_{\boldsymbol{\lambda}}(t, \mathcal{R}_{\Omega}^{(t)}, \overline{\tau}_{\Omega}^{r}(t))\,. \tag{7.11}$$

Similarly $f_{\boldsymbol{\theta}}(t, \boldsymbol{q}_{\Phi}^{(t)}, \boldsymbol{y}, \tau_{\Phi}^{q}(t))$ also satisfies the weak pseudo-Lipschitz continuity property.

Theorem 4 is then given to describe the limiting behavior of the scalar variables in the adaptive-GAMP algorithm [66].

Theorem 4. *Consider the adaptive-GAMP with scalar variances under the Assumption 2. $\forall t$, the components of the following sets of scalars empirically converges with bounded moments of order $k = 2$:*

$$\lim_{N\to\infty} \boldsymbol{\psi}_{in} \overset{\mathrm{PL}(k)}{=} \overline{\boldsymbol{\psi}}_{in}, \quad \lim_{N\to\infty} \boldsymbol{\psi}_{out} \overset{\mathrm{PL}(k)}{=} \overline{\boldsymbol{\psi}}_{out} \tag{7.12a}$$

$$\lim_{N\to\infty} \boldsymbol{\psi}_{\tau} = \overline{\boldsymbol{\psi}}_{\tau} \tag{7.12b}$$

$$\lim_{N\to\infty} \hat{\boldsymbol{\theta}}^{(t+1)} = \overline{\boldsymbol{\theta}}^{(t+1)}, \quad \lim_{N\to\infty} \hat{\boldsymbol{\lambda}}^{(t+1)} = \overline{\boldsymbol{\lambda}}^{(t+1)}\,. \tag{7.12c}$$

7.2 The state evolution analysis of the PE-GAMP

The SE equations of the proposed PE-GAMP are given in Algorithm 4. In addition to (7.6a-7.6c), the state evolution (SE) analysis of PE-GAMP will study the limiting behavior of $\hat{\boldsymbol{\lambda}}_{\Omega_n}^{(t+1)}, \hat{\boldsymbol{\theta}}_{\Phi_m}^{(t+1)}$ for each $n = 1, \cdots, N$ and $m = 1, \cdots, M$.

Eventually we would like to show that they empirically converge to the

Algorithm 4 The PE-GAMP state evolution

```
Require: The matrix $\boldsymbol{A} \in \mathbb{R}^{M\times N}$; the observation $\boldsymbol{y} \in \mathbb{R}^{M}$; the input and output channels estimation functions $g_{\text{in}}(\cdot), g_{\text{out}}(\cdot)$; the parameter estimation functions $f_{\theta_k}(\cdot), f_{\lambda_l}(\cdot)$.
Initialize $\overline{\tau}^x_\Omega(0), \overline{\boldsymbol{\theta}}^{(0)}, \overline{\boldsymbol{\lambda}}^{(0)}$ and set
    $\boldsymbol{K}^x(0) = \text{cov}\left(\mathcal{X}, \hat{\mathcal{X}}^{(0)}\right) .$    (7.13)
for $t = \{0, 1, \cdots\}$ do
    Output channel update:
        $\overline{\tau}^q_\Phi(t) = \beta\overline{\tau}^x_\Omega(t), \qquad \boldsymbol{K}^q(t) = \beta\boldsymbol{K}^x(t)$    (7.14a)
        $\overline{\tau}^r_\Omega(t) = -\mathbb{E}^{-1}\left[\frac{\partial}{\partial q} g_{\text{out}}\left(t, \mathcal{Q}^{(t)}_\Phi, \overline{\tau}^q_\Phi(t), \mathcal{Y}, \overline{\boldsymbol{\theta}}^{(t)}_{\Phi_m}\right)\right]$    (7.14b)
        $\xi^r(t) = \left(\overline{\tau}^r_\Omega(t)\right)^2 \mathbb{E}\left[g_{\text{out}}\left(t, \mathcal{Q}^{(t)}_\Phi, \overline{\tau}^q_\Phi(t), \mathcal{Y}, \overline{\boldsymbol{\theta}}^{(t)}_{\Phi_m}\right)\right]$    (7.14c)
        $\alpha^r(t) = \overline{\tau}^r_\Omega(t)\mathbb{E}\left[\frac{\partial}{\partial z} g_{\text{out}}\left(t, \mathcal{Q}^{(t)}_\Phi, \overline{\tau}^q_\Phi(t), g(\mathcal{Z}, \mathcal{W}), \overline{\boldsymbol{\theta}}^{(t)}_{\Phi_m}\right)\right] ,$    (7.14d)
    where the expectations are over the random variables $\mathcal{Z}, \mathcal{Q}^{(t)}_\Phi, \mathcal{W}, \mathcal{Y}$.
    Input channel update:
        $\hat{\mathcal{X}}^{(t+1)} = g_{\text{in}}\left(t, \mathcal{R}^{(t)}_\Omega, \overline{\tau}^r_\Omega(t), \overline{\boldsymbol{\lambda}}^{(t)}_{\Omega_n}\right)$    (7.15a)
        $\overline{\tau}^x_\Omega(t+1) = \overline{\tau}^r_\Omega(t)\mathbb{E}\left[\frac{\partial}{\partial r} g_{\text{in}}\left(t, \mathcal{R}^{(t)}_\Omega, \overline{\tau}^r_\Omega(t), \overline{\boldsymbol{\lambda}}^{(t)}_{\Omega_n}\right)\right]$    (7.15b)
        $\boldsymbol{K}^x(t+1) = \text{cov}\left(\mathcal{X}, \hat{\mathcal{X}}^{(t+1)}\right) ,$    (7.15c)
    where the expectation is over the random variables $\mathcal{X}, \mathcal{R}^{(t)}_\Omega$.
```

Algorithm 4 The PE-GAMP state evolution (continued)

```
    Sum-product message passing parameters update: For each $k = 1, \cdots, K$ and $l = 1, \cdots, L$
        $$\overline{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l} = f_{\Omega_n \leftarrow \lambda_l}\left(t, \mathcal{R}^{(t)}_{\Omega}, \overline{\tau}^{r}_{\Omega}(t), \lambda_l, \overline{\boldsymbol{\lambda}}^{(t)}_{\Omega_n} \backslash \overline{\lambda}^{(t)}_{\Omega_n \leftarrow \lambda_l}\right) \tag{7.16a}$$
        $$\overline{\theta}^{(t+1)}_{\Phi_m \leftarrow \theta_k} = f_{\Phi_m \leftarrow \theta_k}\left(t, \mathcal{Q}^{(t)}_{\Phi}, \mathcal{Y}, \overline{\tau}^{q}_{\Phi}(t), \theta_k, \overline{\boldsymbol{\theta}}^{(t)}_{\Phi_m} \backslash \overline{\theta}^{(t)}_{\Phi_m \leftarrow \theta_k}\right) . \tag{7.16b}$$
    Max-sum message passing parameters update: For each $k = 1, \cdots, K$ and $l = 1, \cdots, L$
        $$\overline{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l} = f_{\Omega_n \leftarrow \lambda_l}\left(t, \hat{\mathcal{X}}^{(t)}, \mathcal{R}^{(t)}_{\Omega}, \overline{\tau}^{r}_{\Omega}(t), \lambda_l, \overline{\boldsymbol{\lambda}}^{(t)}_{\Omega_n} \backslash \overline{\lambda}^{(t)}_{\Omega_n \leftarrow \lambda_l}\right) \tag{7.17a}$$
        $$\overline{\theta}^{(t+1)}_{\Phi_m \leftarrow \theta_k} = f_{\Phi_m \leftarrow \theta_k}\left(t, \hat{\mathcal{Z}}^{(t)}, \mathcal{Q}^{(t)}_{\Phi}, \mathcal{Y}, \overline{\tau}^{q}_{\Phi}(t), \theta_k, \overline{\boldsymbol{\theta}}^{(t)}_{\Phi_m} \backslash \overline{\theta}^{(t)}_{\Phi_m \leftarrow \theta_k}\right) . \tag{7.17b}$$
    if $\hat{\mathcal{X}}^{(t+1)}$ reaches convergence then
        $\hat{\mathcal{X}} = \hat{\mathcal{X}}^{(t+1)}$;
        break;
    end if
end for
Return Output $\hat{\mathcal{X}}$;
```

following random vectors for fixed t as $N \to \infty$:

$$\overline{\boldsymbol{\lambda}}_{\Omega_n}^{(t+1)} = \left\{ \overline{\lambda}_{\Omega_n \leftarrow \lambda_l}^{(t+1)} \middle| \, l = 1, \cdots, L \right\} \tag{7.18a}$$

$$\overline{\boldsymbol{\theta}}_{\Phi_m}^{(t+1)} = \left\{ \overline{\theta}_{\Phi_m \leftarrow \theta_k}^{(t+1)} \middle| \, k = 1, \cdots, K \right\} . \tag{7.18b}$$

To simplify notations, we assume the following for the *sum-product* message passing:

$$h_{\Omega_n \leftarrow \lambda_l}^{\Omega_j}(\cdot) = \Delta_{\Omega_j \to \lambda_l}^{(t)} + \frac{1}{N-1} \log p(\lambda_l) \tag{7.19a}$$

$$h_{\Phi_m \leftarrow \theta_k}^{\Phi_i}(\cdot) = \Delta_{\Phi_i \to \theta_k}^{(t)} + \frac{1}{M-1} \log p(\theta_k) \, . \tag{7.19b}$$

For *max-sum* message passing, we assume:

$$\begin{aligned} h_{\Omega_n \leftarrow \lambda_l}^{\Omega_j}(\cdot) &= \Delta_{\Omega_j \to \lambda_l}^{(t)} \\ &+ \frac{1}{N-1} \left(\log p(\lambda_l) + \log \Omega_n \left(\hat{x}_n^{(t)}, \lambda_l, \hat{\boldsymbol{\lambda}}_{\Omega_n}^{(t)} \backslash \hat{\lambda}_{\Omega_n \leftarrow \lambda_l}^{(t)} \right) \right) \end{aligned} \tag{7.20a}$$

$$\begin{aligned} h_{\Phi_m \leftarrow \theta_k}^{\Phi_i}(\cdot) &= \Delta_{\Phi_i \to \theta_k}^{(t)} \\ &+ \frac{1}{M-1} \left(\log p(\theta_k) + \log \Phi_m \left(y_m, \hat{\mathbf{x}}^{(t)}, \theta_k, \hat{\boldsymbol{\theta}}_{\Phi_m}^{(t)} \backslash \hat{\theta}_{\Phi_m \leftarrow \theta_k}^{(t)} \right) \right) . \end{aligned} \tag{7.20b}$$

Since the parameter estimation of the *max-sum* message passing and the MAP parameter estimation of the *sum-product* message passing basically have the same form given in (7.21), their state evolution analysis can be derived similarly. For the sake of conciseness, we will only give the empirical convergence proofs for the MAP and MMSE parameter estimations of the *sum-product* message passing.

7.2.1 MAP parameter estimation state evolution

We can also write the estimation functions as follows:

$$\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l} = \arg\max_{\lambda_l} \frac{1}{N-1} \textstyle\sum_{j \neq n} h^{\Omega_j}_{\Omega_n \leftarrow \lambda_l}(\cdot) \tag{7.21a}$$

$$\hat{\theta}^{(t+1)}_{\Phi_m \leftarrow \theta_k} = \arg\max_{\theta_k} \frac{1}{M-1} \textstyle\sum_{i \neq m} h^{\Phi_i}_{\Phi_m \leftarrow \theta_k}(\cdot)\,. \tag{7.21b}$$

In the large system limit $N \to \infty$, the state evolution equations (7.16) of the parameters update step in *sum-product* message passing can then be written as:

$$\begin{aligned}\overline{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l} &= f_{\Omega_n \leftarrow \lambda_l}(\cdot) \\ &= \arg\max_{\lambda_l} \mathbb{E}\left[h^{\Omega_j}_{\Omega_n \leftarrow \lambda_l}\left(t, \mathcal{R}^{(t)}_{\Omega}, \overline{\tau}^{r}_{\Omega}(t), \lambda_l, \overline{\boldsymbol{\lambda}}^{(t)}_{\Omega_n} \backslash \overline{\lambda}^{(t)}_{\Omega_n \leftarrow \lambda_l}\right)\right]\end{aligned} \tag{7.22a}$$

$$\begin{aligned}\overline{\theta}^{(t+1)}_{\Phi_m \leftarrow \theta_k} &= f_{\Phi_m \leftarrow \theta_k}(\cdot) \\ &= \arg\max_{\theta_k} \mathbb{E}\left[h^{\Phi_i}_{\Phi_m \leftarrow \theta_k}\left(t, \mathcal{Q}^{(t)}_{\Phi}, \mathcal{Y}, \overline{\tau}^{q}_{\Phi}(t), \theta_k, \overline{\boldsymbol{\theta}}^{(t)}_{\Phi_m} \backslash \overline{\theta}^{(t)}_{\Phi_m \leftarrow \theta_k}\right)\right]\,,\end{aligned} \tag{7.22b}$$

where the expectations are over the random variables $\mathcal{R}^{(t)}_{\Omega}$ and $\left\{\mathcal{Q}^{(t)}_{\Phi}, \mathcal{Y}\right\}$ respectively.

Our proof of the convergence of the scalars in (6.13,7.6) will make use of the Theorem 4 from [66]. First, we give the following adapted assumptions for the MAP parameter estimation.

Assumption 3. *The priors on the parameters:* $\{p(\boldsymbol{\lambda}), \boldsymbol{\lambda} \in \mathcal{U}_{\lambda}\}$, $\{p(\boldsymbol{\theta}), \boldsymbol{\theta} \in \mathcal{U}_{\theta}\}$ *and the parameter estimation functions should satisfy:*

a) The priors $p(\boldsymbol{\lambda}) < \infty$, $p(\boldsymbol{\theta}) < \infty$ *are bounded, and the sets* $\mathcal{U}_{\lambda}, \mathcal{U}_{\theta}$ *are compact.*

b) For the sum-product *message passing, the following estimations are well-defined,* *unique.*

$$\begin{aligned}
&\lambda^*_{\Omega_n \leftarrow \lambda_l} = \\
&\arg\max_{\lambda_l \in \mathcal{U}_\lambda} \mathbb{E}\left[h^{\Omega_j}_{\Omega_n \leftarrow \lambda_l}(t, \mathcal{R}^{(t)}_{\Omega}, \tau^r_\Omega(t), \lambda_l, \hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n} \backslash \hat{\lambda}^{(t)}_{\Omega_n \leftarrow \lambda_l})\right]
\end{aligned} \tag{7.23a}$$

$$\begin{aligned}
&\theta^*_{\Phi_m \leftarrow \theta_k} = \\
&\arg\max_{\theta_k \in \mathcal{U}_\theta} \mathbb{E}\left[h^{\Phi_i}_{\Phi_m \leftarrow \theta_k}(t, \mathcal{Q}^{(t)}_{\Phi}, \mathcal{Y}, \tau^q_\Phi(t), \theta_k, \hat{\boldsymbol{\theta}}^{(t)}_{\Phi_m} \backslash \hat{\theta}^{(t)}_{\Phi_m \leftarrow \theta_k})\right],
\end{aligned} \tag{7.23b}$$

where the expectations are with respect to $\mathcal{R}^{(t)}_{\Omega}$ *and* $\left\{\mathcal{Q}^{(t)}_{\Phi}, \mathcal{Y}\right\}$.

c) $h^{\Omega_j}_{\Omega_n \leftarrow \lambda_l}(\cdot)$ *is pseudo-Lipschitz continuous of order* 2 *in* r_{Ω_n}*, it is also continuous in* λ_l *uniformly over* r_{Ω_n} *in the following sense: For every* $\epsilon > 0, \widetilde{\tau}^r_\Omega, \widetilde{\boldsymbol{\lambda}} \in \mathcal{U}_\lambda$*, there exists an open neighborhood* $\rho(\widetilde{\tau}^r_\Omega, \widetilde{\boldsymbol{\lambda}})$ *of* $(\widetilde{\tau}^r_\Omega, \widetilde{\boldsymbol{\lambda}} \in \mathcal{U}_\lambda)$*, such that* $\forall (\tau^r_\Omega, \boldsymbol{\lambda}) \in \rho(\widetilde{\tau}^r_\Omega, \widetilde{\boldsymbol{\lambda}})$ *and all* r,

$$\left| h^{\Omega_j}_{\Omega_n \leftarrow \lambda_l}(t, r_{\Omega_n}, \tau^r_\Omega, \boldsymbol{\lambda}) - h^{\Omega_j}_{\Omega_n \leftarrow \lambda_l}(t, r_{\Omega_n}, \widetilde{\tau}^r_\Omega, \widetilde{\boldsymbol{\lambda}}) \right| < \epsilon\,. \tag{7.24}$$

d) $h^{\Phi_i}_{\Phi_m \leftarrow \theta_k}(\cdot)$ *is pseudo-Lipschitz continuous of order* 2 *in* (q_{Φ_m}, y_m)*, it is also continuous in* θ_k *uniformly over* q_{Φ_m} *and* y_m.

7.2.2 MMSE parameter estimation state evolution

For the MMSE parameter estimation, the estimation functions can be written as follows:

$$\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l} = \int_{\lambda_l} \lambda_l \frac{\exp(\frac{1}{N-1}\sum_{j \neq n} h^{\Omega_j}_{\Omega_n \leftarrow \lambda_l}(\cdot))}{\int_{\lambda_l} \exp(\frac{1}{N-1}\sum_{j \neq n} h^{\Omega_j}_{\Omega_n \leftarrow \lambda_l}(\cdot))} \tag{7.25a}$$

$$\hat{\theta}^{(t+1)}_{\Phi_m \leftarrow \theta_k} = \int_{\theta_k} \theta_k \frac{\exp(\frac{1}{M-1}\sum_{i \neq m} h^{\Phi_i}_{\Phi_m \leftarrow \theta_k}(\cdot))}{\int_{\theta_k} \exp(\frac{1}{M-1}\sum_{i \neq m} h^{\Phi_i}_{\Phi_m \leftarrow \theta_k}(\cdot))}\,. \tag{7.25b}$$

The state evolution equations (7.16) of the parameters update step in Algorithm 4 can then be written as:

$$\overline{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l} = f_{\Omega_n \leftarrow \lambda_l}(\cdot) = \int_{\lambda_l} \lambda_l \frac{\exp(\mathbb{E}\left[h^{\Omega_j}_{\Omega_n \leftarrow \lambda_l}(\cdot)\right])}{\int_{\lambda_l} \exp(\mathbb{E}\left[h^{\Omega_j}_{\Omega_n \leftarrow \lambda_l}(\cdot)\right])} \tag{7.26a}$$

$$\overline{\theta}^{(t+1)}_{\Phi_m \leftarrow \theta_k} = f_{\Phi_m \leftarrow \theta_k}(\cdot) = \int_{\theta_k} \theta_k \frac{\exp(\mathbb{E}\left[h^{\Phi_i}_{\Phi_m \leftarrow \theta_k}(\cdot)\right])}{\int_{\theta_k} \exp(\mathbb{E}\left[h^{\Phi_i}_{\Phi_m \leftarrow \theta_k}(\cdot)\right])}, \tag{7.26b}$$

where the expectations are over the random variables $\mathcal{R}^{(t)}_{\Omega}$ and $\left\{\mathcal{Q}^{(t)}_{\Phi}, \mathcal{Y}\right\}$. To prove the convergence, we assume the following adapted assumptions for MMSE parameter estimation.

Assumption 4. *The priors on the parameters:* $\{p(\boldsymbol{\lambda}), \boldsymbol{\lambda} \in \mathcal{U}_\lambda\}$, $\{p(\boldsymbol{\theta}), \boldsymbol{\theta} \in \mathcal{U}_\theta\}$ *and the parameter estimation functions should satisfy:*

a) Assumption 3(a).

b) Assumption 3(c).

c) Assumption 3(d).

7.3 Empirical convergence analysis

We next give the following Lemma 5 about the estimation functions $f_{\Omega_n \leftarrow \lambda_l}(\cdot)$, $f_{\Phi_m \leftarrow \theta_k}(\cdot)$ for the proposed PE-GAMP:

Lemma 5. *Under Assumption 3 for MAP parameter estimation and Assumption 4 for MMSE parameter estimation, the estimation functions* $f_{\Omega_n \leftarrow \lambda_l}\left(t, \boldsymbol{r}^{(t)}_{\Omega}, \tau^r_{\Omega}(t), \lambda_l, \hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n} \backslash \hat{\lambda}^{(t)}_{\Omega_n \leftarrow \lambda_l}\right)$ *can be considered as a function of* $\boldsymbol{r}^{(t)}_{\Omega}$ *that satisfies the weak pseudo-Lipschitz continuity property: If the sequence of vector* $\boldsymbol{r}^{(t)}_{\Omega}$ *indexed by N empirically converges with bounded moments of order* $k = 2$ *and the* *sequence of scalers* $\tau^r_\Omega(t)$, $\hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n} \backslash \hat{\lambda}^{(t)}_{\Omega_n \leftarrow \lambda_l}$ *also converge as follows:*

$$\lim_{N\to\infty} \boldsymbol{r}^{(t)}_\Omega \overset{\mathrm{PL}(k)}{=} \mathcal{R}^{(t)}_\Omega \tag{7.27a}$$

$$\lim_{N\to\infty} \tau^r_\Omega(t) = \overline{\tau}^r_\Omega(t) \tag{7.27b}$$

$$\lim_{N\to\infty} \hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n} \backslash \hat{\lambda}^{(t)}_{\Omega_n \leftarrow \lambda_l} = \overline{\boldsymbol{\lambda}}^{(t)}_{\Omega_n} \backslash \overline{\lambda}^{(t)}_{\Omega_n \leftarrow \lambda_l} \ . \tag{7.27c}$$

Then,

$$\begin{aligned} &\lim_{N\to\infty} f_{\Omega_n \leftarrow \lambda_l} \left(t, \boldsymbol{r}^{(t)}_\Omega, \tau^r_\Omega(t), \lambda_l, \hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n} \backslash \hat{\lambda}^{(t)}_{\Omega_n \leftarrow \lambda_l}\right) \\ &= f_{\Omega_n \leftarrow \lambda_l} \left(t, \mathcal{R}^{(t)}_\Omega, \overline{\tau}^r_\Omega(t), \lambda_l, \overline{\boldsymbol{\lambda}}^{(t)}_{\Omega_n} \backslash \overline{\lambda}^{(t)}_{\Omega_n \leftarrow \lambda_l}\right) \ . \end{aligned} \tag{7.28}$$

Similarly, $f_{\Phi_m \leftarrow \theta_k} \left(t, \boldsymbol{q}^{(t)}_\Phi, \boldsymbol{y}, \tau^q_\Phi(t), \theta_k, \hat{\boldsymbol{\theta}}^{(t)}_{\Phi_m} \backslash \hat{\theta}^{(t)}_{\Phi_m \leftarrow \theta_k}\right)$ *also satisfies the weak pseudo-Lipschitz continuity property.*

Proof. Here we give the proof for $f_{\Omega_n \leftarrow \lambda_l}(\cdot)$, the proof for $f_{\Phi_m \leftarrow \theta_k}(\cdot)$ can be derived similarly.

1. **MAP Parameter Estimation:** The proof of the continuity of $f_{\Omega_n \leftarrow \lambda_l}(\cdot)$, $f_{\Phi_m \leftarrow \theta_k}(\cdot)$ is adapted from the work in [66]. In the t-th iteration, the following estimation indexed by signal dimensionality N can be computed:

   $$\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}[N] = f_{\Omega_n \leftarrow \lambda_l} \left(t, \boldsymbol{r}^{(t)}_\Omega, \tau^r_\Omega(t), \lambda_l, \hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n} \backslash \hat{\lambda}^{(t)}_{\Omega_n \leftarrow \lambda_l}\right) \ . \tag{7.29}$$

   We then have a sequence $\{\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}[N]\}$ indexed by $N = 2, 3, \cdots$. Since $\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}[N] \in \mathcal{U}_\lambda$ and $\mathcal{U}_\lambda$ is compact, it suffices to show that any sequence $\{\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}[N]\}$ converges to the same limiting point λ^*_l shown in (7.23a).

According to (7.21a), we have:

$$\begin{aligned}&\sum_{j\neq n} h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(t, \boldsymbol{r}^{(t)}_{\Omega_n}, \tau^r_{\Omega}(t), \hat{\lambda}^{(t+1)}_{\Omega_n\leftarrow\lambda_l}[N], \hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n}\backslash\hat{\lambda}^{(t)}_{\Omega_n\leftarrow\lambda_l})\\ &\quad\geq \sum_{j\neq n} h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(t, \boldsymbol{r}^{(t)}_{\Omega_n}, \tau^r_{\Omega}(t), \lambda^*_l, \hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n}\backslash\hat{\lambda}^{(t)}_{\Omega_n\leftarrow\lambda_l})\,.\end{aligned}\tag{7.30}$$

Suppose that $\hat{\lambda}^{(t+1)}_{\Omega_n\leftarrow\lambda_l}[N]$ converges to some point $\hat{\lambda}^{(t+1)}_{\Omega_n\leftarrow\lambda_l}$: $\hat{\lambda}^{(t+1)}_{\Omega_n\leftarrow\lambda_l}[N] \to \hat{\lambda}^{(t+1)}_{\Omega_n\leftarrow\lambda_l}$ as $N \to \infty$. With (7.27a) and the continuity condition of the open neighborhood $\rho(\widetilde{\tau}^r_{\Omega}, \widetilde{\boldsymbol{\lambda}})$ in Assumption 3(c), we have:

$$\begin{aligned}&\sum_{j\neq n} h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(t, \boldsymbol{r}^{(t)}_{\Omega_n}, \overline{\tau}^r_{\Omega}(t), \hat{\lambda}^{(t+1)}_{\Omega_n\leftarrow\lambda_l}[N], \overline{\boldsymbol{\lambda}}^{(t)}_{\Omega_n}\backslash\overline{\lambda}^{(t)}_{\Omega_n\leftarrow\lambda_l})\\ &\quad\geq \sum_{j\neq n} h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(t, \boldsymbol{r}^{(t)}_{\Omega_n}, \overline{\tau}^r_{\Omega}(t), \lambda^*_l, \overline{\boldsymbol{\lambda}}^{(t)}_{\Omega_n}\backslash\overline{\lambda}^{(t)}_{\Omega_n\leftarrow\lambda_l})\,.\end{aligned}\tag{7.31}$$

Since $h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(\cdot)$ is pseudo-Lipschitz continuous in $r^{(t)}_{\Omega_n}$, the left-hand side of (7.31) can be rewritten as follows as $N \to \infty$:

$$\begin{aligned}&\frac{1}{N-1}\sum_{j\neq n} h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(t, \boldsymbol{r}^{(t)}_{\Omega_n}, \overline{\tau}^r_{\Omega}(t), \hat{\lambda}^{(t+1)}_{\Omega_n\leftarrow\lambda_l}[N], \overline{\boldsymbol{\lambda}}^{(t)}_{\Omega_n}\backslash\overline{\lambda}^{(t)}_{\Omega_n\leftarrow\lambda_l})\\ &= \mathbb{E}\left[\sum_{j\neq n} h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(t, \mathcal{R}^{(t)}_{\Omega}, \overline{\tau}^r_{\Omega}(t), \hat{\lambda}^{(t+1)}_{\Omega_n\leftarrow\lambda_l}[N], \overline{\boldsymbol{\lambda}}^{(t)}_{\Omega_n}\backslash\overline{\lambda}^{(t)}_{\Omega_n\leftarrow\lambda_l})\right]\,.\end{aligned}\tag{7.32}$$

The right-hand side of (7.31) can be rewritten similarly. Equation (7.31) then becomes:

$$\begin{aligned}&\mathbb{E}\left[h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(t, \mathcal{R}^{(t)}_{\Omega}, \overline{\tau}^r_{\Omega}(t), \hat{\lambda}^{(t+1)}_{\Omega_n\leftarrow\lambda_l}[N], \overline{\boldsymbol{\lambda}}^{(t)}_{\Omega_n}\backslash\overline{\lambda}^{(t)}_{\Omega_n\leftarrow\lambda_l})\right]\\ &\geq \mathbb{E}\left[h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(t, \mathcal{R}^{(t)}_{\Omega}, \overline{\tau}^r_{\Omega}(t), \lambda^*_l, \overline{\boldsymbol{\lambda}}^{(t)}_{\Omega_n}\backslash\overline{\lambda}^{(t)}_{\Omega_n\leftarrow\lambda_l})\right]\,.\end{aligned}\tag{7.33}$$

Assumption 3(b) states that λ^*_l is the unique maxima of the right-hand side, we then have:

$$\lim_{N\to\infty} \hat{\lambda}^{(t+1)}_{\Omega_n\leftarrow\lambda_l}[N] = \lambda^*_l \,, \tag{7.34}$$

which proves (7.28).

2. **MMSE Parameter Estimation:** Using the compactness of the sets $\mathcal{U}_\lambda$ in Assumption 4(a) and the continuity condition of the open neighborhood $\rho(\widetilde{\tau}^r_\Omega, \widetilde{\boldsymbol{\lambda}})$ in Assumption 4(b), we have the following:

$$\begin{aligned} &\lim_{N\to\infty} \frac{1}{N-1} \sum_{j\neq n} h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(t, \boldsymbol{r}^{(t)}_{\Omega_n}, \tau^r_\Omega(t), \lambda_l, \hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n} \backslash \hat{\lambda}^{(t)}_{\Omega_n\leftarrow\lambda_l}) \\ &= \frac{1}{N-1} \sum_{j\neq n} h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(t, \boldsymbol{r}^{(t)}_{\Omega_n}, \overline{\tau}^r_\Omega(t), \lambda_l, \overline{\boldsymbol{\lambda}}^{(t)}_{\Omega_n} \backslash \overline{\lambda}^{(t)}_{\Omega_n\leftarrow\lambda_l}) \,. \end{aligned} \tag{7.35}$$

Since $h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}$ is pseudo-Lipschitz continuous in $r^{(t)}_{\Omega_n}$, we also have:

$$\begin{aligned} &\lim_{N\to\infty} \frac{1}{N-1} \sum_{j\neq n} h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(t, \boldsymbol{r}^{(t)}_{\Omega_n}, \overline{\tau}^r_\Omega(t), \lambda_l, \overline{\boldsymbol{\lambda}}^{(t)}_{\Omega_n} \backslash \overline{\lambda}^{(t)}_{\Omega_n\leftarrow\lambda_l}) \\ &= \mathbb{E}\left[h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(t, \mathcal{R}^{(t)}_\Omega, \overline{\tau}^r_\Omega(t), \lambda_l, \overline{\boldsymbol{\lambda}}^{(t)}_{\Omega_n} \backslash \overline{\lambda}^{(t)}_{\Omega_n\leftarrow\lambda_l}) \right] \,. \end{aligned} \tag{7.36}$$

Combining (7.35) and (7.36), we then have:

$$\begin{aligned} &\lim_{N\to\infty} \frac{1}{N-1} \sum_{j\neq n} h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(t, \boldsymbol{r}^{(t)}_{\Omega_n}, \tau^r_\Omega(t), \lambda_l, \hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n} \backslash \hat{\lambda}^{(t)}_{\Omega_n\leftarrow\lambda_l}) \\ &= \mathbb{E}\left[h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(t, \mathcal{R}^{(t)}_\Omega, \overline{\tau}^r_\Omega(t), \lambda_l, \overline{\boldsymbol{\lambda}}^{(t)}_{\Omega_n} \backslash \overline{\lambda}^{(t)}_{\Omega_n\leftarrow\lambda_l}) \right] \,. \end{aligned} \tag{7.37}$$

Using the continuity property of the exponential function $\exp(\cdot)$, as $N\to\infty$ we can get:

$$\frac{\exp(\frac{1}{N-1}\sum_{j\neq n} h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(\cdot))}{\int_{\lambda_l}\exp(\frac{1}{N-1}\sum_{j\neq n} h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(\cdot))} = \frac{\exp(\mathbb{E}\left[h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(\cdot)\right])}{\int_{\lambda_l}\exp(\mathbb{E}\left[h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(\cdot)\right])} \,. \tag{7.38}$$

Since the set $\mathcal{U}_\lambda$ is compact and the mean of a probability distribution is unique, we have:

$$\begin{aligned}\lim_{N\to\infty}\int_{\lambda_l}\lambda_l\frac{\exp(\frac{1}{N-1}\sum_{j\neq n}h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(\cdot))}{\int_{\lambda_l}\exp(\frac{1}{N-1}\sum_{j\neq n}h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(\cdot))} &= \\ \int_{\lambda_l}\lambda_l\frac{\exp(\mathbb{E}\left[h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(\cdot)\right])}{\int_{\lambda_l}\exp(\mathbb{E}\left[h^{\Omega_j}_{\Omega_n\leftarrow\lambda_l}(\cdot)\right])}&,\end{aligned} \tag{7.39}$$

which proves (7.28).

□

Additionally, we make the following assumptions about the proposed PE-GAMP algorithm.

Assumption 5. *The PE-GAMP solves a series of estimation problems indexed by the input signal dimension N:*

a) Assumptions 1(a) to 1(d) with $k=2$.

b) The scalar estimation function $g_{in}(t,r_{\Omega_n},\tau^r_\Omega,\boldsymbol{\lambda})$ *and its derivative* $g'_{in}(t,r_{\Omega_n},\tau^r_\Omega,\boldsymbol{\lambda})$ *with respect to* r_{Ω_n} *are continuous in* $\boldsymbol{\lambda}$ *uniformly over* r_{Ω_n}*: For every* $\epsilon>0,t,\widetilde{\tau}^r_\Omega,\widetilde{\boldsymbol{\lambda}}\in\mathcal{U}_\lambda$*, there exists an open neighborhood* $\rho(\widetilde{\tau}^r_\Omega,\widetilde{\boldsymbol{\lambda}})$ *of* $(\widetilde{\tau}^r_\Omega,\widetilde{\boldsymbol{\lambda}}\in\mathcal{U}_\lambda)$ *such that* $\forall(\tau^r_\Omega,\boldsymbol{\lambda})\in\rho(\widetilde{\tau}^r_\Omega,\widetilde{\boldsymbol{\lambda}})$ *and r,*

$$|g_{in}(t,r_{\Omega_n},\tau^r_\Omega,\boldsymbol{\lambda})-g_{in}(t,r_{\Omega_n},\widetilde{\tau}^r_\Omega,\widetilde{\boldsymbol{\lambda}})|<\epsilon \tag{7.40a}$$

$$|g'_{in}(t,r_{\Omega_n},\tau^r_\Omega,\boldsymbol{\lambda})-g'_{in}(t,r_{\Omega_n},\widetilde{\tau}^r_\Omega,\widetilde{\boldsymbol{\lambda}})|<\epsilon\,. \tag{7.40b}$$

In addition, $g_{in}(\cdot),g'_{in}(\cdot)$ *are pseudo-Lipschitz continuous in* r_{Ω_n} *with a Lipschitz constant that can be selected continuously in* τ^r_Ω *and* $\boldsymbol{\lambda}$*.* $g_{out}(t,q_{\Phi_m},\tau^q_\Phi,y_m,\boldsymbol{\theta})$*,* $g'_{out}(t,q_{\Phi_m},\tau^q_\Phi,y_m,\boldsymbol{\theta})$ *also satisfy analogous continuity assumptions with respect to* $q,y,\tau^q_\Phi,\boldsymbol{\theta}$*.*

c) For each $m = 1, \cdots, M$ and $n = 1, \cdots, N$, the components of the initial condition $\hat{\boldsymbol{\lambda}}^{(0)}_{\Omega_n}, \hat{\boldsymbol{\theta}}^{(0)}_{\Phi_m}$ converge as follows:

$$\lim_{N\to\infty} (\hat{\boldsymbol{\lambda}}^{(0)}_{\Omega_n}, \hat{\boldsymbol{\theta}}^{(0)}_{\Phi_m}) = (\overline{\boldsymbol{\lambda}}^{0}_{\Omega_n}, \overline{\boldsymbol{\theta}}^{(0)}_{\Phi_m})\,. \tag{7.41}$$

Specifically, Assumptions 5(a) and 5(b) are the same as those in [66]; Assumptions 5(c) is made for the proposed PE-GAMP. We then have the following Corollary 1 using Theorem 4:

Corollary 1. *Consider the proposed PE-GAMP with scalar variances under the Assumptions ⌈3,5⌋ for MAP parameter estimation and Assumptions ⌈4,5⌋ for MMSE parameter estimation. Then for any fixed iteration number t: the scalar components of (6.13,7.6) empirically converge with bounded moments of order $k = 2$ as follows:*

$$\lim_{N\to\infty} \boldsymbol{\psi}_{in} \overset{\mathrm{PL}(k)}{=} \overline{\boldsymbol{\psi}}_{in}, \quad \lim_{N\to\infty} \boldsymbol{\psi}_{out} \overset{\mathrm{PL}(k)}{=} \overline{\boldsymbol{\psi}}_{out} \tag{7.42a}$$

$$\lim_{N\to\infty} \boldsymbol{\psi}_{\tau} = \overline{\boldsymbol{\psi}}_{\tau} \tag{7.42b}$$

$$\lim_{N\to\infty} \hat{\boldsymbol{\theta}}^{(t+1)}_{\Phi_m} = \overline{\boldsymbol{\theta}}^{(t+1)}_{\Phi_m}, \quad \lim_{N\to\infty} \hat{\boldsymbol{\lambda}}^{(t+1)}_{\Omega_n} = \overline{\boldsymbol{\lambda}}^{(t+1)}_{\Omega_n}\,. \tag{7.42c}$$

Proof. We only need to show the empirical convergences of $\hat{\boldsymbol{\lambda}}^{(t+1)}_{\Omega_n}, \hat{\boldsymbol{\theta}}^{(t+1)}_{\Phi_m}$. From Assumption 3 we can get Lemma 5, which corresponds to Assumption 2(c). Using Theorem 4, we have:

$$\lim_{N\to\infty} r^{(t)}_{\Omega_n} \overset{\mathrm{PL}(k)}{=} \mathcal{R}^{(t)}_{\Omega} \tag{7.43a}$$

$$\lim_{N\to\infty} (q^{(t)}_{\Phi_m}, y_m) \overset{\mathrm{PL}(k)}{=} (\mathcal{Q}^{(t)}_{\Phi}, \mathcal{Y}) \tag{7.43b}$$

$$\lim_{N\to\infty} \boldsymbol{\psi}_{\tau} = \overline{\boldsymbol{\psi}}_{\tau}\,. \tag{7.43c}$$

The empirical convergences of the parameters can be proved using induction. For $t = 0$, the convergences of $\hat{\boldsymbol{\lambda}}^{(0)}_{\Omega_n}, \hat{\boldsymbol{\theta}}^{(0)}_{\Phi_m}$ hold according to (7.41) in Assumption

5(c). With (7.41, 7.43a), we can use Lemma 5 to obtain:

$$\lim_{N\to\infty} \hat{\boldsymbol{\lambda}}_{\Omega_n}^{(1)} = \overline{\boldsymbol{\lambda}}_{\Omega_n}^{(1)}, \quad \lim_{N\to\infty} \hat{\boldsymbol{\theta}}_{\Phi_m}^{(1)} = \overline{\boldsymbol{\theta}}_{\Phi_m}^{(1)}. \tag{7.44}$$

The convergences of the rest scalars can be obtained directly using Theorem 4. Hence the following holds for any t.

$$\lim_{N\to\infty} \hat{\boldsymbol{\lambda}}_{\Omega_n}^{(t+1)} = \overline{\boldsymbol{\lambda}}_{\Omega_n}^{(t+1)}, \quad \lim_{N\to\infty} \hat{\boldsymbol{\theta}}_{\Phi_m}^{(t+1)} = \overline{\boldsymbol{\theta}}_{\Phi_m}^{(t+1)}. \tag{7.45}$$

□

Chapter 8

Probabilistic sparse signal recovery via PE-GAMP

In this chapter we present detailed PE-GAMP formulations of the following three probabilistic sparse signal recovery models that assume the input and out channel distributions given in section 2.4.

- *Bernoulli-Gaussian mixture* (BGm) input channel (2.7) "+" *Additive white Gaussian noise* (AWGN) output channel (2.10).
- *Bernoulli-Exponential mixture* (BEm) input channel (2.8) "+" *Additive white Gaussian noise* (AWGN) output channel (2.10).
- *Laplace* input channel (2.9) "+" *Additive white Gaussian noise* (AWGN) output channel (2.10).

For experimental evaluation, we choose the PE-GAMP with MAP parameter estimation and compare it with EM based parameter estimation approach on both simulated and real datasets. Experiments show that the proposed PE-GAMP not only has a wider applicability, but also is more robust and outperforms EM based approach in adversarial conditions.

8.1 Max-sum message passing

We analyze the various channels as follows:

1. **Bernoulli-Gaussian mixture Input Channel:** BGm input channel is not really suited for the *max-sum* message passing. If we compute (6.54b), the maximizing x_j would be 0, which makes both the parameter estimation and signal recovery impossible.

2. **Bernoulli-Exponential mixture Input Channel:** BEx input channel is also not suited for the *max-sum* message passing for the same reason as the BGm input channel.

3. **Laplace Input Channel:** (6.54b) can be written as follows:

$$\Delta_{\Omega_j \to \lambda_1}^{(t+1)} = \max_{x_j} \left[\log \lambda_1 - \lambda_1 \left| x_j \right| - \frac{\left(x_j - r_{\Omega_j}^{(t)} \right)^2}{2\tau_{\Omega_j}^r(t)} \right] . \tag{8.1}$$

   The maximizing x_j is given by the soft-thresholding method:

$$\tilde{x}_n^{(t+1)} = \left(\left| r_{\Omega_j}^{(t)} \right| - \lambda_1 \tau_{\Omega_j}^r(t) \right)_+ \cdot \operatorname{sign}\left(r_{\Omega_j}^{(t)} \right) . \tag{8.2}$$

   If $\left| r_{\Omega_j}^{(t)} \right| > \lambda_1 \tau_{\Omega_j}^r(t)$, we have:

$$\Delta_{\Omega_j \to \lambda_1}^{(t+1)} = \log \lambda_1 - \lambda_1 \left| r_{\Omega_j}^{(t)} \right| + \frac{1}{2} \lambda_1^2 \tau_{\Omega_j}^r(t) . \tag{8.3}$$

   If $\left| r_{\Omega_j}^{(t)} \right| \leq \lambda_1 \tau_{\Omega_j}^r(t)$, we have:

$$\Delta_{\Omega_j \to \lambda_1}^{(t+1)} = \log \lambda_1 - \frac{1}{2\tau_{\Omega_j}^r(t)} \left(r_{\Omega_j}^{(t)} \right)^2 . \tag{8.4}$$

   We can see from (8.3, 8.4) that the λ_1 that maximizes (6.50a) is always ∞, which makes the estimation of λ_1 impossible.

4. **Additive White Gaussian Noise Output Channel:** (6.54c) can be written as follows:

$$\begin{aligned}\Delta_{\Phi_i \to \theta_1}^{(t+1)} = \max_{z_i} \Bigg[& -\frac{1}{2}\log\theta_1 - \frac{1}{2\theta_1}(y_i - z_i)^2 \\ & -\frac{1}{2\tau_{\Phi_i}^q(t)}\left(z_i - q_{\Phi_i}^{(t)}\right)^2 \Bigg] .\end{aligned} \tag{8.5}$$

The maximizing z_i is:

$$\tilde{z}_m = \frac{y_i \tau_{\Phi_i}^q(t) + q_{\Phi_i}^{(t)}\theta_1}{\theta_1 + \tau_{\Phi_i}^q(t)} . \tag{8.6}$$

We then have:

$$\begin{aligned}&\Delta_{\Phi_i \to \theta_1}^{(t+1)} \\ &= -\frac{1}{2}\log\theta_1 + \frac{\left(q_{\Phi_i}^{(t)}\right)^2\theta_1 + 2q_{\Phi_i}^{(t)} y_i \tau_{\Phi_i}^q(t) - y_i^2 \tau_{\Phi_i}^q(t)}{2\tau_{\Phi_i}^q(t)\left(\theta_1 + \tau_{\Phi_i}^q(t)\right)} .\end{aligned} \tag{8.7}$$

We can see from (8.7) that the θ_1 that maximizes (6.50b) is always 0, which makes the estimation of θ_1 impossible.

❖ **Discussion:** For the two models with BGm and BEm input channels, *max-sum* message passing cannot produce any useful MAP estimation of $\boldsymbol{x}$. In this case, we can only use *sum-product* message passing to perform MMSE estimation of $\boldsymbol{x}$.

For the model with Laplace input channel, although *max-sum* message passing can be used to obtain the MAP estimation of $\boldsymbol{x}$, it cannot be used to compute the MAP estimation of λ_1, since the $\hat{\lambda}_1$ that maximizes (6.14) is always ∞ and the maximizing $\hat{\theta}_1$ is always 0. On the other hand, *sum-product* message passing can be used to compute the MMSE estimation and MAP estimation of x_n based on $p(x_n|\boldsymbol{y})$, however they don't have the best recovery performance. Here we propose to employ *sum-product* message passing to compute the "marginal" MAP estimates $\{\hat{\lambda}_1, \hat{\theta}_1\}$ using the marginal posterior distributions $p(\lambda_1|\boldsymbol{y}), p(\theta_1|\boldsymbol{y})$, as opposed to the MAP estimates in (6.50). $\{\hat{\lambda}_1, \hat{\theta}_1\}$ can then be used as the inputs to *max-sum* message passing to obtain the MAP estimate of $\boldsymbol{x}$. This essentially is the Lasso mentioned at the beginning of this paper, except now that we have provided a way to automatically estimate the parameters.

8.2 Sum-product message passing

In this case, the two recovery models mentioned earlier both rely on *sum-product* message passing to perform parameter estimation. For the *sum-product* message passing, "MMSE parameter estimation" is often quite difficult to compute, in this paper we will focus on using the "MAP parameter estimation" approach to estimate the parameters. Since we don't have any knowledge about the priors of $\boldsymbol{\lambda}, \boldsymbol{\theta}$, we will fairly choose the *uniform* prior for each parameter.

The proposed PE-GAMP computes MAP estimations of the parameters in the *sum-product* message passing as follows:

$$\begin{aligned}\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l} &= \arg\max_{\lambda_l} h^{(t+1)}_{\Omega_n \leftarrow \lambda_l}(\cdot) \\ &= \arg\max_{\lambda_l} \sum_{j \neq n} \Delta^{(t+1)}_{\Omega_j \rightarrow \lambda_l} + \log p(\lambda_l)\end{aligned} \tag{8.8a}$$

$$\begin{aligned}\hat{\theta}^{(t+1)}_{\Phi_m \leftarrow \theta_k} &= \arg\max_{\theta_k} h^{(t+1)}_{\Phi_m \leftarrow \theta_k}(\cdot) \\ &= \arg\max_{\theta_k} \sum_{i \neq k} \Delta^{(t+1)}_{\Phi_i \rightarrow \theta_k} + \log p(\theta_k)\,.\end{aligned} \tag{8.8b}$$

Specifically, we use the line search method given in the following Algorithm 5 to find $\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}$.

Algorithm 5 MAP parameter estimation via line search method

```
Require: $\hat{\lambda}^{(t)}_{\Omega_n \leftarrow \lambda_l}$, $\frac{\partial h^{(t+1)}_{\Omega_n \leftarrow \lambda_l}}{\partial \lambda_l}$, $0 < \zeta < 1, \eta_+ > 0, \eta_- < 0$
Set $\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}(0) = \hat{\lambda}^{(t)}_{\Omega_n \leftarrow \lambda_l}$.
for $i = 1, 2, \cdots$ do
    if $\frac{\partial h^{(t+1)}_{\Omega_n \leftarrow \lambda_l}}{\partial \lambda_l}\Big|_{\lambda_l = \hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}(i-1)} > 0$ then
        $\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l} = \hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}(i-1) + \eta_+ \,.$    (8.9)
        while $h^{(t+1)}_{\Omega_n \leftarrow \lambda_l}\Big|_{\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}} < h^{(t+1)}_{\Omega_n \leftarrow \lambda_l}\Big|_{\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}(i-1)}$ do
            $\eta_+ = \eta_+ \cdot \zeta$    (8.10a)
            $\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l} = \hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}(i-1) + \eta_+ \,.$    (8.10b)
        end while
    else if $\frac{\partial h^{(t+1)}_{\Omega_n \leftarrow \lambda_l}}{\partial \lambda_l}\Big|_{\lambda_l = \hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}(i-1)} < 0$ then
        $\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l} = \hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}(i-1) + \eta_- \,.$    (8.11)
        while $h^{(t+1)}_{\Omega_n \leftarrow \lambda_l}\Big|_{\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}} < h^{(t+1)}_{\Omega_n \leftarrow \lambda_l}\Big|_{\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}(i-1)}$ do
            $\eta_- = \eta_- * \zeta$    (8.12a)
            $\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l} = \hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}(i-1) + \eta_- \,.$    (8.12b)
        end while
    else
        break;
    end if
```

Algorithm 5 MAP parameter estimation via line search method (continued)

```
    Set $\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}(i) = \hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}$
    if $\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}(i)$ reaches convergence then
        break;
    end if
end for
Return Output $\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}$;
```

The maximizing $\hat{\theta}^{(t+1)}_{\Phi_m \leftarrow \theta_k}$ can be found similarly. The line search method requires computing the derivatives of $h^{(t+1)}_{\Omega_n \leftarrow \lambda_l}(\cdot), h^{(t+1)}_{\Phi_m \leftarrow \theta_k}(\cdot)$ (8.8) with respect to the parameters λ_l, θ_k.

$$h^{(t+1)}_{\Omega_n \leftarrow \lambda_l}(\cdot) = \sum_{j \neq n} \Delta^{(t+1)}_{\Omega_j \rightarrow \lambda_l} + \log p(\lambda_l) \tag{8.13a}$$

$$h^{(t+1)}_{\Phi_m \leftarrow \theta_k}(\cdot) = \sum_{i \neq m} \Delta^{(t+1)}_{\Phi_i \rightarrow \theta_k} + \log p(\theta_k)\,. \tag{8.13b}$$

The derivatives of $\log p(\lambda_l), \log p(\theta_k))$ depends on the chosen priors and are easy to compute. Here we give the derivatives of $\Delta^{(t+1)}_{\Omega_j \rightarrow \lambda_l}, \Delta^{(t+1)}_{\Phi_i \rightarrow \theta_k}$ with respect to λ_l, θ_k in details.

1. **Bernoulli-Gaussian mixture Input Channel:** BGm distribution is given in (2.7). We then have:

$$\begin{aligned}
&\log \int_{-\infty}^{\infty} \Omega_j(x_j, \boldsymbol{\lambda}) \cdot \exp\left(\Delta^{(t+1)}_{\Omega_j \leftarrow x_j}\right) dx_j \\
&= \log \int_{x_j} p(x_j|\boldsymbol{\lambda}) \exp\left(-\frac{1}{2\tau^r_{\Omega_j}(t)}\left(x_j - r^{(t)}_{\Omega_j}\right)\right) \\
&= \log\left[(1-\lambda_1)\cdot\kappa_1 + \lambda_1 \sum_{c=1}^{C} \lambda_{c+1}\cdot\kappa_2(\lambda_{c+2}, \lambda_{c+3})\right] \\
&= \log\left[(1-\lambda_1)\cdot\kappa_1 + \lambda_1 \sum_{c=1}^{C} \lambda_{c+1}\cdot\kappa_2(c)\right],
\end{aligned} \tag{8.14}$$

where κ_1 doesn't depend on $\boldsymbol{\lambda}$; for the c-th Gaussian mixture, $\kappa_2(c) = \kappa_2(\lambda_{c+2}, \lambda_{c+3})$ depends on $\lambda_{c+2}, \lambda_{c+3}$.

$$\kappa_1 = \exp\left(-\frac{\left(r_{\Omega_j}^{(t)}\right)^2}{2\tau_{\Omega_j}^{r}(t)}\right) \tag{8.15a}$$

$$\kappa_2(c) = \sqrt{\frac{\tau_{\Omega_j}^{r}(t)}{\lambda_{c+3} + \tau_{\Omega_j}^{r}(t)}} \exp\left(-\frac{1}{2}\frac{\left(\lambda_{c+2} - r_{\Omega_j}^{(t)}\right)^2}{\left(\lambda_{c+3} + \tau_{\Omega_j}^{r}(t)\right)}\right). \tag{8.15b}$$

Here, (8.14) is essentially (6.12c). Let $\kappa_3(\boldsymbol{\lambda})$ be as follows:

$$\kappa_3(\boldsymbol{\lambda}) = \frac{\lambda_1}{(1-\lambda_1)\cdot\kappa_1 + \lambda_1 \sum_{c=1}^{C} \lambda_{c+1}\cdot\kappa_2(c)}. \tag{8.16}$$

Let $\boldsymbol{\lambda}\backslash\lambda_l$ denote the parameter sequence generated by removing λ_l from $\boldsymbol{\lambda}$. Taking derivatives of (6.12c) with respect to $\lambda_1, \lambda_{c+2}, \lambda_{c+3}$, we have:

$$\frac{\partial \Delta_{\Omega_j\to\lambda_1}^{(t+1)}}{\partial \lambda_1} = \frac{-\kappa_1 + \sum_{c=1}^{C} \hat{\lambda}_{c+1}^{(t)}\cdot\kappa_2\left(\hat{\lambda}_{c+2}^{(t)}, \hat{\lambda}_{c+3}^{(t)}\right)}{(1-\lambda_1)\cdot\kappa_1 + \lambda_1\sum_{c=1}^{C} \hat{\lambda}_{c+1}^{(t)}\cdot\kappa_2\left(\hat{\lambda}_{c+2}^{(t)}, \hat{\lambda}_{c+3}^{(t)}\right)} \tag{8.17a}$$

$$\begin{aligned}\frac{\partial \Delta_{\Omega_j\to\lambda_{c+2}}^{(t+1)}}{\partial \lambda_{c+2}} &= \kappa_3\left(\hat{\boldsymbol{\lambda}}^{(t)}\backslash\hat{\lambda}_{c+2}^{(t)}, \lambda_{c+2}\right) \\ &\quad\times \frac{-\kappa_2\left(\lambda_{c+2}, \hat{\lambda}_{c+3}^{(t)}\right)\left(\lambda_{c+2} - r_{\Omega_j}^{(t)}\right)}{\hat{\lambda}_{c+2}^{(t)} + \tau_{\Omega_j}^{r}(t)}\end{aligned} \tag{8.17b}$$

$$\begin{aligned}\frac{\partial \Delta_{\Omega_j\to\lambda_{c+3}}^{(t+1)}}{\partial \lambda_{c+3}} &= -\kappa_3\left(\hat{\boldsymbol{\lambda}}^{(t)}\backslash\hat{\lambda}_{c+3}^{(t)}, \lambda_{c+3}\right)\frac{\kappa_2\left(\hat{\lambda}_{c+2}^{(t)}, \lambda_{c+3}\right)}{2\left(\lambda_{c+3} + \tau_{\Omega_j}^{r}(t)\right)} \\ &\quad\times\left(1 - \frac{\left(\hat{\lambda}_{c+2}^{(t)} - r_{\Omega_j}^{(t)}\right)^2}{\lambda_{c+3} + \tau_{\Omega_j}^{r}(t)}\right),\end{aligned} \tag{8.17c}$$

where $\hat{\boldsymbol{\lambda}}^{(t)}$ are the estimated parameters in the previous t-th iteration.

The updates for the weights λ_{c+1} are more complicated, they need to satisfy the nonnegative and sum-to-one constrains. Here we can rewrite the weight λ_{c+1} as follows :

$$\lambda_{c+1} = \frac{\exp(\omega_c)}{\sum_{k=1}^{C} \exp(\omega_k)} \,, \tag{8.18}$$

where $\omega_c \in \mathbb{R}$. We then can remove the constrains on λ_{c+1} and maximize $\Delta^{(t+1)}_{\Omega_j \to \lambda_{c+1}}$ with respect to ω_c instead. The derivative is then:

$$\begin{aligned}
\frac{\partial \Delta^{(t+1)}_{\Omega_j \to \lambda_{c+1}}}{\partial \omega_c} &= \frac{\partial \Delta^{(t+1)}_{\Omega_j \to \lambda_{c+1}}}{\partial \lambda_{c+1}} \cdot \frac{\partial \lambda_{c+1}}{\partial \omega_c} \\
&= \frac{\hat{\lambda}_1^{(t)}}{(1-\hat{\lambda}_1^{(t)}) \cdot \kappa_1 + \hat{\lambda}_1^{(t)} \sum_{k=1}^{C} \lambda_{k+1} \cdot \kappa_2 \left(\hat{\lambda}_{k+2}^{(t)}, \hat{\lambda}_{k+3}^{(t)} \right)} \\
&\quad \times \sum_{k=1}^{C} \kappa_2 \left(\hat{\lambda}_{k+2}^{(t)}, \hat{\lambda}_{k+3}^{(t)} \right) \left(\lambda_{k+1} \cdot \mathbf{1}_{(k=c)} - \lambda_{k+1} \lambda_{c+1} \right) \,,
\end{aligned} \tag{8.19}$$

where $\mathbf{1}_{(k=c)} = 1$ if $k = c$ and $\mathbf{1}_{(k=c)} = 0$ if $k \neq c$.

2. **Bernoulli-Exponential mixture Input Channel:** BEm distribution is given in (2.8), we then have:

$$\begin{aligned}
&\log \int_0^{\infty} \Omega_j(x_j, \boldsymbol{\lambda}) \cdot \exp \left(\Delta^{(t+1)}_{\Omega_j \leftarrow x_j} \right) dx_j \\
&= \log \int_{x_j} p(x_j|\boldsymbol{\lambda}) \exp \left(-\frac{1}{2\tau^r_{\Omega_j}(t)} \left(x_j - r^{(t)}_{\Omega_j} \right) \right) \\
&= \log \left[(1-\lambda_1) \cdot \kappa_1 + \lambda_1 \sum_{c=1}^{C} \lambda_{c+1} \cdot \kappa_2(\lambda_{c+2}) \right] \\
&= \log \left[(1-\lambda_1) \cdot \kappa_1 + \lambda_1 \sum_{c=1}^{C} \lambda_{c+1} \cdot \kappa_2(c) \right] \,,
\end{aligned} \tag{8.20}$$

where κ_1 is the same as (8.15a); for the c-th Exponential mixture, $\kappa_2(c)$

depends on λ_{c+2}.

$$\kappa_2(c) = \lambda_{c+2} \cdot \sqrt{\frac{\pi \tau^r_{\Omega_j}(t)}{2}} \cdot \operatorname{erfcx}\left(-\frac{r^{(t)}_{\Omega_j} - \lambda_{c+2} \cdot \tau^r_{\Omega_j}(t)}{\sqrt{2\tau^r_{\Omega_j}(t)}}\right) , \tag{8.21}$$

where erfcx$(\cdot)$ is the scaled complementary error function. Taking the derivative w.r.t. λ_1, λ_{c+2}, we have:

$$\frac{\partial \Delta^{(t+1)}_{\Omega_j \to \lambda_1}}{\partial \lambda_1} = \frac{-\kappa_1 + \sum_{c=1}^{C} \hat{\lambda}^{(t)}_{c+1} \cdot \kappa_2\left(\hat{\lambda}^{(t)}_{c+2}\right)}{(1-\lambda_1) \cdot \kappa_1 + \lambda_1 \sum_{c=1}^{C} \hat{\lambda}^{(t)}_{c+1} \cdot \kappa_2\left(\hat{\lambda}^{(t)}_{c+2}\right)} \tag{8.22a}$$

$$\begin{aligned} \frac{\partial \Delta^{(t+1)}_{\Omega_j \to \lambda_{c+2}}}{\partial \lambda_{c+2}} &= \kappa_3\left(\hat{\boldsymbol{\lambda}}^{(t)} \backslash \hat{\lambda}^{(t)}_{c+2}, \lambda_{c+2}\right) \\ &\times \hat{\lambda}^{(t)}_{c+1}\left(\lambda_{c+2}\tau^r_{\Omega_j}(t) + \left(r^{(t)}_{\Omega_j} - \lambda_{c+2}\tau^r_{\Omega_j}(t)\right)\kappa_2(c)\right) . \end{aligned} \tag{8.22b}$$

We write the mixture weights λ_{c+1} in the same form as (8.18), and take the derivative w.r.t. ω_c:

$$\begin{aligned} \frac{\partial \Delta^{(t+1)}_{\Omega_j \to \lambda_{c+1}}}{\partial \omega_c} &= \frac{\partial \Delta^{(t+1)}_{\Omega_j \to \lambda_{c+1}}}{\partial \lambda_{c+1}} \cdot \frac{\partial \lambda_{c+1}}{\partial \omega_c} \\ &= \frac{\hat{\lambda}^{(t)}_1}{(1-\hat{\lambda}^{(t)}_1) \cdot \kappa_1 + \hat{\lambda}^{(t)}_1 \sum_{k=1}^{C} \lambda_{k+1} \cdot \kappa_2\left(\hat{\lambda}^{(t)}_{k+2}\right)} \\ &\times \sum_{k=1}^{C} \kappa_2\left(\hat{\lambda}^{(t)}_{k+2}\right)\left(\lambda_{k+1} \cdot \mathbf{1}_{(k=c)} - \lambda_{k+1}\lambda_{c+1}\right) . \end{aligned} \tag{8.23}$$

3. **Laplace Input Channel:** Laplace distribution is given in (2.9). Similarly,

we have:

$$\begin{aligned}
&\log \int_{-\infty}^{\infty} \Omega_j(x_j, \boldsymbol{\lambda}) \cdot \exp\left(\Delta^{(t+1)}_{\Omega_j \leftarrow x_j}\right)\, dx_j \\
&= \log \int_{x_j} p(x_j|\boldsymbol{y}) \cdot \exp\left(-\frac{\left(x_j - r^{(t)}_{\Omega_j}\right)^2}{2\tau^r_{\Omega_j}(t)}\right) \\
&= \log\left[\lambda_1 \kappa_1 \left(\kappa_2(\lambda_1) + \kappa_3(\lambda_1)\right)\right] - \log 2\,,
\end{aligned} \tag{8.24}$$

where κ_1 is the same as (8.15a), and $\kappa_2(\lambda_1), \kappa_3(\lambda_1)$ depend on λ_1. They can be described as follows:

$$\kappa_2(\lambda_1) = \lambda_1 \cdot \sqrt{\frac{\pi \tau^r_{\Omega_j}(t)}{2}} \cdot \operatorname{erfcx}\left(-\frac{r^{(t)}_{\Omega_j} - \lambda_1 \cdot \tau^r_{\Omega_j}(t)}{\sqrt{2\tau^r_{\Omega_j}(t)}}\right) \tag{8.25a}$$

$$\kappa_3(\lambda_1) = \lambda_1 \cdot \sqrt{\frac{\pi \tau^r_{\Omega_j}(t)}{2}} \cdot \operatorname{erfcx}\left(\frac{r^{(t)}_{\Omega_j} + \lambda_1 \cdot \tau^r_{\Omega_j}(t)}{\sqrt{2\tau^r_{\Omega_j}(t)}}\right)\,. \tag{8.25b}$$

Taking derivative of (6.12c) with respect to λ_1, we have:

$$\frac{\partial \Delta^{(t+1)}_{\Omega_j \to \lambda_1}}{\partial \lambda_1} = \frac{1}{\lambda_1} + \frac{\kappa_4(\lambda_1) + \kappa_5(\lambda_1)}{\kappa_2(\lambda_1) + \kappa_3(\lambda_1)}\,, \tag{8.26}$$

where $\kappa_4(\lambda_1), \kappa_5(\lambda_1)$ depend on λ_1. They can be expressed as:

$$\kappa_4(\lambda_1) = \lambda_1 \tau^r_{\Omega_j}(t) + \left(r^{(t)}_{\Omega_j} - \lambda_1 \tau^r_{\Omega_j}(t)\right) \kappa_2(\lambda_1) \tag{8.27a}$$

$$\kappa_5(\lambda_1) = -\lambda_1 \tau^r_{\Omega_j}(t) + \left(r^{(t)}_{\Omega_j} + \lambda_1 \tau^r_{\Omega_j}(t)\right) \kappa_3(\lambda_1)\,. \tag{8.27b}$$

4. **Additive White Gaussian Noise Output Channel:** The white Gaussian

distribution is shown in 2.10. We have the following:

$$\log \int_{-\infty}^{\infty} \Phi(y_i, z_i, \boldsymbol{\theta}) \exp\left(\frac{-1}{2\tau^q_{\Phi_i}(t)} \left(z_i - q^{(t)}_{\Phi_i} \right)^2 \right) dz_m$$

$$= \frac{1}{2} \log \tau^q_{\Phi_i}(t) - \frac{1}{2} \log \left(\theta_1 + \tau^q_{\Phi_i}(t) \right) \tag{8.28}$$

$$- \frac{1}{2} \frac{1}{\left(\theta_1 + \tau^q_{\Phi_i}(t) \right)} \left(y_i - q^{(t)}_{\Phi_i} \right)^2 .$$

Taking derivative of (6.12d) with respect to θ_1, we have:

$$\frac{\partial \Delta^{(t+1)}_{\Phi_i \to \theta_1}}{\partial \theta_1} = \frac{\left(y_i - q^{(t)}_{\Phi_i} \right)^2}{2\left(\theta_1 + \tau^q_{\Phi_i}(t) \right)^2} - \frac{1}{2\left(\theta_1 + \tau^q_{\Phi_i}(t) \right)} . \tag{8.29}$$

8.2.1 Comparison with EM parameter estimation

Here we discuss the differences between the proposed PE-GAMP with MAP parameter estimation and the EM-GAMP with EM parameter estimation [26, 65].

First of all, the EM parameter estimation is essentially maximum likelihood estimation. EM [27] tries to find the parameters $\boldsymbol{\lambda}, \boldsymbol{\theta}$ that maximize the likelihood $p(\boldsymbol{y}|\boldsymbol{\lambda}), p(\boldsymbol{y}|\boldsymbol{\theta})$. While the proposed PE-GAMP with MAP parameter estimation tries to maximize the following posterior distributions at nodes Ω_n, Φ_m using Bayes' rule:

$$p_{\Omega_n}(\boldsymbol{\lambda}|\boldsymbol{y}) \propto p_{\Omega_n}(\boldsymbol{y}|\boldsymbol{\lambda}) p(\boldsymbol{\lambda}) \tag{8.30a}$$

$$p_{\Phi_m}(\boldsymbol{\theta}|\boldsymbol{y}) \propto p_{\Phi_m}(\boldsymbol{y}|\boldsymbol{\theta}) p(\boldsymbol{\theta}) . \tag{8.30b}$$

Compared to EM estimation, the MAP estimation is able to draw information from the priors $p(\boldsymbol{\lambda}), p(\boldsymbol{\theta}))$ to guide the estimation process.

Secondly, the two methods also differ in the way they compute the maximizing parameters. For the sake of simplification and a fair comparison, we will assume the priors of the parameters $p(\boldsymbol{\lambda}), p(\boldsymbol{\theta})$ to be uniform distributions. Specifically, EM treats $\boldsymbol{x}, \boldsymbol{w}$ as hidden variables and maximizes $\mathbb{E}[\log p(\boldsymbol{x}, \boldsymbol{w}; \boldsymbol{\lambda}, \boldsymbol{\theta}) | \boldsymbol{y}, \hat{\boldsymbol{\lambda}}^{(t)}, \hat{\boldsymbol{\theta}}^{(t)}]$ iteratively until convergence. Take the parameter λ_l for example, in the $(t+1)$-th iteration the following expression will be maximized under the GAMP framework [26]:

$$\begin{aligned} &\max_{\lambda_l} \sum_j \int p\left(x_j | \boldsymbol{y}, \hat{\lambda}_l^{(t)}\right) \log p(x_j|\lambda_l)\, dx_j \\ &\propto \sum_j \int p\left(x_j | \hat{\lambda}_l^{(t)}\right) \mathcal{N}\left(x_j; r_{\Omega_j}^{(t)}, \tau_{\Omega_j}^r(t)\right) \log p(x_j|\lambda_l)\, dx_j\,, \end{aligned} \tag{8.31}$$

where $\hat{\lambda}_l^{(t)}$ is the estimated parameter in the previous t-th iteration. The closed-form expression for Bernoulli-Gaussian mixture distribution can be found in [26]. However, (8.31) is quite difficult to evaluate for more complicated distributions, which greatly limits its applicabilities. The proposed PE-GAMP with MAP parameter estimation has a much simpler expression though:

$$\max_{\lambda_l} \sum_{j \neq n} \log \int p(x_j|\lambda_l) \mathcal{N}\left(x_j; r_{\Omega_j}^{(t)}, \tau_{\Omega_j}^r(t)\right)\, dx_j\,. \tag{8.32}$$

This enables us to consider more complex distributions with the proposed PE-GAMP. For instance, in this dissertation we have included the formulations to estimate the parameters for sparse signals with Laplace prior and Bernoulli-Exponential mixture prior.

8.3 Experimental results

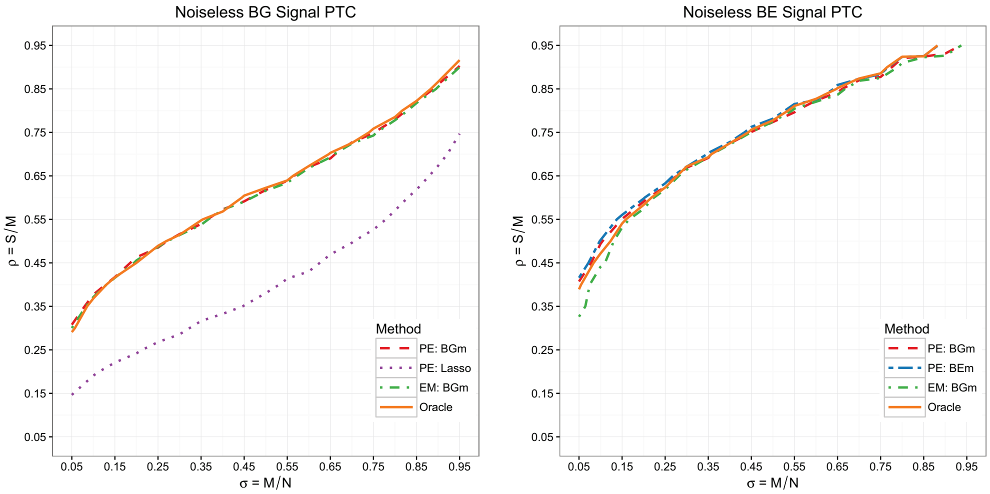
8.3.1 Simulated sparse signal recovery

We first perform noiseless sparse signal recovery experiments and compare the empirical phase transition curves (PTC) of PE-GAMP and EM-BGm-GAMP [26]. Besides, oracle experiments where the "true" parameters are known are also performed. Specifically, we fix $N = 1000$ and vary the sampling ratio $\sigma = \frac{M}{N} \in [0.05, 0.1, 0.15, \cdots, 0.95]$ and the sparsity ratio $\rho = \frac{S}{M} \in [0.05, 0.1, 0.15, \cdots, 0.95]$, where S is the sparsity of the signal, i.e. the number of nonzero coefficients. For each combination of σ and ρ, we randomly generate 100 pairs of $\{\boldsymbol{x}, \boldsymbol{A}\}$: A is a $M \times N$ random Gaussian matrix with normalized and centralized rows; the *nonzero* entries of the sparse signal $\boldsymbol{x} \in \mathbb{R}^N$ are i.i.d. generated according to the following two different distributions:

1. Gaussian distribution $x \sim \mathcal{N}(0, 1)$.
2. Exponential distribution $x \sim \exp(-x)$, $x \geq 0$.

In other words, the sparse signals $\boldsymbol{x}$ follow Bernoulli-Gaussian (BG) and Bernoulli-Exponential (BE) distributions respectively. Given the measurement vector $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ and the sensing matrix $\boldsymbol{A}$, we try to recover the signal $\boldsymbol{x}$. If $\epsilon = \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2 / \|\boldsymbol{x}\|_2 < 10^{-3}$, the recovery is considered to be a success. Based on the 100 trials, we compute the success recovery rate for each combination of σ and ρ and plot the PTCs in Fig. 8.1.

The PTC is the contour that corresponds to the 0.5 success rate in the domain $(\sigma, \rho) \in (0, 1)^2$, it divides the domain into a "success" phase (lower right) and a "failure" phase (upper left). For the BG sparse signals (Fig. 8.1(a)), the PE-BGm-GAMP and EM-BGm-GAMP perform equally well and match the performance



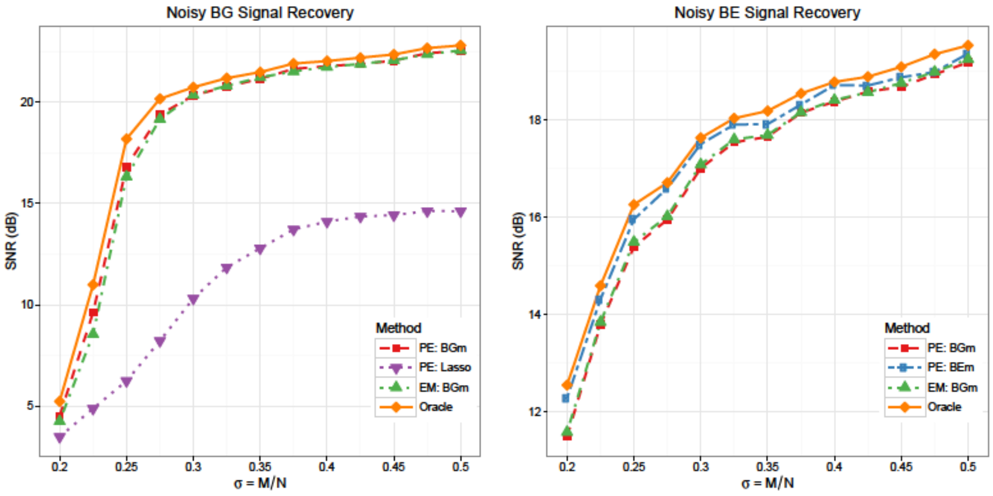
(a) (b)

Figure 8.1: The phase transition curves (PTC) of different GAMP methods in the noiseless case. (a) Bernoulli-Gaussian (BG) sparse signal; (b) Bernoulli-Exponential (BE) sparse signal.

of the oracle-GAMP. The BGm prior they assumed about the sparse signal is a perfect match, which is much better than Laplace prior assumed by PE-Lasso-GAMP.

For the BE sparse signals (Fig. 8.1(b)), the BEm prior assumed by PE-BEm-GAMP is the perfect match. However, we can see that the PTC of PE-BGm-GAMP is only slightly worse, the BGm prior is still a strong contestant in this case. Although both PE-BGm-GAMP and EM-BGm-GAMP assume the BGm prior, PE-BGm-GAMP is more robust and performs better than EM-BGm-GAMP when the sampling rate is low. PE-BEm-GAMP is the only one that matches the performance of the oracle-GAMP.

We next try to recover the sparse signal $\boldsymbol{x}$ from a noisy measurement vector $\boldsymbol{y}$. Specifically, we fix $S = 100$, $N = 1000$ and increase the number of measurement



(a) (b)

Figure 8.2: The signal-to-noise-ratio (SNR) of the recovered sparse signals using different GAMP methods in the noisy case. (a) Bernoulli-Gaussian (BG) sparse signal; (b) Bernoulli-Exponential (BE) sparse signal.

M. $\boldsymbol{y} \in \mathbb{R}^M$ is generated as follows:

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \nu \boldsymbol{w}\,, \tag{8.33}$$

where $\nu > 0$ controls the amount of noise added to $\boldsymbol{y}$, the entries of $\boldsymbol{w}$ are i.i.d Gaussian $\mathcal{N}(0,1)$. We choose $\nu = 0.05$ for the BG sparse signals and $\nu = 0.1$ for the BE sparse signals. This creates a measurement $\boldsymbol{y}$ with signal to noies ratio (SNR) around 20 dB. We randomly generate 100 triples of $\{\boldsymbol{x}, \boldsymbol{A}, \boldsymbol{w}\}$. The average SNRs of the recovered signals $\hat{\boldsymbol{x}}$ are shown in Fig. 8.2.

In the noisy case, the oracle-GAMP performs the best as expected since the "true" parameters are used to recover the sparse signal, and the GAMP methods using estimated parameters are not bad either. For the BG sparse signals (Fig. 8.2(a)), we can see that PE-BGm-GAMP performs better than EM-BGm-GAMP when the sampling ratio is small. Since BGm is a better match than the Laplace prior, both PE-BGm-GAMP and EM-BGm-GAMP perform much better than

PE-Lasso-GAMP. For the BE sparse signals (Fig. 8.2(b)), the BEm prior is a better match than the BGm prior. PE-BEm-GAMP is able to perform better than PE-BGm-GAMP and EM-BGm-GAMP, especially when the sampling ratio is small. Additionally, the solutions produced by PE-BEm-GAMP is guaranteed to be non-negative, while those by PE-BGm-GAMP and EM-BGm-GAMP generally contains negative coefficients. For applications that requires non-negative sparse solutions, such as hyper-spectral unmixing [67], non-negative sparse coding for image classification [68], etc, PE-BEm-GAMP offers a convenient way to solve the parameter estimation problem.

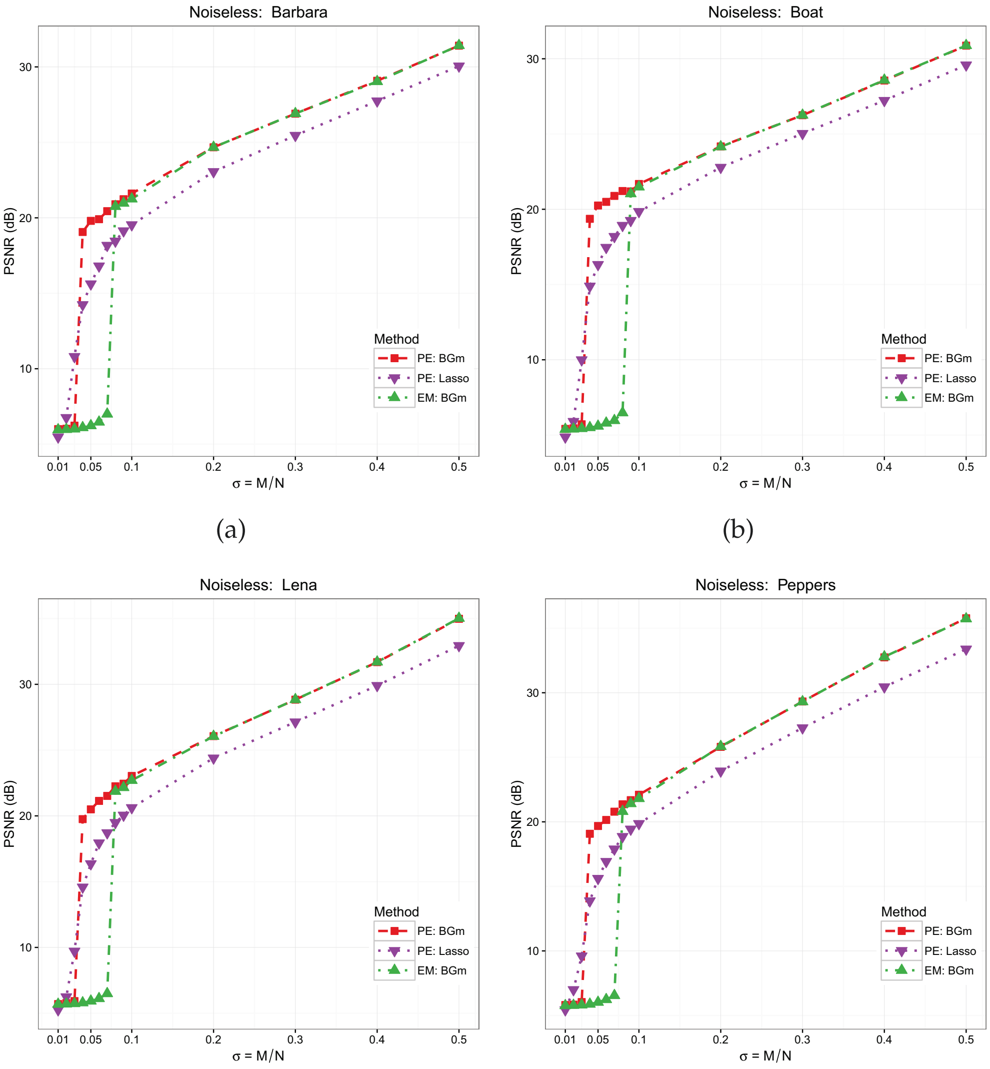
8.3.2 Real image recovery

Real images are considered to be approximately sparse under some proper basis, such as the DCT basis, wavelet basis, etc. Here we compare the recovery performances of PE-BGm-GAMP, PE-Lasso-GAMP, and EM-BGm-GAMP based on varying noiseless and noisy measurements of the 4 real images in Fig. 4.2 as before: Barbara, Boat, Lena, Peppers. The experimental setup is different from the experiments in section 4.2.2: in order to ensure the convergence of the algorithms, we use the Daubechies 6 (db6) wavelet [52] as the sparsifying basis $\boldsymbol{V}$ and i.i.d. random Gaussian matrix $\boldsymbol{U}$ as the sampling matrix. The sensing matrix $\boldsymbol{A}$ is then $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{V}$.

For the noisy recovery, the entries of the noise are generated using i.i.d. Gaussian distribution $\mathcal{N}(0, 1)$, and then scaled to make sure the SNR of the measurement vector $\boldsymbol{y}$ is around 30 dB. The peak-signal-to-noise-ratio (PSNR) of the recovered images from noiseless and noisy measurements are shown in Fig. 8.3 and 8.4 respectively. We can see that both PE-BGm-GAMP and EM-BGm-GAMP perform better than PE-Lasso-GAMP when the sampling ratio $\sigma > 0.1$.

When σ is small, PE-BGm-GAMP and PE-Lasso-GAMP are more robust and generally perform better than EM-BGm-GAMP.

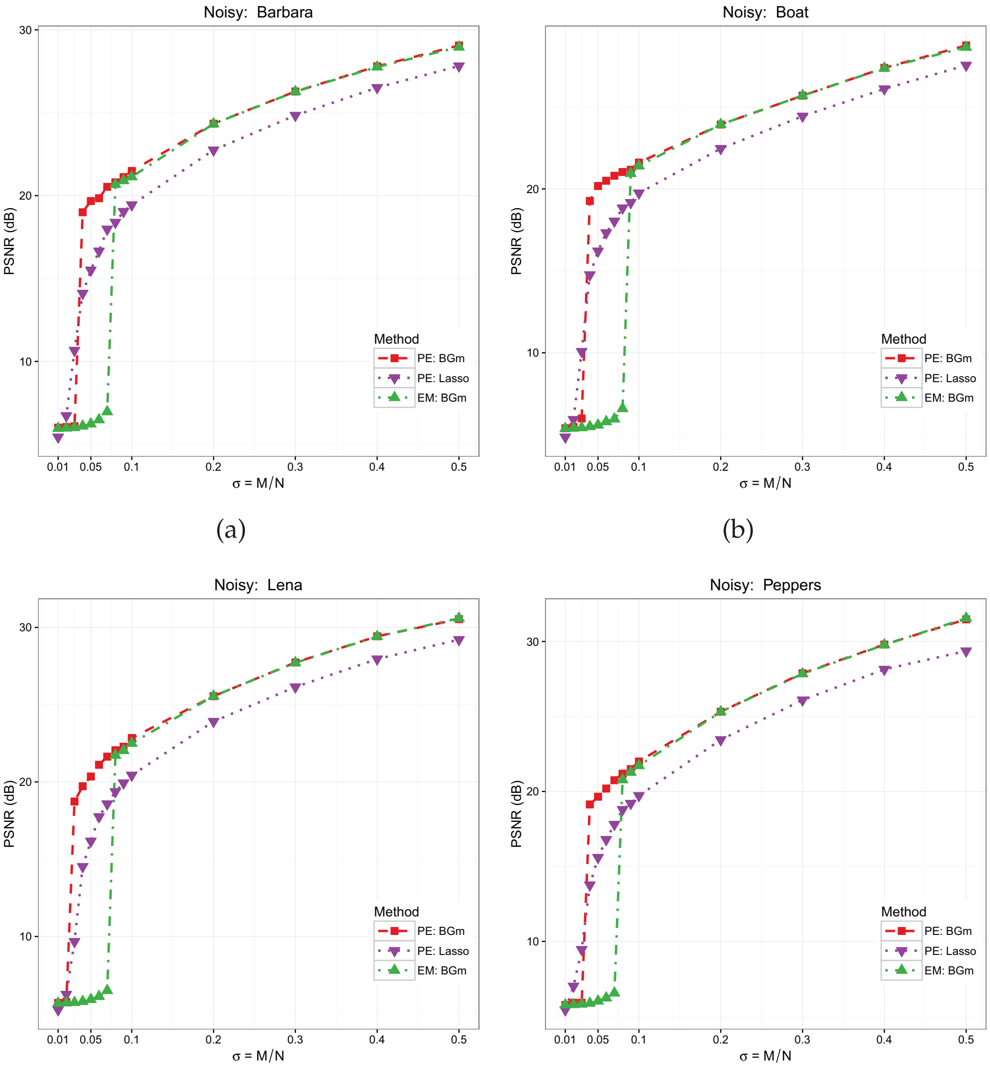
The recovered “Lena” images under the sampling ratio 0.05 and 0.2 from noiseless and noisy measurements are shown in Fig. 8.5-Fig. 8.8 respectively.



(a) (b)

(c) (d)

Figure 8.3: The peak-signal-to-noise-ratio (PSNR) of the recovered images from "noiseless" measurements using different GAMP methods. (a) Barbara; (b) Boat; (c) Lena; (d) Peppers.



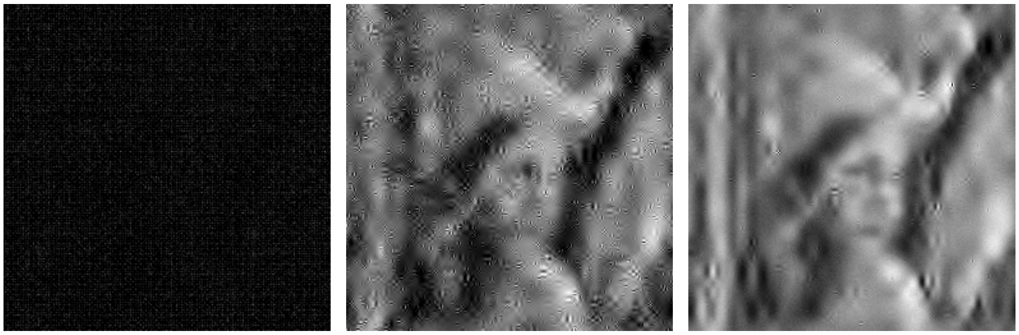
(a)

(b)

(c)

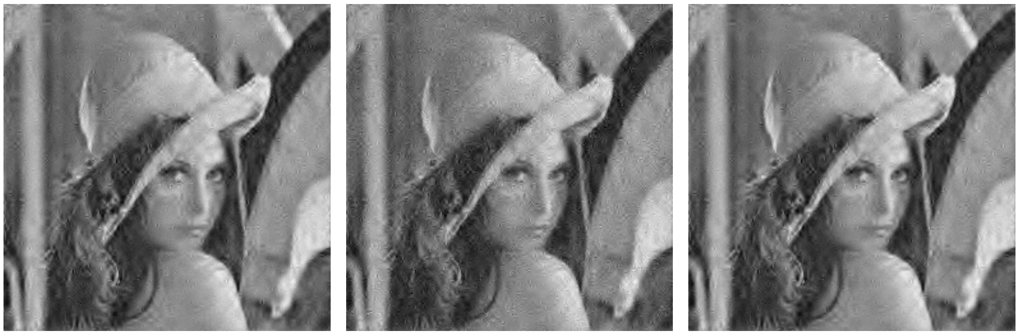
(d)

Figure 8.4: The peak-signal-to-noise-ratio (PSNR) of the recovered images from “noisy” measurements using different GAMP methods. (a) Barbara; (b) Boat; (c) Lena; (d) Peppers.



(a) (b) (c)

Figure 8.5: The recovered lena image using different approaches with a sampling rate of 0.05 from noiseless measurements: a) EM-GAMP (BGm prior) 5.93 dB; (b) PE-GAMP (Laplace prior) 16.36 dB; (c) PE-GAMP (BGm prior) 20.50 dB.



(a) (b) (c)

Figure 8.6: The recovered lena image using different approaches with a sampling rate of 0.2 from noiseless measurements: a) EM-GAMP (BGm prior) 26.05 dB; (b) PE-GAMP (Laplace prior) 24.39 dB; (c) PE-GAMP (BGm prior) 26.06 dB.

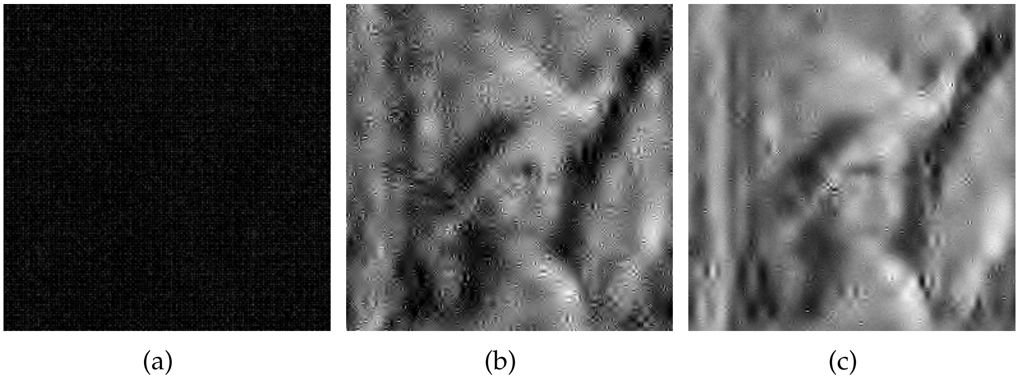


Figure 8.7: The recovered lena image using different approaches with a sampling rate of 0.05 from noisy measurements: a) EM-GAMP (BGm prior) 5.92 dB; (b) PE-GAMP (Laplace prior) 16.17 dB; (c) PE-GAMP (BGm prior) 20.35 dB.

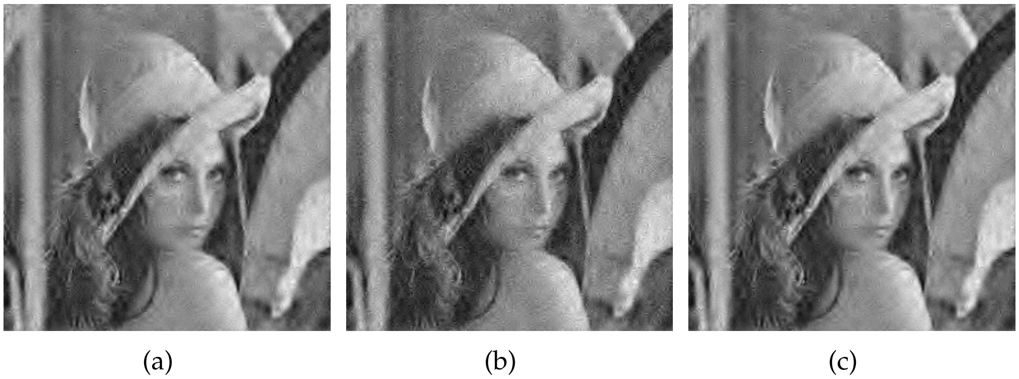


Figure 8.8: The recovered lena image using different approaches with a sampling rate of 0.2 from noisy measurements: a) EM-GAMP (BGm prior) 25.55 dB; (b) PE-GAMP (Laplace prior) 23.91 dB; (c) PE-GAMP (BGm prior) 25.55 dB.

8.3.3 Non-negative sparse coding for image classification

The image classification task typically involves two steps: 1) extracting features, and 2) training a classifier based on such features. In the first step, low-level descriptors, such as SIFT [69], HOG [70], etc, are extracted from local image patches, and then encoded to produce the high-level representations of the images, usually a vector $\boldsymbol{v} \in \mathbb{R}^D$. Here we use the popular Bag-of-Words (BoW) model [71, 72] to encode the low level SIFT descriptors $\boldsymbol{y} \in \mathbb{R}^M$. To do this, we first need to assign each $\boldsymbol{y}$ to one or several "visual words" in some pre-trained dictionary/codebook $\boldsymbol{A}$. In [68], it is shown that this process can be formulated as a sparse coding problem:

$$\begin{aligned} \min_{\boldsymbol{x}} \quad & \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \\ \text{subject to:} \quad & \boldsymbol{x} \geq 0, \quad \boldsymbol{x} \text{ is sparse.} \end{aligned} \tag{8.34}$$

where $\boldsymbol{x}$ is the sparse code of $\boldsymbol{y}$ in the dictionary $\boldsymbol{A}$. In [68], the sparsity constrain on $\boldsymbol{x}$ is enforced with the l_1 norm regularization, i.e. Lasso. Both PE-BGm-GAMP and EM-BGm-GAMP can produce negative sparse codes, and are not suited for the task. Here we can use the proposed PE-BEm-GAMP to solve the above non-negative sparse coding problem.

Specifically, we perform image classification on the popular Caltech-101 dataset [73], which contains 9144 images belonging to 102 classes (101 object classes and a background class). Following the suggestions of the original dataset [73], we randomly select 30 samples per class for training and *up to* 50 samples per class for testing. This process is randomly repeated 10 times and the average classification accuracy is computed as the final result.

Each image is converted to gray-scale and resized to be no larger than 300×300 pixels while preserving the aspect ratio. The normalized local SIFT

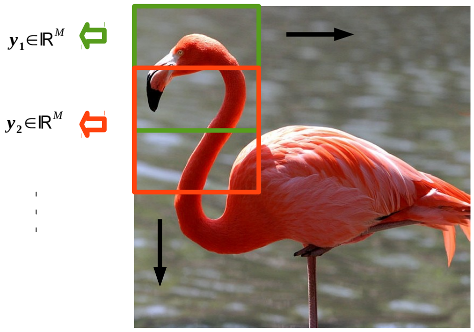


Figure 8.9: Low-level SIFT features are densely sampled from local image patches.

descriptors $\boldsymbol{y} \in \mathbb{R}^{128} \geq 0$ are extracted from 16×16 image patches densely sampled on the grid with a step size of 8 pixels [74], as is shown in Fig. 8.9. We use k-means [75] to train a 128×1024 normalized dictionary $\boldsymbol{A}$. After the non-negative sparse coding, each local image patch is converted to a sparse vector $\boldsymbol{x} \in \mathbb{R}^{1024} \geq 0$. For each image, those sparse vectors are then *max-pooled* using a 3-level spatial pyramid matching [76] to produce a vector $\boldsymbol{v} \in \mathbb{R}^{21504}$. As is usually done, linear support vector machine (SVM) [77, 78] is used as the classifier and the parameters of SVM are chosen using cross-validation. The average classification accuracy across all classes is $60.22 \pm 0.94\%$. The confusion matrix of the classification results is shown in Fig. 8.10.

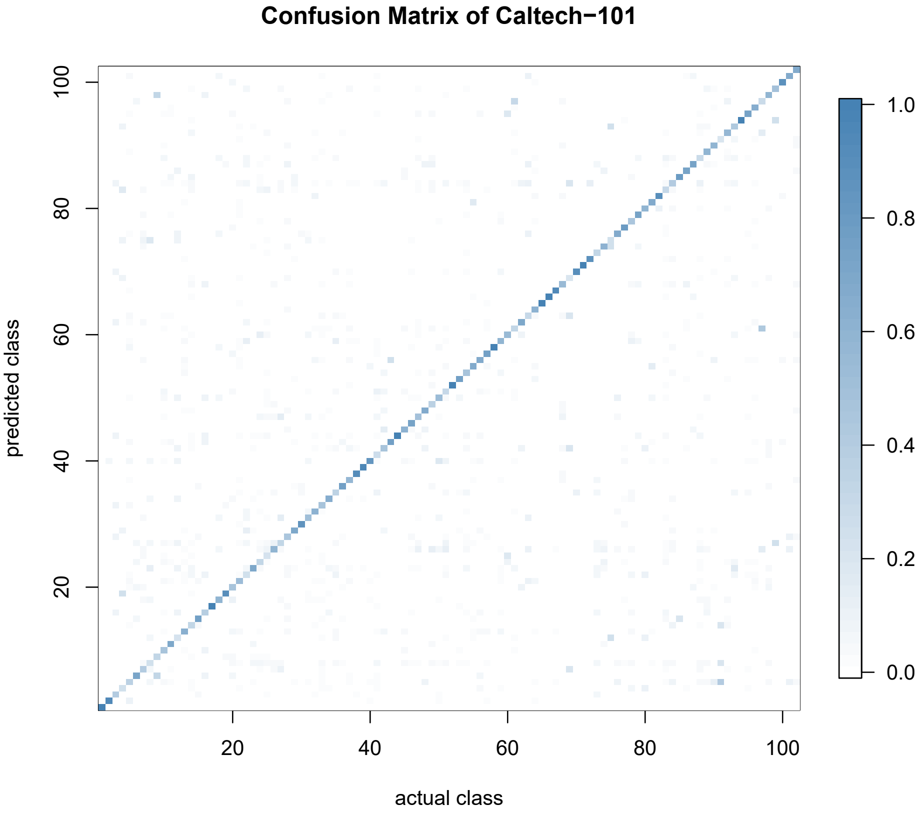


Figure 8.10: The confusion matrix of the classification results on Caltech-101 dataset.

Part IV

Conclusion and future work

Chapter 9

Conclusion and future work

In this dissertation we choose compressive sensing as the tool to process/analyze the large amounts of data generated nowadays, and focus on the sparse signal recovery problem that provides the foundation for various compressive sensing applications. Two major approaches, namely the *sparsity-regularization* approach and the *probabilistic* approach, are considered and explored.

For the sparsity-regularization approach, we propose the Shannon entropy function $h_p(\boldsymbol{x})$ and Rényi entropy function $h_{p,\alpha}(\boldsymbol{x})$ of the sparse signal $\boldsymbol{x}$ as the sparsity regularizers, and give efficient iterative algorithms to perform sparse signal recovery by minimizing them. Experiments on simulated and real data show the advantage the proposed Shannon entropy function minimization and Rényi entropy function minimization approaches have over other state-of-the-art approaches.

We have proved in section 3.3 that the two entropy functions promote sparse solutions in the sense that minimizing them in an orthant of the Euclidean sparse leads to solutions on the boundary of said orthant, i.e. sparser solutions. Extensive experiments conducted in this dissertation not only confirm our proofs, but also reveal the exciting observation that the two entropy functions promote sparsity better than the popular l_1-norm and l_p-norm. This motivates us to explore theoretical guarantees of the advantage over other approaches in the future by establishing error bounds on the recovered signal $\hat{\boldsymbol{x}}$ and providing sufficient conditions under which the successful recovery is warranted.

For the probabilistic approach, we propose the generalized approximate message passing with built-in parameter estimation (PE-GAMP) framework to compute the MMSE or MAP estimation of the sparse signal $\boldsymbol{x}$. By treating the parameters $\{\boldsymbol{\lambda}, \boldsymbol{\theta}\}$ as unknown random variables, we can jointly estimate them along with the signal that follows a variety of complex signal/noise distributions such as the Laplace distribution, Bernoulli-Exponential mixture distribution, etc. Experiments on simulated and real data show that the proposed PE-GAMP is more robust, much simpler and has a wider applicability compared to the EM based parameter estimation method.

In the large system limit as $N \to \infty$, state evolution analysis of the PE-GAMP shows the variables in (7.6,7.18) are able to achieve empirical convergence under the assumptions made in Chapter 7. One notable assumption is that the entries of the sensing matrix $\boldsymbol{A}$ should be i.i.d distributed according to the Gaussian $\mathcal{N}(0, \frac{1}{M})$. In section 8.3.3, the entries of the dictionary $\boldsymbol{A}$ used to perform non-negative sparse coding for image classification violate this assumption, however, the proposed PE-GAMP is still able to achieve convergence and recover the sparse codes. This suggests that some of assumptions made in Chapter 7 could be further relaxed. In order to explore and widen the applicability of the PE-GAMP, we would like to further investigate its empirical convergence behavior with more generalized sensing matrices in our future work.

In Chapter 6 we present the formulations to solve the sparse signal recovery problem where $\boldsymbol{A}$ can be written explicitly in a closed form. In the future, we would like to adapt the PE-GAMP framework to solve other more complicated compressive sensing problems that could not afford such a luxury, for instance, the low-rank matrix completion and robust principal component analysis.

In section 8.3.3, we choose the *k*-means centroids as the dictionary to perform nonnegative sparse coding for the image classification task. In the future we would also like to study how to apply the proposed PE-GAMP to the efficient training of an adaptive dictionary [79, 80], which could benefit popular compressive sensing applications such as sparse coding, image denoising [81], sparse representation classification [8].

References

[1] D. L. Donoho. “Compressed sensing”. In: *IEEE Transactions on Information Theory* 52.4 (2006), pp. 1289–1306.

[2] E. J. Candès and M. B. Wakin. “An introduction to compressive sampling”. In: *IEEE Signal Processing Magazine* 25.2 (2008), pp. 21–30.

[3] M. Lustig, D. L. Donoho, J. M. Santos, and J. M. Pauly. “Compressed sensing MRI”. In: *IEEE Signal Processing Magazine* 25.2 (2008), pp. 72–82.

[4] U. Gamper, P. Boesiger, and S. Kozerke. “Compressed sensing in dynamic MRI”. In: *Magnetic resonance in medicine* 59.2 (2008), pp. 365–373.

[5] M. Trakimas, T. Hancock, and S. Sonkusale. “A compressed sensing analog-to-information converter with edge-triggered SAR ADC core”. In: *IEEE International Symposium on Circuits and Systems*. 2012, pp. 3162–3165.

[6] D. Gangopadhyay, E. G. Allstot, A. M. R. Dixon, K. Natarajan, S. Gupta, and D. J. Allstot. “Compressed sensing analog front-end for bio-sensor applications”. In: *IEEE Journal of Solid-State Circuits* 49.2 (2014), pp. 426–438.

[7] D. Craven, B. McGinley, L. Kilmartin, M. Glavin, and E. Jones. “Compressed sensing for bioelectric signals: a review”. In: *IEEE Journal of Biomedical and Health Informatics* 19.2 (2015), pp. 529–540.

[8] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma. “Robust face recognition via sparse representation”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31.2 (2009), pp. 210–227.

[9] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T. S. Huang, and S. Yan. “Sparse representation for computer vision and pattern recognition”. In: *Proceedings of the IEEE* 98.6 (2010), pp. 1031–1044.

[10] A. Y. Yang, S. S. Sastry, A. Ganesh, and Y. Ma. “Fast ℓ_1-minimization algorithms and an application in robust face recognition: A review”. In: *IEEE International Conference on Image Processing*. 2010, pp. 1849–1852.

[11] Y. Chen, N. M. Nasrabadi, and T. D Tran. “Hyperspectral image classification using dictionary-based sparse representation”. In: *IEEE Transactions on Geoscience and Remote Sensing* 49.10 (2011), pp. 3973–3985.

[12] Y. Chen, N. M. Nasrabadi, and T. D. Tran. “Hyperspectral image classification via kernel sparse representation”. In: *IEEE Transactions on Geoscience and Remote sensing* 51.1 (2013), pp. 217–231.

[13] B. K. Natarajan. “Sparse approximate aolutions to linear systems”. In: *SIAM Journal on Computing* 24.2 (1995), pp. 227–234.

[14] S. G. Mallat and Z. Zhang. “Matching pursuits with time-frequency dictionaries”. In: *IEEE Transactions on Signal Processing* 41.12 (1993), pp. 3397–3415.

[15] J. A. Tropp and A. C. Gilbert. “Signal recovery from random measurements via orthogonal matching pursuit”. In: *IEEE Transactions on Information Theory* 53.12 (2007), pp. 4655–4666.

[16] W. Dai and O. Milenkovic. “Subspace pursuit for compressive sensing signal reconstruction”. In: *IEEE Transactions on Information Theory* 55.5 (2009), pp. 2230–2249.

[17] D. Needella and J.A. Tropp. “CoSaMP: iterative signal recovery from incomplete and inaccurate samples”. In: *Applied and Computational Harmonic Analysis* 26.3 (2009), pp. 301 –321.

[18] E. J. Candès, J. K. Romberg, and T. Tao. “Stable signal recovery from incomplete and inaccurate measurements”. In: *Communications on Pure and Applied Mathematics* 59.8 (2006), pp. 1207–1223. ISSN: 1097-0312.

[19] R. Chartrand. “Exact reconstructions of sparse signals via nonconvex minimization”. In: *IEEE Signal Processing Letters* 14 (2007), pp. 707–710.

[20] S. Foucart and M.-J. Lai. “Sparsest solutions of underdetermined linear systems via lp-minimization for”. In: *Applied and Computational Harmonic Analysis* 26.3 (2009), pp. 395 –407. ISSN: 1063-5203.

[21] E. J. Candès, M. B. Wakin, and S. P. Boyd. “Enhancing sparsity by reweighted ℓ_1- minimization”. In: *Journal of Fourier Analysis and Applications* 14.5 (2008), pp. 877–905.

[22] S. Rangan. “Generalized approximate message passing for estimation with random linear mixing”. In: *IEEE International Symposium on Information Theory (ISIT)*. 2011, pp. 2168–2172.

[23] D. L. Donoho, A. Maleki, and A. Montanari. “Message-passing algorithms for compressed sensing”. In: *Proceedings of the National Academy of Sciences* 106.45 (2009), pp. 18914–18919.

[24] M. J. Wainwright and M. I. Jordan. “Graphical models, exponential families, and variational inference”. In: *Foundations and Trends in Machine Learning* 1.1-2 (2008), pp. 1–305.

[25] F. Krzakala, M. Mézard, F. Sausset, Y. Sun, and L. Zdeborová. “Probabilistic reconstruction in compressed sensing: algorithms, phase diagrams, and threshold achieving matrices”. In: *Journal of Statistical Mechanics: Theory and Experiment* 2012.08 (2012), P08009.

[26] J. P. Vila and P. Schniter. “Expectation-maximization Gaussian-mixture approximate message passing”. In: *IEEE Transactions on Signal Processing* 61.19 (2013), pp. 4658–4672.

[27] A. P. Dempster, N. M. Laird, and D. B. Rubin. “Maximum likelihood from incomplete data via the EM algorithm”. In: *Journal of the Royal Statistical Society* 39.1 (1977), pp. 1–38.

[28] E. Candès and T. Tao. “Decoding by linear programming”. In: *IEEE Transactions on Information Theory* 51.12 (2005), pp. 4203–4215.

[29] E. J. Candès. “The restricted isometry property and its implications for compressed sensing”. In: *Comptes Rendus Mathematique* 346.9âĂŞ10 (2008), pp. 589–592.

[30] A. M. Tillmann and M. E. Pfetsch. “The computational complexity of the restricted isometry property, the nullspace property, and related concepts in compressed sensing”. In: *IEEE Transactions on Information Theory* 60.2 (2014), pp. 1248–1259.

[31] E. J. Candès and J. K. Romberg. “Stable signal recovery from incomplete and inaccurate measurements”. In: *Technical report* (2006).

[32] T. D. Tran. “Sparse recovery and compressed sensing”. In: *Department of ECE, Johns Hopkins University* 520.648 ().

[33] R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin. “A simple proof of the restricted isometry property for random matrices”. In: *Constructive Approximation* 28.3 (2008), pp. 253–263.

[34] T. T. Do, T. D. Tran, and L. Gan. “Fast compressive sampling with structurally random matrices”. In: *IEEE International Conference on Acoustics, Speech and Signal Processing*. 2008, pp. 3369–3372.

[35] T. T. Do, L. Gan, N. H. Nguyen, and T. D. Tran. “Fast and efficient compressive sensing using structurally random matrices”. In: *IEEE Transactions on Signal Processing* 60.1 (2012), pp. 139–154.

[36] H. Rauhut. “Compressive sensing and structured random matrices”. In: *Theoretical Foundations and Numerical Methods for Sparse Recovery* 9 (2010), pp. 1–92.

[37] M. A. T. Figueiredo, R. D. Nowak, and S. J. Wright. “Gradient grojection for sparse reconstruction: application to compressed sensing and other inverse problems”. In: *IEEE Journal of Selected Topics in Signal Processing* 1.4 (2007), pp. 586–597.

[38] A. Beck and M. Teboulle. “A fast iterative shrinkage-thresholding algorithm for linear inverse problems”. In: *SIAM Journal on Imaging Sciences* 2.1 (2009), pp. 183–202.

[39] J. Yang and Y. Zhang. “Alternating direction algorithms for ℓ_1-problems in compressive sensing”. In: *SIAM Journal on Scientific Computing* 33.1 (2011), pp. 250–278.

[40] A. Y. Yang, Z. Zhou, A. G. Balasubramanian, S.S. Sastry, and Y. Ma. “Fast ℓ_1-minimization algorithms for robust face recognition.” In: *IEEE Transactions on Image Processing* 22.8 (2013), pp. 3234–3246.

[41] T. M. Cover and J. A. Thomas. *Elements of information theory*. Wiley-Interscience, 2006.

[42] A. Rényi. “On measures of information and entropy”. In: *Proceedings of the 4th Berkeley symposium on mathematics, statistics and probability*. Vol. 1. 1960, pp. 547–561.

[43] M. Ben-Bassat and J. Raviv. “Rényi’s entropy and the probability of error”. In: *IEEE Transactions on Information Theory* 24.3 (1978), pp. 324–331.

[44] G. M. Bosyk, S. Zozor, F. Holik, M. Portesi, and P. W. Lamberti. “A family of generalized quantum entropies: definition and properties”. In: *Quantum Information Processing* 15.8 (2016), pp. 3393–3420.

[45] P. A. Bromiley, N. A. Thacker, and E. Bouhova-Thacker. *Shannon entropy, Rényi entropy, and information*. Tech. rep. Tina Memo No. 2004-004, 2004.

[46] Geek3 from Wikimedia Commons. *Plot of the Rényi entropies of a bipartite system in the interval [0, 1]:* 2014. URL: http://commons.wikimedia.org/wiki/File:mplwp_reny_entropy012inf.svg.

[47] B. Martinet. “Brève communication. régularisation d’inéquations variationnelles par approximations successives”. fre. In: *ESAIM: Mathematical Modelling and Numerical Analysis - Modélisation Mathématique et Analyse Numérique* 4.R3 (1970), pp. 154–158.

[48] N. Parikh and S. Boyd. “Proximal algorithms”. In: *Foundations and Trends in Optimization* 1.3 (2013), pp. 123–231.

[49] K. Eriksson, D. Estep, and C. Johnson. “Applied mathematics body and soul: Vol I: derivatives and geometry in IR3”. In: *Springer-Verlag Publishing* (2004), pp. 149–164.

[50] E. Hale, W. Yin, and Y. Zhang. “Fixed-point continuation method for ℓ_1-regularized minimization with applications to Compressed sensing”. In: *CAAM Technical Report* (2007), TR07–07.

[51] R. E. Carrillo, J. D. McEwen, D. Van De Ville, J. P. Thiran, and Y. Wiaux. “Sparsity averaging for compressive imaging”. In: *IEEE Signal Processing Letters* 20.6 (2013), pp. 591–594.

[52] I. Daubechies. *Ten lectures on wavelets*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 1992.

[53] S. Setzer. “Split Bregman algorithm, Douglas-Rachford splitting and frame shrinkage”. In: *Scale Space and Variational Methods in Computer Vision: Second International Conference*. Ed. by X.-C. Tai, K. Mørken, M. Lysaker, and K.-A. Lie. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 464–476.

[54] E. J. Candès, X. Li, Y. Ma, and J. Wright. “Robust principal component analysis?” In: *Journal of the ACM* 58.3 (2011), 11:1–11:37.

[55] J. Wright, A. Ganesh, S. Rao, Y. Peng, and Y. Ma. “Robust principal component analysis: exact recovery of corrupted low-rank matrices via convex optimization”. In: *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2009, pp. 2080–2088.

[56] Z. Lin, M. Chen, and Y. Ma. “The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices”. In: *UIUC Technical Report* UILU-ENG-09-2215 (2009).

[57] Z. Lin, A. Ganesh, J. Wright, L. Wu, M. Chen, and Y. Ma. “Fast convex optimization algorithms for exact recovery of a corrupted low-rank matrix”. In: UILU-ENG-09-2214. 2009.

[58] J.-F. Cai, E. J. Candès, and Z. Shen. “A singular value thresholding algorithm for matrix completion”. In: *SIAM Journal on Optimization* 20.4 (2010), pp. 1956–1982.

[59] M. Tao and X. Yuan. “Recovering low-rank and sparse components of matrices from incomplete and noisy observations”. In: *SIAM Journal on Optimization* 21.1 (2011), pp. 57–81.

[60] C. Lu, J. Tang, S. Yan, and Z. Lin. “Generalized nonconvex nonsmooth low-rank minimization”. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 2014, pp. 4130–4137.

[61] D. L. Donoho, A. Maleki, and A. Montanari. “Message-passing algorithms for compressed sensing: I. Motivation and construction”. In: *Proceedings of the Information Theory Workshop* (2010).

[62] M. Bayati and A. Montanari. “The dynamics of message passing on dense graphs, with applications to compressed sensing”. In: *IEEE Transactions on Information Theory* 57.2 (2011), pp. 764–785.

[63] C. Guo and M. E. Davies. “Near optimal compressed sensing without priors: parametric SURE approximate message passing”. In: *IEEE Transactions on Signal Processing* 63.8 (2015), pp. 2130–2141.

[64] C. A. Metzler, A. Maleki, and R. G. Baraniuk. “From denoising to compressed sensing”. In: *IEEE Transactions on Information Theory* 62.9 (2016), pp. 5117–5144.

[65] J. Vila and P. Schniter. “Expectation-maximization Bernoulli-Gaussian approximate message passing”. In: *Conference Record of the 45th Asilomar Conference on Signals, Systems and Computers*. 2011, pp. 799–803.

[66] U. S. Kamilov, S. Rangan, A. K. Fletcher, and M. Unser. “Approximate message passing with consistent parameter estimation and applications to sparse learning”. In: *IEEE Transactions on Information Theory* 60.5 (2014), pp. 2969–2985.

[67] M. D. Iordache, J. M. Bioucas-Dias, and A. Plaza. “Sparse unmixing of hyperspectral data”. In: *IEEE Transactions on Geoscience and Remote Sensing* 49.6 (2011), pp. 2014–2039.

[68] J. Yang, K. Yu, Y. Gong, and T. Huang. “Linear spatial pyramid matching using sparse coding for image classification”. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 2009.

[69] D. G. Lowe. “Object recognition from local scale-invariant features”. In: *Proceedings of the IEEE International Conference on Computer Vision*. Vol. 2. 1999, pp. 1150–1157.

[70] N. Dalal and B. Triggs. “Histograms of oriented gradients for human detection”. In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Vol. 1. 2005, pp. 886–893.

[71] J. Sivic and A. Zisserman. “Video Google: a text retrieval approach to object matching in videos”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2003, pp. 1470–1477.

[72] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray. “Visual categorization with bags of keypoints”. In: *In Workshop on Statistical Learning in Computer Vision, ECCV*. 2004, pp. 1–22.

[73] L. Fei-Fei, R. Fergus, and P. Perona. “Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories”. In: *IEEE CVPR Workshop on Generative-Model Based Vision* (2004).

[74] A. Vedaldi and B. Fulkerson. *VLFeat: An open and portable library of computer vision algorithms*. `http://www.vlfeat.org/`. 2008.

[75] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu. "An efficient k-means clustering algorithm: analysis and implementation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24.7 (July 2002), pp. 881–892.

[76] S. Lazebnik, C. Schmid, and J. Ponce. "Beyond bags of features: spatial pyramid matching for recognizing natural scene categories". In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Vol. 2. 2006, pp. 2169–2178.

[77] C. Cortes and V. Vapnik. "Support-vector networks". In: *Machine Learning* 20.3 (1995), pp. 273–297.

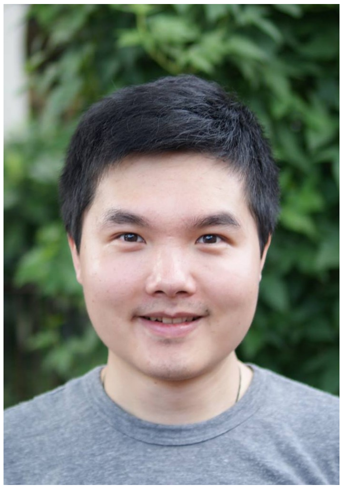
[78] C.-C. Chang and C.-J. Lin. "LIBSVM: A library for support vector machines". In: *ACM Transactions on Intelligent Systems and Technology* 2 (3 2011). Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`, 27:1–27:27.

[79] M. Aharon, M. Elad, and A. Bruckstein. "K -SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation". In: *IEEE Transactions on Signal Processing* 54.11 (2006), pp. 4311–4322.

[80] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. "Online Dictionary Learning for Sparse Coding". In: *Proceedings of the 26th Annual International Conference on Machine Learning*. 2009, pp. 689–696.

[81] M. Elad and M. Aharon. "Image Denoising Via Sparse and Redundant Representations Over Learned Dictionaries". In: *IEEE Transactions on Image Processing* 15.12 (2006), pp. 3736–3745.

Curriculum Vitae



Mr. Shuai Huang was born in Xi'an, Shaanxi province, China on November 24th, 1988. He received the Bachelor of Science degree in Information Science and Technology from Harbin Institute of Technology in July, 2009, and enrolled in the Electrical and Computer Engineering Ph.D. program at the Johns Hopkins University in September, 2009. He received outstanding student scholarship during his undergraduate years at Harbin Institute of Technology, and won both the best graduation thesis and outstanding graduate of Class 2009 awards. His Ph.D. research focuses on exploring the sparse structures embedded in the data using compressive sensing techniques, and his paper has won the IEEE Ganesh N. Ramaswamy Memorial Student Grant in 2013 IEEE International Conference on Acoustics, Speech and Signal Processing.