Solving Linear and Bilinear Inverse Problems using Approximate Message Passing Methods

Dissertation

Presented in Partial Fulfillment of the Requirements for the Degree Doctor of Philosophy in the Graduate School of The Ohio State University

By

Subrata Sarkar, B.Tech., M.S.

Graduate Program in Electrical and Computer Engineering

The Ohio State University

2020

Dissertation Committee:

Dr. Philip Schniter, Advisor

Dr. Lee C. Potter

Dr. Rizwan Ahmad

Abstract

Recovering an unknown vector from its noisy linear measurements is an important problem that arises in many fields. In linear inverse problems, the forward operator is perfectly known, whereas in bilinear inverse problems, the forward operator has some unknown parameters. Most existing recovery algorithms are either slow to converge or give a final solution that is not accurate. In this dissertation, we develop algorithms using Approximate Message Passing (AMP) methods to solve linear and bilinear inverse problems.

First, we consider the computationally efficient Vector Approximate Message Passing (VAMP) algorithm, which is based on the Expectation Propagation framework. VAMP minimizes a cost function, also known as the Gibbs free energy in statistical physics, under moment-matching constraints. It iteratively calls a denoiser that is chosen based on prior knowledge about the unknown vector that we want to recover. VAMP has a remarkable property, that when the sensing matrix is a typical instance of a large right-rotationally invariant random matrix, the per-iteration macroscopic behaviour of VAMP can be exactly predicted by a set of scalar equations that yield the so called state-evolution (SE). The state-evolution can be used to predict the MSE performance of VAMP at each iteration. The SE was first proven for separable Lipschitz denoisers. In this work, we extend the state-evolution to a larger class of non-separable Lipschitz denoisers. Empirical results show that the SE also accurately

predicts VAMP's performance in bilinear problems solved that have been converted to linear problems using the "lifting" technique.

In bilinear inverse problems, the forward operator is written as a linear combination of known matrices with unknown weights. Problems such as dictionary learning, CS with matrix uncertainty, self-calibration, are all instances of bilinear inverse problems. We propose a new algorithm called Bilinear Adaptive Vector Approximate Message Passing (BAd-VAMP). Our method is based on the EM-VAMP algorithm, where Expectation Maximization (EM) is combined with VAMP to estimate the unknown parameters in the likelihood and prior. We show that our BAd-VAMP method is efficient and robust to the conditioning of the forward operator, which is a common problem with algorithms based on the earlier AMP algorithm.

Finally, we consider the problem of magnetic resonance imaging (MRI). MRI is a safe, non-invasive method to capture internal images of our body. MRI's long data-acquisition time is one of its biggest drawbacks. In accelerated MRI, a small number of samples in k-space are acquired and the goal is to accurately reconstruct the image from these few measurements. We develop two algorithms for MRI reconstruction using Approximate Message Passing methods with non-separable denoisers. Our first algorithm is a modification of the existing AMP algorithm, where we propose a scaling in k-space to debias the denoiser input error. Our second algorithm is a modification of VAMP where we propose a robust damping scheme to stabilize VAMP in MRI. We compare our algorithms against conventional plug-and-play algorithms on the fastMRI knee dataset.

This is dedicated to Baba and Ma

Acknowledgments

I am grateful to several people who helped me in completing this dissertation. I would like to first thank my advisor Professor Phil Schniter for his guidance and support throughout my PhD. His creative solutions to difficult problems, work ethic and deep intuition helped me overcome several obstacles in my research. I have learnt to be a better thinker, writer and programmer from him. I am thankful to him for involving me in his research collaborations. This dissertation would not be possible without him.

I would like to thank my committee members, Professor Rizwan Ahmad and Professor Lee Potter, for their helpful feedbacks in my candidacy exam. I thank Tricia Toothman, for organizing social events like the weekly coffee social and monthly meetings with the department chair. My IPS labmates, Nithin Sugavanam, Evan Byrne, Ted Reehorst, Michael Wharton and Saurav Shastri for providing a friendly and supportive community.

I am fortunate to have Bortik Bandyopadhyay and Soumik Mandal as my close friends. I will miss our evening discussions on coding, machine learning and careers. They helped me stay motivated in this unprecedented time, when the entire world is struggling with the pandemic.

Finally, I am indebted to my family for their unconditional love and support. I thank Baba who taught me valuable life lessons and encouraged me to be curious and

think big; Maa, for her love and care; Didi and Kishor da, for always looking after me. I thank my family for trusting my decisions and believing in me. I couldn't have finished my PhD without your love.

Vita

May 2014	B.Tech. Electronics and Communication Engineering, Indian Institute of Technology, Guwahati
May 2016	M.S. Electrical and Computer Engineering, The Ohio State University
2016 - present	Graduate Research Assistant, The Ohio State University

Publications

Research Publications

A. K. Fletcher, P. Pandit, S. Rangan, S. Sarkar, P. Schniter "Plug-in Estimation in High-Dimensional Linear Inverse Problems: A Rigorous Analysis," *Advances in Neural Information Processing Systems*, pp. 7440-7449, 2018.

S. Sarkar, A. K. Fletcher, S. Rangan, P. Schniter "Bilinear Recovery Using Adaptive Vector-AMP," *IEEE Transactions on Signal Processing*, vol. 67, no. 13, pp. 3383-3396, 2019.

S. Rangan, P. Schniter, A. K. Fletcher, S. Sarkar "On the Convergence of Approximate Message Passing with Arbitrary Matrices," *IEEE Transactions on Information Theory*, vol. 65, no. 9, pp. 5339-5351, 2019.

S. Sarkar, A. K. Fletcher, S. Rangan, P. Schniter "Bilinear Recovery Using Adaptive Vector-AMP," *Proc. Asilomar Conf. on Signals, Systems, and Computers*, (Pacific Grove, CA), Nov. 2019.

K.H. Ngo, M. Guillaud, A. Decurninge, S. Yang, S. Sarkar, P. Schniter "Non-Coherent Multi-User Detection Based on Expectation Propagation," *Proc. Asilomar Conf. on Signals, Systems, and Computers*, (Pacific Grove, CA), Nov. 2019.

Fields of Study

Major Field: Electrical and Computer Engineering

Studies in:

Machine Learning
Signal Processing

Table of Contents

List of Tables

Table **Page**

List of Figures

Chapter 1: Introduction

In many problems of interest in science and engineering we need to reliably recover an unknown high-dimensional vector from noisy measurements. The unknown vector could denote model parameters, sensor data, etc. This type of problem commonly arises in several areas of computational imaging such as tomographic imaging, radar, magnetic resonance imaging (MRI), etc. Several technological advancements in the last two decades has made it possible to solve these complex problems by designing efficient computational algorithms.

In these problems, we have some knowledge of the underlying process that generated the observed measurements from the signal, i.e., we have some knowledge of the likelihood function. Assuming the true signal that we are interested to recover is $\boldsymbol{x}$, then the observations $\boldsymbol{y} \in \mathbb{R}^M$ are generated according to the model

$$\boldsymbol{y} = \mathcal{A}(\boldsymbol{x}) + \boldsymbol{w}, \tag{1.1}$$

where $\mathcal{A}$ is the forward operator and $\boldsymbol{w}$ is measurement noise. The goal in (1.1) is to recover the signal $\boldsymbol{x}$ from measurements $\boldsymbol{y}$. These type of problems are known as "inverse problems" as we aim to invert the measurement process to recover the signal that generated it.

The physics of the measurement device decides $\mathcal{A}$. In this dissertation, we primarily focus on linear $\mathcal{A}$. In some applications, we have perfect knowledge of $\mathcal{A}$ and in some cases it could be an approximation of reality. When it is perfectly known, the measurements $\boldsymbol{y}$ are a linear combination of $\boldsymbol{x}$ and such problems are also called "linear inverse" problems. We also consider the case when $\mathcal{A}$ is known up to some unknown parameters $\boldsymbol{b}$. We represent the parametric forward operator as $\mathcal{A}_{\boldsymbol{b}}$. The unknown parameters capture the uncertainy in $\mathcal{A}$, and we aim to jointly recover the signal $\boldsymbol{x}$ and learn the parameters $\boldsymbol{b}$.

Inverse problems are often ill-posed [91] because the forward operator is non-invertible which makes it hard to solve. We need to design recovery algorithms that are robust to noise, model mismatches and ill-conditioning of the problem because in applications such as medical imaging a false diagnosis based on the recovered image can be fatal.

In this dissertation, we develop algorithms for solving linear and bilinear inverse problems using the Approximate Message Passing framework. We develop some theoretical results on the VAMP [73] algorithm. We propose a computationally efficient algorithm called BAdVAMP for solving bilinear inverse problems. In MRI, we modify the AMP [32] and VAMP [73] algorithms by using a robust damping scheme and scaling the k-space measurements and we empirically compare the performance against well-known plug-and-play algorithms such as PnP-ADMM [95] and PnP-FISTA [44].

1.1 Standard Linear Model

In a Standard Linear Model (SLM), $\mathcal{A}$ is a linear operator that can be represented as a matrix $\boldsymbol{A} \in \mathbb{R}^{M \times N}$. We can write (1.1) as,

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{w}, \quad \boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I}). \tag{1.2}$$

Compressive Sensing [19, 31] is a well known application where the number of measurements are less than the unknowns, i.e., $M \ll N$. In this case, (1.2) represents an under-determined system of equations and so it is not possible to recover $\boldsymbol{x}$ unless it has some known structure. In Compressive Sensing, $\boldsymbol{x}$ is known to be sparse, meaning only a subset of its elements are non-zero. One approach to recover $\boldsymbol{x}$ is to solve,

$$\widehat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_0 \quad \text{s.t.} \quad \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2 \leq \sqrt{M}\sigma. \tag{1.3}$$

In (1.3), we seek the sparsest $\boldsymbol{x}$ that is consistent with the measurements by exhaustively searching over all possible solutions. This approach is known to be NP-hard [60] making it intractable even for moderately sized problems. Lasso is a popular alternative approach that recovers $\boldsymbol{x}$ by solving the ℓ_1-regularized least squares problem,

$$\widehat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda \|\boldsymbol{x}\|_1. \tag{1.4}$$

ISTA [27], AMP [32], VAMP [73], etc., are some algorithms for solving the Lasso optimization problem (1.4). When $\boldsymbol{A}$ satisfies the Restricted Isometry property (RIP) [34] and $\boldsymbol{x}$ is sparse enough, there exists theoretical guarantees on the achievable recovery performance of Lasso.

The measurement model in (1.2) is also applicable in several modalities of medical imaging, e.g., Computation Tomography (CT), Magnetic Resonance Imaging (MRI),

etc. In CT, $\boldsymbol{A}$ is the Radon transform and in MRI $\boldsymbol{A}$ is a sub-sampled DFT matrix. Images typically admit a sparse representation under a suitable basis, hence a popular approach to solve these imaging problems is using Compressive Sensing. If $\boldsymbol{x}$ is sparse with respect to basis $\boldsymbol{\Phi}$, then there exists two approaches to recover $\boldsymbol{x}$; i) synthesis, and ii) analysis approach. Formally, the synthesis approach is to solve

$$\widehat{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha}} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{\Phi}\boldsymbol{\alpha}\|_2^2 + \lambda\|\boldsymbol{\alpha}\|_1 \tag{1.5a}$$

$$\Rightarrow \widehat{\boldsymbol{x}} = \boldsymbol{\Phi}\widehat{\boldsymbol{\alpha}}. \tag{1.5b}$$

Whereas in the analysis approach we solve

$$\widehat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{\Phi}^{\mathsf{H}}\boldsymbol{x}\|_1. \tag{1.6}$$

The two approaches are equivalent when $\boldsymbol{\Phi}$ is an orthonormal basis. These methods have led to the advancement in computational imaging for several years. A common drawback is that in many practical problems of interest $\boldsymbol{x}$ may have a more complicated structure than sparsity in a known basis.

To solve (1.4), one usually uses an iterative algorithm where one of the steps in the algorithm involves a proximal operator that enforces prior knowledge of $\boldsymbol{x}$. The proximal operator can also be identified as a signal denoiser. It was shown in [79, 95] that we can solve any linear inverse problem if we have a good denoiser for the class of signals we are interested in recovering. In the age of "Big Data", we may have access to several examples of $\boldsymbol{x}$'s from the signal class that we can use to train a sophisticated denoiser using deep learning [104]. This approach has led to the development of "Plug-and-Play" algorithms achieving superior results.

1.2 Standard Bilinear Model

If the forward operator is not completely known, the inverse problem becomes more challenging. In a bilinear model, the forward operator is a matrix valued linear function,

$$\mathcal{A}_{\boldsymbol{b}} = \boldsymbol{A}_0 + \sum_{i=1}^{Q} b_i \boldsymbol{A}_i, \tag{1.7}$$

where $\{\boldsymbol{A}_i\}$ are known matrices and $\boldsymbol{b}$ are unknown parameters. In some applications, we have tuples of multiple signal and measurements $\{(\boldsymbol{y}_l, \boldsymbol{x}_l)\}_{l=1}^{L}$, also known as the multiple measurement vector (MMV) model. We can stack them column wise to construct the measurement matrix $\boldsymbol{Y}$ and the signal matrix $\boldsymbol{X}$

$$\boldsymbol{Y} = [\boldsymbol{y}_1, \dots, \boldsymbol{y}_L] \in \mathbb{R}^{M \times L} \tag{1.8a}$$

$$\boldsymbol{X} = [\boldsymbol{x}_l, \dots, \boldsymbol{x}_L] \in \mathbb{R}^{N \times L}. \tag{1.8b}$$

The goal is to jointly recover $\boldsymbol{X}$ and $\boldsymbol{b}$ from measurements $\boldsymbol{Y} = \mathcal{A}_{\boldsymbol{b}}(\boldsymbol{X}) + \boldsymbol{W}$, where $\boldsymbol{W}$ is AWGN noise. It is said to be bilinear because the measurements $\boldsymbol{Y}$ are linear in $\boldsymbol{X}$ for fixed $\boldsymbol{b}$ and linear in $\boldsymbol{b}$ for fixed $\boldsymbol{X}$. This problem has applications in matrix completion [21], robust principle component analysis (RPCA) [20], dictionary learning [81], self-calibration [51], blind deconvolution [47], joint-channel/symbol estimation, compressive sensing with matrix uncertainty [105], and many other tasks.

1.3 Outline and Contribution

In this section, we provide a brief overview of our contributions and the outline of this dissertation.

1.3.1 State Evolution of VAMP

VAMP [73] is a computationally efficient algorithm to solve linear inverse problems. At every iteration t, VAMP generates pseudo-measurements $\boldsymbol{r}_1^t$ and scalar precision γ_1^t such that for certain large random $\boldsymbol{A}$, it ensures the statistical model

$$\boldsymbol{r}_1^t = \boldsymbol{x} + \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma_1^t). \tag{1.9}$$

Moreover, the mean squared error (MSE) performance of VAMP can be exactly predicted by a set of scalar equations known as "state evolution" (SE). These simpler SE equations can be used to study the convergence properties and the accuracy of VAMP in the high dimensional regime for different problems. The state evolution of VAMP was earlier proven for separable Lipschitz denoisers in [73]. In chapter 2, we extend the state evolution of VAMP from separable to a larger class of non-separable Lipschitz denoisers. We also consider bilinear problems such as self-calibration [51] and CS with matrix uncertainty [105], and apply the "lifting" method from [3, 23, 28, 51] to transform it into a linear inverse problem. We then use VAMP to solve the lifted problem and empirically show that the state evolution prediction holds.

1.3.2 The BAdVAMP Algorithm

In chapter 3, we propose the Bilinear Adaptive Vector Approximate Message Passing (BAd-VAMP) algorithm to solve bilinear recovery problems. We combine variational inference with maximum likelihood (ML) estimation to develop our algorithm. We assume the prior density of $\boldsymbol{X}$ is known up to some parameters $\boldsymbol{\theta}_x$ and the measurement noise is AWGN with unknown noise precision γ_w. Our goal is to estimate the parameters $\boldsymbol{\Theta} \triangleq \{\boldsymbol{b}, \boldsymbol{\theta}_x, \gamma_w\}$ and recover the unknown matrix $\boldsymbol{X} \in \mathbb{R}^{N \times L}$ from

measurements $\boldsymbol{Y} \in \mathbb{R}^{M \times L}$,

$$\boldsymbol{Y} = \left(\boldsymbol{A}_0 + \sum_{i=1}^{Q} b_i \boldsymbol{A}_i \right) \boldsymbol{X} + \boldsymbol{W} \tag{1.10a}$$

$$\boldsymbol{X} \sim p_{\mathsf{X}}(\cdot; \boldsymbol{\theta}_x) \tag{1.10b}$$

$$\boldsymbol{W} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1/\gamma_w), \tag{1.10c}$$

where $\{\boldsymbol{A}_i\}$ are known matrices. We seek to learn the ML estimate of the parameters $\boldsymbol{\Theta}$ and, under that estimate, compute the minimum mean-squared error (MMSE) estimate of $\boldsymbol{X}$. We solve several bilinear problems such as dictionary learning, calibration in tomography, CS with matrix uncertainty, etc. We demonstrate numerically that the proposed approach is competitive with other state-of-the-art approaches to bilinear recovery, including lifted VAMP and Bilinear GAMP.

1.3.3 VAMP for MRI

In chapter 4, we consider the problem of image reconstruction in Magnetic Resonance Imaging (MRI). MRI is a non-invasive method to generate high resolution internal images of our body. The MRI scanner produces measurements $\boldsymbol{y} \in \mathbb{C}^M$ which are noisy sub-sampled Fourier samples of the unknown image $\boldsymbol{x}$. The goal is to reconstruct the image $\boldsymbol{x} \in \mathbb{C}^N$ from $\boldsymbol{y} \in \mathbb{C}^M$,

$$\boldsymbol{y} = \boldsymbol{S}\boldsymbol{F}\boldsymbol{x} + \boldsymbol{w}, \tag{1.11}$$

where $\boldsymbol{F}$ is the DFT matrix, $\boldsymbol{S}$ is a sampling matrix and $\boldsymbol{w}$ is white Gaussian noise. To generate high quality images and reduce the data acquisition time, most modern commercially available MRI scanners use multiple receiver coils in which case the

measurements $\boldsymbol{y} \in \mathbb{C}^M$ take the form

$$\boldsymbol{y} = \begin{bmatrix} \boldsymbol{S}\boldsymbol{F}\boldsymbol{C}_1 \\ \vdots \\ \boldsymbol{S}\boldsymbol{F}\boldsymbol{C}_K \end{bmatrix} \boldsymbol{x} + \boldsymbol{w} \tag{1.12}$$

where $\{\boldsymbol{C}_k\}_{k=1}^K$ are diagonal matrices representing the coil's sensitivity maps.

We develop algorithms based on the Approximate Message Passing [32] framework using a plug-and-play (PnP) manner. PnP method allows us to use traditional optimization algorithms by calling a powerful image denoiser at every iteration inside the algorithm. The AMP [9] and VAMP [73] algorithms are based on the sensing matrix $\boldsymbol{A}$ in (1.2) being i.i.d. Gaussian and right rotationally invariant, respectively, which generally doesn't include the sensing matrix in MRI. As a result, these algorithms are known to diverge. In this work, we introduce some adaptive damping schemes to improve their convergence properties in MRI. In VAMP, to solve the LMMSE subproblem, we use fast iterative solvers such as conjugate gradient [7] or LSQR [65]. In AMP, motivated by the VDAMP [57] algorithm, we use a similar scaling in the k-space which improves the accuracy of the final result. We use the recently released fastMRI [103] dataset for validation.

1.4 Notation

We use boldface uppercase letters to denote matrices (e.g., $\boldsymbol{X}$), and we use $\mathrm{tr}\{\boldsymbol{X}\}$, $\boldsymbol{X}^{\mathsf{H}}$, $\boldsymbol{X}^{\mathsf{T}}$ to denote the trace, conjugate-transpose and transpose of the matrix $\boldsymbol{X}$ respectively. We denote vectors by boldface lowercase letters (e.g., $\boldsymbol{x}$) and scalars by non-bold letters (e.g., x). Given a matrix $\boldsymbol{X}$, we use $\boldsymbol{x}_l$ to denote the lth column and x_{nl} to denote the element in the nth row and lth column. We use $\|\boldsymbol{X}\|_*$ to denote the nuclear norm of $\boldsymbol{X}$, i.e., sum of its singular values. For a square matrix

$\boldsymbol{X} \in \mathbb{R}^{N\times N}$, we use $\mathrm{diag}(\boldsymbol{X})$ to denote the vector of elements on the diagonal of $\boldsymbol{X}$ and we use $\langle \boldsymbol{X} \rangle \triangleq \frac{1}{N}\sum_{i=1}^{N} x_{ii}$ to denote the average of the diagonal elements of $\boldsymbol{X}$. For a random vector $\boldsymbol{x}$ with density b, we use $\mathbb{E}[f(\boldsymbol{x})|b]$ to denote the expectation of $f(\boldsymbol{x})$, i.e., $\mathbb{E}[f(\boldsymbol{x})|b] = \int f(\boldsymbol{x})b(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}$, and we use $\mathrm{var}[f(\boldsymbol{x})|b]$ for the corresponding variance. We denote the density of a Gaussian random vector with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ as $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. We use $\mathrm{Diag}(\boldsymbol{x})$ to denote the diagonal matrix created from vector $\boldsymbol{x}$.

Chapter 2: Vector Approximate Message Passing

2.1 Introduction

In this chapter, we will discuss a computationally efficient algorithm to solve the linear inverse problems mentioned in chapter 1. Recall that in linear inverse problems, our goal is to recover the unknown vector $\boldsymbol{x} \in \mathbb{R}^N$ from measurements $\boldsymbol{y} \in \mathbb{R}^M$,

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{w}, \tag{2.1}$$

where $\boldsymbol{A} \in \mathbb{R}^{M \times N}$ is a known matrix and $\boldsymbol{w} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \gamma_w^{-1})$ is additive white Gaussian noise of precision γ_w. We assume that the unknown vector $\boldsymbol{x}$ has a known prior density $p_{\mathsf{x}}(\cdot)$ which encodes all our prior knowledge about $\boldsymbol{x}$. We aim to compute the minimum mean-squared error (MMSE) estimate of $\boldsymbol{x}$, i.e.,

$$\widehat{\boldsymbol{x}}_{\mathsf{MMSE}} \triangleq \mathbb{E}[\boldsymbol{x}|\boldsymbol{y}]. \tag{2.2}$$

The expectation is taken over the posterior density

$$p_{\mathsf{x}|\mathsf{y}}(\boldsymbol{x}|\boldsymbol{y}) = \frac{1}{Z(\boldsymbol{y})} p_{\mathsf{x}}(\boldsymbol{x}) p_{\mathsf{y}|\mathsf{x}}(\boldsymbol{y}|\boldsymbol{x}), \tag{2.3}$$

using the likelihood function

$$p_{\mathsf{y}|\mathsf{x}}(\boldsymbol{y}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{y}; \boldsymbol{A}\boldsymbol{x}, \gamma_w^{-1}\boldsymbol{I}) \tag{2.4}$$

and the normalization constant $Z(\boldsymbol{y})$ given as

$$Z(\boldsymbol{y}) = p_{\mathsf{y}}(\boldsymbol{y}) \tag{2.5a}$$

$$= \int p_{\mathsf{x}}(\boldsymbol{x}) p_{\mathsf{y}|\mathsf{x}}(\boldsymbol{y}|\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}. \tag{2.5b}$$

For high dimensional $\boldsymbol{x}$, the integrals in (2.2) and (2.5b) are difficult to compute directly. Thus we need to use alternate methods. Vector Approximate Message Passing (VAMP) [75] is a computationally efficient iterative algorithm to compute the *maximum a posteriori* (MAP) estimate or approximately compute the MMSE estimate of the unknown $\boldsymbol{x}$ in case of infinitely large right rotationally invariant $\boldsymbol{A}$.

2.2 Variational Inference

Variational inference (VI) [98] can be used to bypass the computation of $Z(\boldsymbol{y})$. For example, notice that the true posterior $p_{\mathsf{x}|\mathsf{y}}$ can be recovered by solving the variational optimization (over densities)

$$\widehat{q} = \arg\min_{q} D_{\mathsf{KL}}(q \,\|\, p_{\mathsf{x}|\mathsf{y}}), \tag{2.6}$$

where $D_{\mathsf{KL}}(q \,\|\, p)$ denotes the KL divergence from p to q, i.e.,

$$D_{\mathsf{KL}}(q \,\|\, p) \triangleq \int q(\boldsymbol{x}) \ln \frac{q(\boldsymbol{x})}{p(\boldsymbol{x})} \, \mathrm{d}\boldsymbol{x}. \tag{2.7}$$

Substituting $p_{\mathsf{x}|\mathsf{y}}$ from (2.3) into (2.7) gives us

$$D_{\mathsf{KL}}(q \,\|\, p_{\mathsf{x}|\mathsf{y}}) = D_{\mathsf{KL}}(q \,\|\, p_{\mathsf{x}}) + D_{\mathsf{KL}}(q \,\|\, p_{\mathsf{y}|\mathsf{x}}) + H(q) + \ln Z(\boldsymbol{y}) \tag{2.8}$$

where $H(q) \triangleq -\int q(\boldsymbol{x}) \ln q(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}$ is the differential entropy of the random vector $\boldsymbol{x}$ with density q. Thus it follows from (2.6) and (2.8) that

$$\widehat{q} = \arg\min_{q} \big\{ D_{\mathsf{KL}}(q \,\|\, p_{\mathsf{x}}) + D_{\mathsf{KL}}(q \,\|\, p_{\mathsf{y}|\mathsf{x}}) + H(q) \big\}, \tag{2.9}$$

which bypasses $Z(\boldsymbol{y})$. Still, solving (2.9) is difficult in most cases of interest. The typical response is to constrain q to be in a certain family of distributions, $\mathcal{F}$, by solving

$$\widehat{q} = \arg\min_{q\in\mathcal{F}} \big\{ D_{\mathsf{KL}}(q \,\|\, p_{\mathbf{x}}) + D_{\mathsf{KL}}(q \,\|\, p_{\mathbf{y}|\mathbf{x}}) + H(q) \big\}. \tag{2.10}$$

Some popular choices of $\mathcal{F}$ are,

i. *Mean Field Approximation*: family of separable densities

$$\mathcal{F} = \{q(\cdot) \mid q(\boldsymbol{x}) = \prod_{i=1}^{N} q_i(x_i)\} \tag{2.11}$$

ii. *Exponential Family*: specified by sufficient statistics $\phi(\cdot)$ and parameters η,

$$\mathcal{F} = \{q(\cdot) \mid q(\boldsymbol{x}) = \exp\{\eta^{\mathsf{T}}\phi(\boldsymbol{x}) - \ln Z(\eta)\}\} \tag{2.12}$$

which includes normal, Dirichlet, gamma, etc.

One should choose $\mathcal{F}$ such that (2.10) can be solved analytically and it should be powerful enough to capture the useful components of the true posterior $p_{\mathbf{x}|\mathbf{y}}$. Since a bad choice of $\mathcal{F}$ can compromise $\widehat{q}$ in (2.10) and its mean, we take a different approach.

2.3 Expectation Consistent Approximate Inference

Using the "Gibbs free energy"

$$J(q_1, q_2, q_3) \triangleq D_{\mathsf{KL}}(q_1 \| p_{\mathbf{x}}) + D_{\mathsf{KL}}(q_2 \| p_{\mathbf{y}|\mathbf{x}}) + H(q_3), \tag{2.13}$$

one can rewrite (2.9) as

$$\arg\min_{q_1}\min_{q_2}\max_{q_3} J(q_1, q_2, q_3) \tag{2.14a}$$

$$\text{s.t. } q_1 = q_2 = q_3. \tag{2.14b}$$

Algorithm 1 EC algorithm [75]

initialize:
  $\boldsymbol{r}_1^0, \gamma_1^0$
for $t = 0, \ldots, T_{\max}$ **do**
  $\boldsymbol{x}_1^t = \boldsymbol{g}_1(\boldsymbol{r}_1^t, \gamma_1^t)$
  $1/\eta_1^t = \langle \boldsymbol{g}_1'(\boldsymbol{r}_1^t, \gamma_1^t) \rangle / \gamma_1^t$
  $\gamma_2^t = \eta_1^t - \gamma_1^t$
  $\boldsymbol{r}_2^t = (\eta_1^t \boldsymbol{x}_1^t - \gamma_1^t \boldsymbol{r}_1^t)/\gamma_2^t$
  $\boldsymbol{x}_2^t = \boldsymbol{g}_2(\boldsymbol{r}_2^t, \gamma_2^t)$
  $1/\eta_2^t = \langle \boldsymbol{g}_2'(\boldsymbol{r}_2^t, \gamma_2^t) \rangle / \gamma_2^t$
  $\gamma_1^{t+1} = \eta_2^t - \gamma_2^t$
  $\boldsymbol{r}_1^{t+1} = (\eta_2^t \boldsymbol{x}_2^t - \gamma_2^t \boldsymbol{r}_2^t)/(\eta_2^t - \gamma_2^t)$
end for

We minimize over q_1 and q_2 because $D_{\mathsf{KL}}(q_1 \| p_{\mathbf{x}})$ and $D_{\mathsf{KL}}(q_2 \| p_{\mathbf{y}|\mathbf{x}})$ are convex, while we maximize over q_3 because $H(q_3)$ is concave. But, as discussed earlier, (2.14) is difficult to solve.

In the *expectation consistent approximate inference* (EC) scheme proposed by Opper and Winther in [63], the density constraint (2.14b) is relaxed to the first and second moment matching constraints, i.e.,

$$\arg \min_{q_1} \min_{q_2} \max_{q_3} J(q_1, q_2, q_3) \tag{2.15a}$$

$$\text{s.t.} \quad \mathbb{E}[\boldsymbol{x}|q_1] = \mathbb{E}[\boldsymbol{x}|q_2] = \mathbb{E}[\boldsymbol{x}|q_3] \tag{2.15b}$$

$$\operatorname{tr}\{\operatorname{Cov}[\boldsymbol{x}|q_1]\} = \operatorname{tr}\{\operatorname{Cov}[\boldsymbol{x}|q_2]\} = \operatorname{tr}\{\operatorname{Cov}[\boldsymbol{x}|q_3]\}, \tag{2.15c}$$

where $\mathbb{E}[\boldsymbol{x}|q_i]$ denotes $\mathbb{E}[\boldsymbol{x}]$ under $\boldsymbol{x} \sim q_i$. This yields stationary points of the form

$$q_1(\boldsymbol{x}) \propto p_{\mathbf{x}}(\boldsymbol{x}) \exp\left(-\tfrac{\gamma_1}{2} \|\boldsymbol{x} - \boldsymbol{r}_1\|_2^2\right) \tag{2.16a}$$

$$q_2(\boldsymbol{x}) \propto p_{\mathbf{y}|\mathbf{x}}(\boldsymbol{y}|\boldsymbol{x}) \exp\left(-\tfrac{\gamma_2}{2} \|\boldsymbol{x} - \boldsymbol{r}_2\|_2^2\right) \tag{2.16b}$$

$$q_3(\boldsymbol{x}) \propto \exp\left(-\tfrac{\eta}{2} \|\boldsymbol{x} - \widehat{\boldsymbol{x}}\|_2^2\right), \tag{2.16c}$$

for $\{\boldsymbol{r}_1, \gamma_1, \boldsymbol{r}_2, \gamma_2, \widehat{\boldsymbol{x}}, \eta\}$ that lead to satisfaction of (2.15b)-(2.15c). There are several algorithms whose fixed points are the stationary point of EC (e.g., ADATAP [64], S-AMP [17]).

Various approaches can be used to solve for $\{\boldsymbol{r}_1, \gamma_1, \boldsymbol{r}_2, \gamma_2, \widehat{\boldsymbol{x}}, \eta\}$. One is to alternate the update of $\{(\boldsymbol{r}_1, \gamma_1), (\widehat{\boldsymbol{x}}, \eta)\}$ and $\{(\boldsymbol{r}_2, \gamma_2), (\widehat{\boldsymbol{x}}, \eta)\}$ such that, at each iteration, the moments of q_3 are consistent with either q_1 or q_2. This approach is summarized in Alg. 1 using

$$\boldsymbol{g}_1(\boldsymbol{r}_1, \gamma_1) \triangleq \frac{\int \boldsymbol{x}\, p_{\mathbf{x}}(\boldsymbol{x}) \mathcal{N}(\boldsymbol{x}; \boldsymbol{r}_1, \boldsymbol{I}/\gamma_1)\, \mathrm{d}\boldsymbol{x}}{\int p_{\mathbf{x}}(\boldsymbol{x}) \mathcal{N}(\boldsymbol{x}; \boldsymbol{r}_1, \boldsymbol{I}/\gamma_1)\, \mathrm{d}\boldsymbol{x}} \tag{2.17}$$

$$\boldsymbol{g}_2(\boldsymbol{r}_2, \gamma_2) \triangleq \frac{\int \boldsymbol{x}\, p_{\mathbf{y}|\mathbf{x}}(\boldsymbol{y}|\boldsymbol{x}) \mathcal{N}(\boldsymbol{x}; \boldsymbol{r}_2, \boldsymbol{I}/\gamma_2)\, \mathrm{d}\boldsymbol{x}}{\int p_{\mathbf{y}|\mathbf{x}}(\boldsymbol{y}|\boldsymbol{x}) \mathcal{N}(\boldsymbol{x}; \boldsymbol{r}_2, \boldsymbol{I}/\gamma_2)\, \mathrm{d}\boldsymbol{x}}, \tag{2.18}$$

which, under these definitions of $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$, can be recognized as an instance of *expectation propagation* (EP) [58, Sec. 3.2], [41, 86]. In lines 4 and 8 of Alg. 1, $\boldsymbol{g}_i'(\boldsymbol{r}_i, \gamma_i) \in \mathbb{R}^N$ denotes the diagonal of the Jacobian matrix of $\boldsymbol{g}_i(\cdot, \gamma_i)$ at $\boldsymbol{r}_i$, i.e.,

$$\boldsymbol{g}_i'(\boldsymbol{r}_i, \gamma_i) \triangleq \mathrm{diag}\left(\frac{\partial \boldsymbol{g}_i(\boldsymbol{r}_i, \gamma_i)}{\partial \boldsymbol{r}_i}\right), \tag{2.19}$$

and $\langle \boldsymbol{x} \rangle$ denotes the average coefficient value, i.e., $\langle \boldsymbol{x} \rangle \triangleq \frac{1}{N}\sum_{i=1}^N x_i$ for $\boldsymbol{x} \in \mathbb{R}^N$.

Recall the likelihood function for the standard linear model,

$$p_{\mathbf{y}|\mathbf{x}}(\boldsymbol{y}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{y}; \boldsymbol{A}\boldsymbol{x}, \gamma_w^{-1}\boldsymbol{I}). \tag{2.20}$$

Using the definition of $\boldsymbol{g}_2(\cdot)$ in (2.18) and the likelihood function in (2.20), we see that $\boldsymbol{g}_2$ is performing an LMMSE estimation whose closed form solution is

$$\boldsymbol{g}_2(\boldsymbol{r}_2, \gamma_2) = (\gamma_2 \boldsymbol{I} + \gamma_w \boldsymbol{A}^{\mathsf{T}}\boldsymbol{A})^{-1}(\gamma_2 \boldsymbol{r}_2 + \gamma_w \boldsymbol{A}^{\mathsf{T}}\boldsymbol{y}) \tag{2.21}$$

$$\langle \boldsymbol{g}_2'(\boldsymbol{r}_2, \gamma_2) \rangle = \gamma_2\, \mathrm{tr}\left\{(\gamma_2 \boldsymbol{I} + \gamma_w \boldsymbol{A}^{\mathsf{T}}\boldsymbol{A})^{-1}\right\}/N. \tag{2.22}$$

Meanwhile, the form of $\boldsymbol{g}_1$ depends on $p_{\mathbf{x}}$ through (2.17).

Algorithm 2 EP algorithm (SVD Form) [75]

define:
  Economy SVD: $\boldsymbol{A} = \boldsymbol{U}\,\mathrm{Diag}(\boldsymbol{s})\boldsymbol{V}^{\mathsf{T}} \in \mathbb{R}^{M\times N}$
  $R = \mathrm{rank}(\boldsymbol{A})$
  $\widetilde{\boldsymbol{y}} = \mathrm{Diag}(\boldsymbol{s})^{-1}\boldsymbol{U}^{\mathsf{T}}\boldsymbol{y}$
initialize:
  $\boldsymbol{r}^0, \gamma^0$
for $t = 0, \ldots, T_{\max}$ **do**
  $\boldsymbol{x}^t = \boldsymbol{g}_1(\boldsymbol{r}^t, \gamma^t)$
  $\alpha^t = \langle \boldsymbol{g}_1'(\boldsymbol{r}^t, \gamma^t)\rangle$
  $\widetilde{\boldsymbol{r}}^t = (\boldsymbol{x}^t - \alpha^t\boldsymbol{r}^t)/(1-\alpha^t)$
  $\widetilde{\gamma}^t = \gamma^t(1-\alpha^t)/\alpha^t$
  $\boldsymbol{d}^t = \gamma_w\,\mathrm{Diag}(\gamma_w\boldsymbol{s}^2 + \widetilde{\gamma}^t\boldsymbol{1})^{-1}\boldsymbol{s}^2$
  $\gamma^{t+1} = \widetilde{\gamma}^t R\langle \boldsymbol{d}^t\rangle/(N - R\langle \boldsymbol{d}^t\rangle)$
  $\boldsymbol{r}^{t+1} = \widetilde{\boldsymbol{r}}^t + \frac{N}{R}\boldsymbol{V}\,\mathrm{Diag}(\boldsymbol{d}^t/\langle \boldsymbol{d}^t\rangle)(\widetilde{\boldsymbol{y}} - \boldsymbol{V}^{\mathsf{T}}\widetilde{\boldsymbol{r}}^t)$
end for

Using the definition of $\boldsymbol{g}_2$ in (2.21), we see that in line 7 of Alg. 1 a matrix inversion of complexity $O(N^3)$ is needed at every iteration, which can significantly slow down the algorithm. However, if we were to pre-compute the (economy) SVD $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^{\mathsf{T}}$, then the per iteration cost of VAMP is dominated by $O(RN)$, where R is the rank of $\boldsymbol{A}$. The SVD version of Alg. 1 is summarized in Alg. 2.

Based on the description above, one might wonder whether the EC stationary point $\widehat{\boldsymbol{x}} = \mathbb{E}[\boldsymbol{x}|q_1] = \mathbb{E}[\boldsymbol{x}|q_2] = \mathbb{E}[\boldsymbol{x}|q_3]$ is a good approximation of the true conditional mean $\mathbb{E}[\boldsymbol{x}|\boldsymbol{y}]$, and additionally one might question whether Algorithm 1 converges to this $\widehat{\boldsymbol{x}}$. Both of these concerns were resolved in the VAMP paper [75]. In particular, [75] showed that, when $\boldsymbol{A}$ is right rotationally invariant and asymptotically large, the per-iteration behavior of Alg. 1 with $\boldsymbol{g}_2$ from (2.21) and Lipschitz $\boldsymbol{g}_1$ is exactly predicted by a scalar state evolution. Furthermore, in the case where $\boldsymbol{g}_1$ is matched to $p_{\mathbf{x}}$, the prior of $\boldsymbol{x}$, as in (2.17), and where $\boldsymbol{g}_2$ uses the true AWGN precision $\gamma_w < \infty$

as in (2.18), the MSE of the fixed point $\widehat{\boldsymbol{x}}$ of Alg. 1 was shown in [75] to match the MMSE predicted by the replica method [93]. This replica prediction is conjectured to be correct [77], in which case the $\widehat{\boldsymbol{x}}$ generated by Alg. 1 under (2.17) and (2.21) will be MMSE for infinitely large, right rotationally invariant $\boldsymbol{A}$ when the state evolution has unique fixed points. Note that, for infinitely large i.i.d. $\boldsymbol{A}$, the replica prediction has been proven to be correct [6, 78].

In the sequel, we will refer to Alg. 1 with generic Lipschitz $\boldsymbol{g}_1$ as the VAMP algorithm, noting that it coincides with EP in the special case of Bayesian $\boldsymbol{g}_1$ from (2.17). VAMP is more general than EP because it can be used with denoisers $\boldsymbol{g}_1$ that are MAP [36], or that have no probabilistic interpretation, and still lead to precisely predictable behavior under infinitely large right rotationally invariant $\boldsymbol{A}$ [35, 75].

In Algorithm 1, lines 6 and 10 perform what is referred to as *Onsager correction* in statistical physics [90]. It ensures that, under a certain large random $\boldsymbol{A}$, the VAMP quantity $(\boldsymbol{r}_1^t, \gamma_1^t)$ obeys the following AWGN statistical model at every iteration:

$$\boldsymbol{r}_1^t = \boldsymbol{x}_0 + \mathcal{N}(\boldsymbol{0}, \gamma_1^t \boldsymbol{I}), \tag{2.23}$$

where $\boldsymbol{x}_0$ is the true vector we are interested in recovering in (2.1). This AWGN-like property is one of the main reasons for VAMP's fast convergence properties. There are several ways to interpret Algorithm 1; one can interpret it as an instance of expectation propagation (EP) [63, 89], or as an instance of belief-propagation (BP) on a non-loopy factor graph with vector-valued nodes [73], or as a variant of alternating direction method of multipliers (ADMM) [14] involving two dual updates with adaptive step size [36]. Another intuitive interpretation of Algorithm 1 is the following:

(i) In line 3, the function $\boldsymbol{g}_1$ estimates the signal assuming from AWGN-corrupted measurements $\boldsymbol{r}_1^t = \boldsymbol{x}_0 + \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma_1^t)$.

(ii) Lines 5 & 6 compute $\boldsymbol{r}_2^t, \gamma_2^t$ to yield the model $\boldsymbol{x}_0 = \boldsymbol{r}_2^t + \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma_2^t)$.

(iii) In line 7, $\boldsymbol{g}_2$ performs LMMSE estimation of $\boldsymbol{x}$ from $\boldsymbol{y}$ under the model

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}_0 + \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma_w), \quad \boldsymbol{x}_0 = \boldsymbol{r}_2^t + \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma_2^t).$$

(iv) Lines 9 & 10 compute $\boldsymbol{r}_1^{t+1}, \gamma_1^{t+1}$ such that the AWGN like model in step (i) holds.

The above discussion suggests that $\boldsymbol{g}_1$ is performing a denoising operation. In the next section, we provide a rigorous analysis of the AWGN-like property of VAMP.

2.4 State Evolution

Under some conditions, the noise precision in the AWGN model in (2.23) can be exactly predicted by a set of scalar recursive equations known as the "state-evolution" (SE). This state-evolution formalism allows us to predict the MSE performance of VAMP at every iteration and study its asymptotic convergence properties. In this section, we introduce the SE formalism in VAMP and the conditions under which it is guaranteed to be true. The SE of VAMP was first proven for separable $\boldsymbol{g}_1$ in [73]. The following results are from our work in [35], where we extended the SE analysis to the non-separable case. We first briefly describe the "large system limit" model and then state the main results.

2.4.1 Large System Limit

We consider a sequence of problems defined by the problem size, N. For each N, we assume $\boldsymbol{x}_0 \in \mathbb{R}^N$ is a "true" vector that we are interested in recovering from measurements $\boldsymbol{y} \in \mathbb{R}^M$ generated according the linear model,

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}_0 + \boldsymbol{w}, \quad \boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \gamma_{w0}^{-1}\boldsymbol{I}). \tag{2.24}$$

The "true" noise precision is γ_{w0}, which could be different from the assumed noise precision used in (2.21) and (2.22). Here, we assume $M = N$ without loss of generality. We can easily extend the following results to the case $M \neq N$, by zero padding $\boldsymbol{s}$ [73]. We assume the SVD of $\boldsymbol{A}$ takes the form,

$$\boldsymbol{A} = \boldsymbol{U}\,\mathrm{Diag}(\boldsymbol{s})\boldsymbol{V}^{\mathsf{T}}, \quad \boldsymbol{s} = (s_1, \ldots, s_N), \tag{2.25}$$

where $\boldsymbol{U} \in \mathbb{R}^{N\times N}$ is any arbitrary orthogonal matrix, $\boldsymbol{s}$ is i.i.d. with components $s_i \in [0, s_{\mathsf{max}}]$ almost surely, and $\boldsymbol{V} \in \mathbb{R}^{N\times N}$ is Haar distributed. The latter means that $\boldsymbol{V}$ is uniformly distributed on the set of $N \times N$ orthogonal matrices, which implies $\boldsymbol{A}$ is *right rotationally invariant*, meaning if we right multiply $\boldsymbol{A}$ by any deterministic orthogonal matrix $\boldsymbol{V}_0$ then the distribution of $\boldsymbol{A}\boldsymbol{V}_0$ is the same as $\boldsymbol{A}$, i.e., $\boldsymbol{A}\boldsymbol{V}_0 \overset{\mathsf{d}}{=} \boldsymbol{A}$. We also assume that the quantities $\{\boldsymbol{x}_0, \boldsymbol{w}, \boldsymbol{s}, \boldsymbol{V}\}$ are all independent. In the large system limit, we let $N \to \infty$ and study the performance of VAMP averaged over the random quantities $\{\boldsymbol{w}, \boldsymbol{s}, \boldsymbol{V}\}$. We next introduce two important definitions the denoiser $\boldsymbol{g}_1$ needs to satisfy.

Definition 1. *A denoiser $\boldsymbol{g}(\cdot,\cdot)$ is uniformly Lipschitz continuous if there exists constants A, B and $C > 0$, such that $\forall \boldsymbol{r}_1, \boldsymbol{r}_2 \in \mathbb{R}^N$, $\gamma_1, \gamma_2 > 0$ and N, we have,*

$$\|\boldsymbol{g}(\boldsymbol{r}_2, \gamma_2) - \boldsymbol{g}(\boldsymbol{r}_1, \gamma_1)\| \leq (A + B|\gamma_2 - \gamma_1|)\|\boldsymbol{r}_2 - \boldsymbol{r}_1\| + C\sqrt{N}|\gamma_2 - \gamma_1|. \tag{2.26}$$

Definition 2. *A denoiser $\boldsymbol{g}(\cdot,\cdot)$ and vector $\boldsymbol{u} \in \mathbb{R}^N$ are said to be convergent under Gaussian noise if the following condition holds: Let $\boldsymbol{z}_1, \boldsymbol{z}_2 \in \mathbb{R}^N$ be two i.i.d. sequences with $\begin{bmatrix} \boldsymbol{z}_1 \\ \boldsymbol{z}_2 \end{bmatrix} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{S} \otimes \boldsymbol{I})$ for some positive definite matrix $\boldsymbol{S} \in \mathbb{R}^{2\times 2}$. Then, all the following limits must exist almost surely $\forall\ \gamma_1, \gamma_2 > 0$, $\boldsymbol{S}$:*

$$\lim_{N\to\infty} \frac{1}{N} \boldsymbol{g}(\boldsymbol{u}+\boldsymbol{z}_1, \gamma_1)^{\mathsf{T}} \boldsymbol{g}(\boldsymbol{u}+\boldsymbol{z}_2, \gamma_2), \quad \lim_{N\to\infty} \frac{1}{N} \boldsymbol{g}(\boldsymbol{u}+\boldsymbol{z}_1, \gamma_1)^{\mathsf{T}} \boldsymbol{u}, \tag{2.27a}$$

$$\lim_{N\to\infty} \frac{1}{N} \boldsymbol{u}^{\mathsf{T}} \boldsymbol{z}_1, \quad \lim_{N\to\infty} \frac{1}{N} \|\boldsymbol{u}\|^2 \tag{2.27b}$$

$$\lim_{N\to\infty} \langle \nabla \boldsymbol{g}(\boldsymbol{u}+\boldsymbol{z}_1, \gamma_1) \rangle = \frac{1}{N S_{12}} \boldsymbol{g}(\boldsymbol{u}+\boldsymbol{z}_1, \gamma_1)^{\mathsf{T}} \boldsymbol{z}_2. \tag{2.27c}$$

Moreover, the limiting values are continuous in $\boldsymbol{S}$, γ_1 and γ_2.

Several denoisers used in practice satisfy the above definitions. In Appendix A.1, we show that the singular value thresholding denoiser, commonly used in low rank matrix recovery [16], satisfy (2.26) and (2.27a).

Given a denoiser satisfying definitions 1 and 2, and an assumed noise precision γ_w, we run Algorithm 1 with the initialization,

$$\boldsymbol{r}_1^0 = \boldsymbol{x}_0 + \mathcal{N}(\boldsymbol{0}, \tau_1^0 \boldsymbol{I}), \tag{2.28}$$

for some initial error variance τ_1^0. In addition, we assume

$$\lim_{N\to\infty} \gamma_1^0 = \overline{\gamma}_1^0, \tag{2.29}$$

almost surely for some $\overline{\gamma}_1^0 \geq 0$. To measure the performance of VAMP, we define two error functions, $\mathcal{E}_1(\cdot)$ and $\mathcal{E}_2(\cdot)$. The error function $\mathcal{E}_1(\cdot)$,

$$\mathcal{E}_1(\gamma_1, \tau_1) \triangleq \lim_{N\to\infty} \frac{1}{N} \|\boldsymbol{g}_1(\boldsymbol{x}_0+\boldsymbol{z}_1, \gamma_1) - \boldsymbol{x}_0\|^2 \tag{2.30a}$$

$$\boldsymbol{z}_1 \sim \mathcal{N}(\boldsymbol{0}, \tau_1 \boldsymbol{I}) \tag{2.30b}$$

characterizes the MSE of the denoiser $\boldsymbol{g}_1(\cdot,\cdot)$ whereas $\mathcal{E}_2(\cdot)$,

$$\mathcal{E}_2(\gamma_2,\tau_2) \triangleq \lim_{N\to\infty} \frac{1}{N}\,\mathbb{E}\,\|\boldsymbol{g}_2(\boldsymbol{x}_0+\boldsymbol{z}_2,\gamma_2)-\boldsymbol{x}_0\|^2 \tag{2.31a}$$

$$\boldsymbol{z}_2 \sim \mathcal{N}(\boldsymbol{0},\tau_2\boldsymbol{I}) \tag{2.31b}$$

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}_0 + \mathcal{N}(\boldsymbol{0},\gamma_{w0}^{-1}\boldsymbol{I}). \tag{2.31c}$$

characterizes the MSE of the linear estimator $\boldsymbol{g}_2(\cdot,\cdot)$. The limit (2.30b) exists almost surely due to the assumption of $\boldsymbol{g}_1(\cdot,\cdot)$ being convergent under Gaussian noise. We also define the *sensitivity functions* as

$$\mathcal{A}_i(\gamma_i,\tau_i) := \lim_{N\to\infty} \langle \nabla \boldsymbol{g}_i(\boldsymbol{x}_0+\boldsymbol{z}_i,\gamma_i)\rangle, \quad \boldsymbol{z}_i \sim \mathcal{N}(\boldsymbol{0},\tau_i\boldsymbol{I}). \tag{2.32}$$

2.4.2 State Evolution of Non-Separable VAMP

Consider the following error vectors in VAMP,

$$\boldsymbol{p}^t \triangleq \boldsymbol{r}_1^t - \boldsymbol{x}_0 \tag{2.33a}$$

$$\boldsymbol{q}^t \triangleq \boldsymbol{V}^{\mathsf{T}}(\boldsymbol{r}_2^t - \boldsymbol{x}_0). \tag{2.33b}$$

At iteration t, the error in the input to the denoiser $\boldsymbol{g}_1(\cdot,\cdot)$ is $\boldsymbol{p}^t$ and the transformed error in the input to the LMMSE estimator $\boldsymbol{g}_2(\cdot,\cdot)$ is $\boldsymbol{q}^t$. In the state evolution analysis, we show that asymptotically these errors are Gaussian. Moreover, we can also exactly predict the error precisions (2.33) via the following recursive scalar *state evolution* equations initialized with $t=0$, τ_1^0 in (2.28) and $\overline{\gamma}_1^0$ in (2.29):

$$\overline{\alpha}_1^t = \mathcal{A}_1(\overline{\gamma}_1^t,\tau_1^t), \quad \overline{\eta}_1^t = \frac{\overline{\gamma}_1^t}{\overline{\alpha}_1^t}, \quad \overline{\gamma}_2^t = \overline{\eta}_1^t - \overline{\gamma}_1^t \tag{2.34a}$$

$$\tau_2^t = \frac{1}{(1-\overline{\alpha}_1^t)^2}\left[\mathcal{E}_1(\overline{\gamma}_1^t,\tau_1^t) - (\overline{\alpha}_1^t)^2\tau_1^t\right], \tag{2.34b}$$

$$\overline{\alpha}_2^t = \mathcal{A}_2(\overline{\gamma}_2^t,\tau_2^t), \quad \overline{\eta}_2^t = \frac{\overline{\gamma}_2^t}{\overline{\alpha}_2^t}, \quad \overline{\gamma}_1^{t+1} = \overline{\eta}_2^t - \overline{\gamma}_2^t \tag{2.34c}$$

$$\tau_1^{t+1} = \frac{1}{(1-\overline{\alpha}_2^t)^2}\left[\mathcal{E}_2(\overline{\gamma}_2^t,\tau_2^t) - (\overline{\alpha}_2^t)^2\tau_2^t\right] \tag{2.34d}$$

We can now state our main result.

Theorem 1. *Assume that the true vector $\boldsymbol{x}_0$ and the denoiser $\boldsymbol{g}_2(\cdot,\cdot)$ satisfies definitions 1 and 2. Also assume that for all iterations t, the solution $\overline{\alpha}_1^t$ from the SE equations (2.34) satisfies $\overline{\alpha}_1^t \in (0,1)$ and $\overline{\gamma}_{ik} > 0$. Then,*

(a) The error vectors $\boldsymbol{p}^t$ and $\boldsymbol{q}^t$ defined in (2.33) satisfy,

$$\lim_{N\to\infty} \frac{1}{N}\|\boldsymbol{p}^t - \widetilde{\boldsymbol{p}}^t\|^2 = 0 \tag{2.35}$$

$$\lim_{N\to\infty} \frac{1}{N}\|\boldsymbol{q}^t - \widetilde{\boldsymbol{q}}^t\|^2 = 0 \tag{2.36}$$

where, $\widetilde{\boldsymbol{p}}^t \sim \mathcal{N}(\boldsymbol{0}, \tau_1^t \boldsymbol{I})$, $\widetilde{\boldsymbol{q}}^t \sim \mathcal{N}(\boldsymbol{0}, \tau_2^t \boldsymbol{I})$ and $\widetilde{\boldsymbol{p}}^t$, $\widetilde{\boldsymbol{q}}^t$ are independent.

(b) For any iteration t, and $i = 1, 2$, we have, almost surely

$$\lim_{N\to\infty} \frac{1}{N}\|\boldsymbol{x}_i^t - \boldsymbol{x}_0\|^2 = \frac{1}{\overline{\eta}_i^t}, \quad \lim_{N\to\infty} (\alpha_i^t, \eta_i^t, \gamma_i^t) = (\overline{\alpha}_i^t, \overline{\eta}_i^t, \overline{\gamma}_i^t). \tag{2.37}$$

Proof. See [35]. □

Theorem 1 thus shows that the error vectors $\boldsymbol{p}^t$ and $\boldsymbol{q}^t$ in (2.33) are approximately i.i.d. Gaussian in a certain high dimensional limit. This result is an extension of the main result in [73] from separable denoisers to non-separable denoisers. The error variances and the mean squared error (MSE) of the VAMP estimates $(\boldsymbol{x}_1^t, \boldsymbol{x}_2^t)$ can be exactly predicted by (2.34), which are the same as the SE equations in [73].

2.5 Numerical Experiments

2.5.1 Image Recovery via Non-Separable VAMP

We consider the problem of recovering an image $\boldsymbol{x}_0 \in \mathbb{R}^N$ from measurements $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}_0 + \boldsymbol{w} \in \mathbb{R}^M$ when $M \ll N$. This problem is commonly seen in computed

Figure 2.1: Image reconstruction using BM3D-VAMP at undersampling ratio of 0.5.

tomography, magnetic resonance imaging, etc., where $\boldsymbol{A}$ depends on the application. In this experiment, we provide empirical evidence of the state evolution in VAMP. We randomly generated $\boldsymbol{A} \in \mathbb{R}^{M \times N}$ from i.i.d. $\mathcal{N}(0,1)$ for $M/N = 0.5$ and set the noise precision γ_w that achieved 40 dB signal to noise ratio (SNR),

$$\mathsf{SNR} \triangleq \frac{\mathbb{E}\,\|\boldsymbol{A}\boldsymbol{x}_0\|^2}{\mathbb{E}\,\|\boldsymbol{w}\|^2}. \tag{2.38}$$

We used the *Barbara* image of size 128×128 (i.e., $N = 128^2$) as the true vector $\boldsymbol{x}_0$ and BM3D [26] as the denoiser $\boldsymbol{g}_1(\cdot,\cdot)$ in VAMP. Figure 2.2 shows the PSNR[1] predicted by the state evolution and the PSNR in VAMP at every iteration averaged over 10 random draws of $\boldsymbol{A}$ and $\boldsymbol{w}$. The state evolution accurately predicts the PSNR in VAMP at every iteration.

[1]We define the PSNR as $\frac{\max_i |x_{0,i}|}{\|\widehat{\boldsymbol{x}} - \boldsymbol{x}_0\|}$.

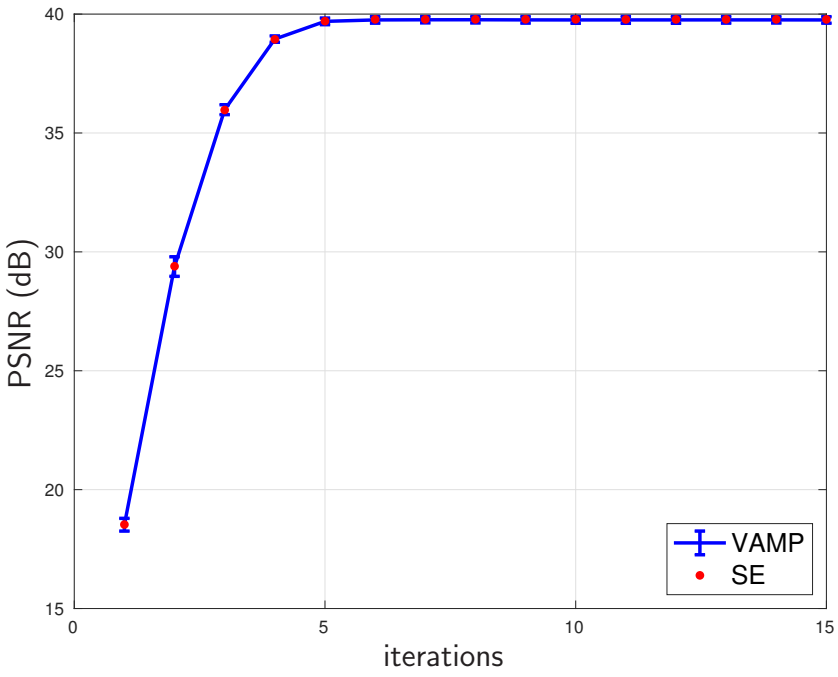

Figure 2.2: SE prediction in VAMP for image recovery

2.5.2 Bilinear Estimation via Lifting

We now use the standard linear inverse model (2.1) to tackle problems in *bilinear* estimation through a technique known as "lifting" [3, 23, 28, 51]. In doing so, we are motivated by applications like blind deconvolution [11], self-calibration [51], compressed sensing (CS) with matrix uncertainty [105] and joint channel-symbol estimation [88]. All cases yield measurements $\boldsymbol{y}$ of the form

$$\boldsymbol{y} = \big(\textstyle\sum_{l=1}^{L} b_l \boldsymbol{A}_l\big)\boldsymbol{c} + \boldsymbol{w} \in \mathbb{R}^M, \tag{2.39}$$

where $\{\boldsymbol{A}_l\}_{l=1}^{L}$ are known, $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma_w)$, and the objective is to recover both $\boldsymbol{b} \triangleq [b_1, \ldots, b_L]^{\mathsf{T}}$ and $\boldsymbol{c} \in \mathbb{R}^P$. This bilinear problem can be "lifted" into a linear

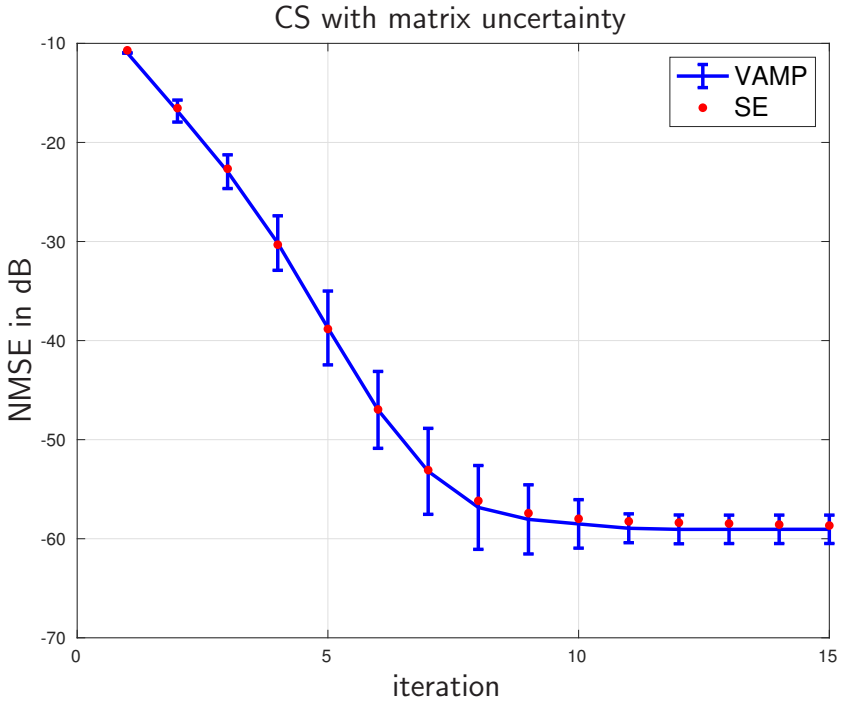

Figure 2.3: SE prediction in VAMP for CS with matrix uncertainty

problem of the form (2.1) by setting

$$\boldsymbol{A} = [\boldsymbol{A}_1, \ldots, \boldsymbol{A}_L] \in \mathbb{R}^{M \times LP} \tag{2.40a}$$

$$\boldsymbol{x} = \mathrm{vec}(\boldsymbol{c}\boldsymbol{b}^{\mathsf{T}}) \in \mathbb{R}^{LP}, \tag{2.40b}$$

where $\mathrm{vec}(\boldsymbol{X})$ vectorizes $\boldsymbol{X}$ by concatenating its columns. When $\boldsymbol{b}$ and $\boldsymbol{c}$ are i.i.d. with known priors, the MMSE denoiser $\boldsymbol{g}_1(\boldsymbol{r}_1, \gamma_1) = \mathbb{E}(\boldsymbol{x}|\boldsymbol{r}_1 = \boldsymbol{x} + \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma_1))$ can be implemented near-optimally by the rank-one AMP algorithm from [72] (see also [30, 48, 55]), with divergence estimated as in [56].

We first consider *CS with matrix uncertainty* [105], where b_1 is known. For these experiments, we generated the unknown $\{b_l\}_{l=2}^{L}$ as i.i.d. $\mathcal{N}(0,1)$ and the unknown $\boldsymbol{c} \in \mathbb{R}^P$ as K-sparse with $\mathcal{N}(0,1)$ nonzero entries. For $b_1 = \sqrt{20}$, $L = 11$, $P = 256$, $K = 10$, i.i.d. $\mathcal{N}(0,1)$ matrices $\{\boldsymbol{A}_i\}$, and SNR $= 40$ dB, Fig. 2.3 shows that the

NMSE on $\boldsymbol{x}$ of lifted VAMP is very close to its SE prediction. We then compared lifted VAMP to PBiGAMP from [66], which applies AMP directly to the (non-lifted) bilinear problem, and to WSS-TLS from [105], which uses non-convex optimization. We also compared to MMSE estimation of $\boldsymbol{b}$ under oracle knowledge of $\boldsymbol{c}$, and MMSE estimation of $\boldsymbol{c}$ under oracle knowledge of support($\boldsymbol{c}$) and $\boldsymbol{b}$. Fig. 2.4 shows the normalized MSE on $\boldsymbol{b}$ and $\boldsymbol{c}$ versus sampling ratio M/P,

$$\mathsf{NMSE}(\widehat{\boldsymbol{b}}) \triangleq \frac{\mathbb{E}\,\|\widehat{\boldsymbol{b}}-\boldsymbol{b}\|^2}{\mathbb{E}\,\|\boldsymbol{b}\|^2}. \tag{2.41}$$

This figure demonstrates that lifted VAMP and PBiGAMP perform close to the oracles and much better than WSS-TLS.

Although lifted VAMP performs similarly to PBiGAMP in Fig. 2.4, its advantage over PBiGAMP becomes apparent with non-i.i.d. $\boldsymbol{A}$. For illustration, we repeated the previous experiment, but with $\boldsymbol{A}$ constructed using the SVD $\boldsymbol{A} = \boldsymbol{U}\,\mathrm{Diag}(\boldsymbol{s})\boldsymbol{V}^{\mathsf{T}}$ with Haar distributed $\boldsymbol{U}$ and $\boldsymbol{V}$ and geometrically spaced $\boldsymbol{s}$. Also, to make the problem more difficult, we set $b_1 = 1$. Figure 2.5 shows the normalized MSE on $\boldsymbol{b}$ and $\boldsymbol{c}$ versus cond($\boldsymbol{A}$) at $M/P = 0.6$. There it can be seen that lifted VAMP is much more robust than PBiGAMP to the conditioning of $\boldsymbol{A}$.

We next consider the *self-calibration* problem [51], where the measurements take the form

$$\boldsymbol{y} = \mathrm{Diag}(\boldsymbol{H}\boldsymbol{b})\boldsymbol{\Psi}\boldsymbol{c} + \boldsymbol{w} \in \mathbb{R}^M. \tag{2.42}$$

Here the matrices $\boldsymbol{H} \in \mathbb{R}^{M\times L}$ and $\boldsymbol{\Psi} \in \mathbb{R}^{M\times P}$ are known and the objective is to recover the unknown vectors $\boldsymbol{b}$ and $\boldsymbol{c}$. Physically, the vector $\boldsymbol{H}\boldsymbol{b}$ represents unknown calibration gains that lie in a known subspace, specified by $\boldsymbol{H}$. Note that (2.42) is

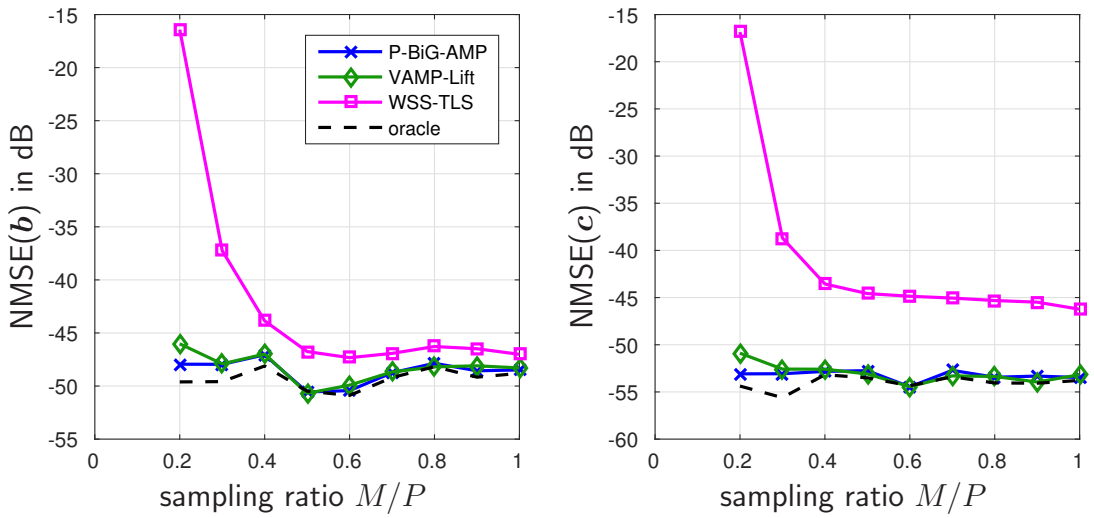

Figure 2.4: NMSE vs. M/P with i.i.d. $\mathcal{N}(0,1)$ $\boldsymbol{A}$.

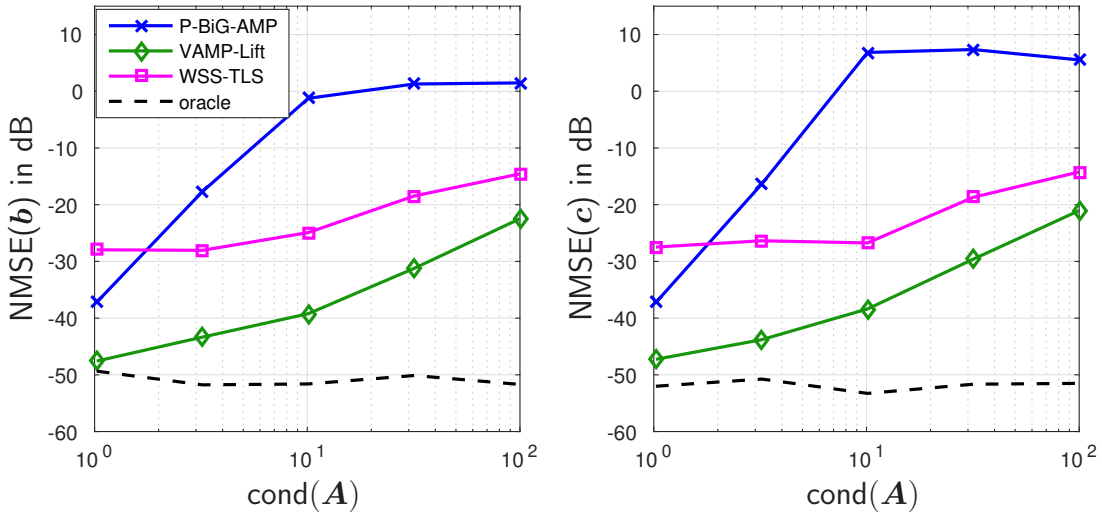

Figure 2.5: NMSE vs. cond($\boldsymbol{A}$) at $M/P = 0.6$.

an instance of (2.39) with $\boldsymbol{A}_l = \text{Diag}(\boldsymbol{h}_l)\boldsymbol{\Psi}$, where $\boldsymbol{h}_l$ denotes the lth column of $\boldsymbol{H}$. Different from "CS with matrix uncertainty", all elements in $\boldsymbol{b}$ are now unknown,

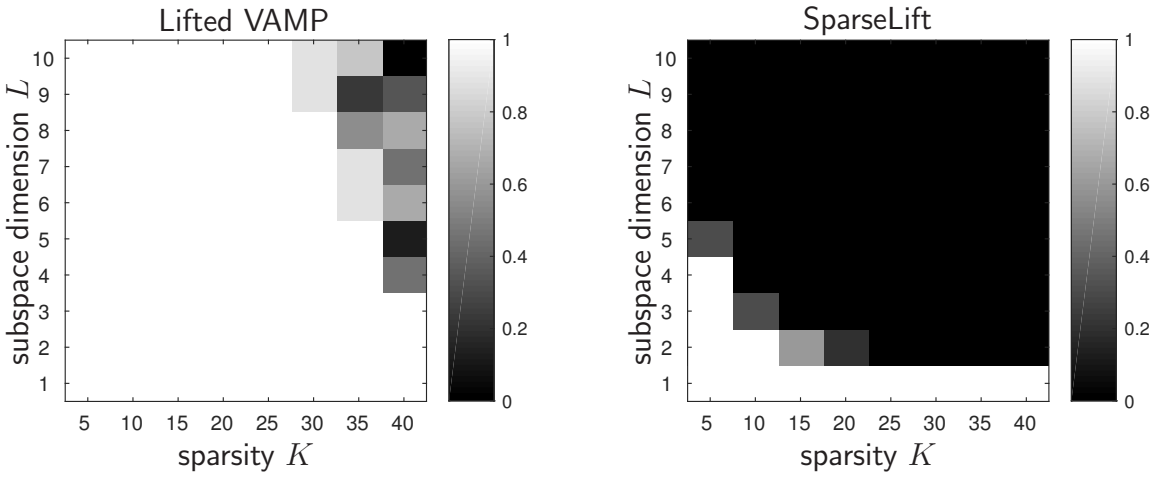

Figure 2.6: Success rate vs. sparsity K and subspace dimension L in self-calibration

and so WSS-TLS [105] cannot be applied. Instead, we compare lifted VAMP to the SparseLift approach from [51], which is based on convex relaxation and has provable guarantees.

For our experiment, we generated $\boldsymbol{\Psi}$ and $\boldsymbol{b} \in \mathbb{R}^L$ as i.i.d. $\mathcal{N}(0,1)$; $\boldsymbol{c}$ as K-sparse with $\mathcal{N}(0,1)$ nonzero entries; $\boldsymbol{H}$ as randomly chosen columns of a Hadamard matrix; and $\boldsymbol{w} = \boldsymbol{0}$. Figure 2.6 plots the success rate versus L and K, where "success" is defined as $\mathbb{E}\,\|\widehat{\boldsymbol{c}}\widehat{\boldsymbol{b}}^{\mathsf{T}} - \boldsymbol{c}\boldsymbol{b}^{\mathsf{T}}\|_F^2 / \mathbb{E}\,\|\boldsymbol{c}\boldsymbol{b}^{\mathsf{T}}\|_F^2 < -60$ dB. The figure shows that, relative to SparseLift, lifted VAMP gives successful recoveries for a wider range of L and K.

Chapter 3: Bilinear Recovery using Adaptive Vector Approximate Message Passing

3.1 Introduction

In this chapter, we are interested in problems that can be formulated as estimation of an unknown structured matrix $\boldsymbol{Z}$ from noisy or incomplete measurements. The type of structure in $\boldsymbol{Z}$ determines the specific subproblem to be solved.

For example, when $\boldsymbol{Z}$ has a low-rank structure and only a subset of its entries are observed, the problem is known as *matrix completion* [21]. When $\boldsymbol{Z} = \boldsymbol{L} + \boldsymbol{S}$ for low-rank $\boldsymbol{L}$ and sparse $\boldsymbol{S}$, the problem of estimating $\boldsymbol{L}$ and $\boldsymbol{S}$ is known as *robust principle components analysis* (RPCA) [20]. When $\boldsymbol{Z} = \boldsymbol{B}\boldsymbol{C}$ with sparse $\boldsymbol{C}$, the problem of estimating $\boldsymbol{B}$ and $\boldsymbol{C}$ is known as *dictionary learning* [81]. When $\boldsymbol{Z} = \boldsymbol{B}\boldsymbol{C}$ with nonnegative $\boldsymbol{B}$ and $\boldsymbol{C}$, the problem is known as *nonnegative matrix factorization* (NMF) [47].

Sometimes $\boldsymbol{Z}$ has a more complicated structure. For example, the problems of *self-calibration* and *blind (circular) deconvolution* [51] can be formulated using $\boldsymbol{Z} = \text{Diag}(\boldsymbol{H}\boldsymbol{b})\boldsymbol{\Psi}\boldsymbol{C}$, where $\boldsymbol{H}$ and $\boldsymbol{\Psi}$ are known and $\boldsymbol{b}$ and $\boldsymbol{C}$ are to be estimated. The problem of *compressive sensing (CS) with matrix uncertainty* [105] can be formulated using $\boldsymbol{z} = \sum_i b_i \boldsymbol{A}_i \boldsymbol{c}$, where $\{\boldsymbol{A}_i\}$ are known and where $\boldsymbol{b}$ and sparse $\boldsymbol{c}$ are to be

estimated. The latter covers the problem of *joint channel-symbol estimation* [43], in which case b_i are the data symbols, $\boldsymbol{c}$ contains (possibly sparse) channel coefficients, and the known $\{\boldsymbol{A}_i\}$ are determined by the modulation scheme. The more general problem of *matrix CS* [22, 100] results from

$$z_m = \mathrm{tr}\{\boldsymbol{A}_m^{\mathsf{T}}(\boldsymbol{L}+\boldsymbol{S})\} \text{ for } m = 1, \ldots, M, \tag{3.1}$$

where $\{\boldsymbol{A}_m\}$ are known and the goal is to estimate low-rank $\boldsymbol{L}$ and sparse $\boldsymbol{S}$.

3.2 Prior Work

Many algorithms have been developed to solve the above problems. Some solve a convex relaxation of the original problem, while others attack non-convex formulations via alternating methods, greedy methods, variational methods, message-passing methods, and other techniques.

For matrix completion, well-known approaches include the nuclear-norm-based convex optimization method IALM [50], the non-convex successive over-relaxation approach LMAFit [101], the Grassmanian gradient-descent approach GROUSE [5], the greedy hard-thresholding approach Matrix-ALPS [46], and the variational-Bayes method VSBL [4]. For RPCA, there are also versions of IALM [50], LMaFit [101], and VSBL [4], as well as a robust cousin of GROUSE, called GRASTA [39]. For dictionary learning, there is the greedy K-SVD algorithm [2], the online SPAMS approach [54], and the ER-SpUD approach from [87]. A unified approach to matrix completion, RPCA, and dictionary learning was proposed in [42, 67, 68] using an extension of the approximate message-passing (AMP) methodology from [32, 71]. The resulting "bilinear generalized AMP" (BiGAMP) algorithm was compared to the aforementioned methods in [68] and found (empirically) to be competitive, if not

superior, in phase transition and runtime. A related approach known as LowRAMP was proposed [55] and analyzed in [49, 59].

For self-calibration and blind deconvolution, well-known approaches include the convex relaxations from [3, 13, 51] and the alternating method from [40]. For CS with matrix uncertainty, there is the award-winning non-convex method [105]. For matrix CS, the well-known papers [1, 22, 102] proposed convex approaches and [46, 100] proposed greedy approaches. See the recent overview [28] for many other works. An AMP-based approach to self-calibration, blind deconvolution, CS with matrix uncertainty, and matrix CS was proposed in [66] and analyzed in [85]. This "parametric BiGAMP" (PBiGAMP) was compared to the above works in [66] and found to yield improved empirical phase transitions.

More recently, AMP methods for bilinear inference were proposed using the "lifting" approach (see, e.g., [3,23,28,51] for seminal papers on lifting). We used the lifting method in chapter 2 to solve some bilinear problems by framing it as a linear inverse problem. We illustrate the main idea behind lifting here. Suppose the measurement vector $\boldsymbol{y} \in \mathbb{R}^M$ takes the form

$$\boldsymbol{y} = \sum_{i=1}^{Q} \sum_{j=1}^{N} b_i \boldsymbol{a}_{i,j} c_j + \boldsymbol{w}, \tag{3.2}$$

where $\boldsymbol{a}_{i,j} \in \mathbb{R}^M$ is known for all i, j and the goal is to recover $\boldsymbol{b} = [b_1, \dots, b_Q]^\mathsf{T}$ and $\boldsymbol{c} = [c_1, \dots, c_N]^\mathsf{T}$ in the presence of white noise $\boldsymbol{w}$. Rewriting the measurements as

$$\boldsymbol{y} = \sum_{i=1}^{Q} b_i \underbrace{\left[\boldsymbol{a}_{i,1}, \cdots, \boldsymbol{a}_{i,N}\right]}_{\triangleq \boldsymbol{A}_i} \boldsymbol{c} + \boldsymbol{w} \tag{3.3}$$

$$= \underbrace{\left[\boldsymbol{A}_1 \cdots \boldsymbol{A}_Q\right]}_{\triangleq \boldsymbol{A}} \begin{bmatrix} b_1\boldsymbol{c} \\ \vdots \\ b_Q\boldsymbol{c} \end{bmatrix} + \boldsymbol{w} \tag{3.4}$$

$$= \boldsymbol{A}\boldsymbol{x} + \boldsymbol{w} \quad \text{for} \quad \boldsymbol{x} = \boldsymbol{b} \otimes \boldsymbol{c} = \mathrm{vec}(\boldsymbol{c}\boldsymbol{b}^\mathsf{T}), \tag{3.5}$$

we see that the noisy bilinear recovery problem (3.2) can be rewritten as the noisy linear recovery problem (3.5) with a rank-one structure on (the matrix form of) $\boldsymbol{x}$. Thus, if this low-rank signal structure can be exploited by a linear inference algorithm, then bilinear inference can be accomplished. This is precisely what was proposed in [80], building on the non-separable-denoising version of the AMP algorithm from [56]. A rigorous analysis of "lifted AMP" was presented in [12].

The trouble with AMP is that its behavior is understood only in the case of large [83] or infinitely large, i.i.d. (sub) Gaussian $\boldsymbol{A}$ [8,9]. Even small deviations from this scenario (e.g., mildly ill-conditioned and/or non-zero-mean $\boldsymbol{A}$) can cause AMP to diverge [18, 74, 96]. The Vector AMP (VAMP) algorithm discussed in chapter 2 addresses this issue. In [75], it was first established that, if $\boldsymbol{A}$ is an infinitely large right-rotationally invariant[2] random matrix and the denoising function used by VAMP is separable and Lipschitz, then VAMP's performance can be exactly predicted by a scalar state-evolution that also provides testable conditions for optimality. Since the class of right-rotationally invariant matrices is much larger than the class of

[2]If $\boldsymbol{A}$ is right-rotationally invariant then its singular value decomposition $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^\mathsf{T}$ has Haar distributed $\boldsymbol{V}$, i.e., $\boldsymbol{V}$ is uniformly distributed over the group of orthogonal matrices.

i.i.d. Gaussian matrices, VAMP is much more robust than AMP with regards to the construction of $\boldsymbol{A}$. For example, VAMP has no problem with ill-conditioned or mean-shifted matrices [75].

In chapter 2 Sec. 2.4, we performed a rigorous analysis of VAMP under *non*-separable Lipschitz denoisers, showing that—here too—VAMP's behavior is exactly predicted by a scalar state-evolution when $\boldsymbol{A}$ is infinitely large and right-rotationally invariant. Moreover, we also demonstrated the success of lifted VAMP on bilinear problems such as self-calibration and CS with matrix uncertainty. In addition, we gave evidence that, like AMP, the PBiGAMP algorithm is sensitive to deviations from the i.i.d. assumptions used in its derivation [66] and analysis [85]. For this reason, lifted VAMP significantly outperformed PBiGAMP in some cases.

In our numerical experiments (Sec. 2.5), we showed that lifted VAMP has good performance and a rigorous analyses under infinitely large right-rotationally invariant random $\boldsymbol{A}$. However, it suffers from computational issues brought on by the lifting itself: The $N+Q$ unknowns $[\boldsymbol{b},\boldsymbol{c}]$ in the bilinear problem (3.2) manifest as NQ unknowns $\boldsymbol{x}$ after lifting to (3.5). This is a serious problem when N and Q are both large. As a concrete example, consider the application of lifting to (square) dictionary learning, where the goal is to recover $\boldsymbol{B} \in \mathbb{R}^{N\times N}$ and sparse $\boldsymbol{C} \in \mathbb{R}^{N\times L}$ from noisy measurements $\boldsymbol{Y} = \boldsymbol{B}\boldsymbol{C} + \boldsymbol{W}$. This bilinear relationship can be lifted via

$$\boldsymbol{Y} = \sum_{ij} b_{i,j}\boldsymbol{A}_{i,j}\boldsymbol{C} + \boldsymbol{W} \tag{3.6}$$

$$= \underbrace{[\boldsymbol{A}_{1,1}\cdots\boldsymbol{A}_{N,N}]}_{\triangleq\, \boldsymbol{A}\in\mathbb{R}^{N\times N^3}} \underbrace{\big(\boldsymbol{b}\otimes\boldsymbol{C}\big)}_{\triangleq\, \boldsymbol{X}\in\mathbb{R}^{N^3\times L}} + \boldsymbol{W}, \tag{3.7}$$

where $\boldsymbol{A}_{i,j} \in \mathbb{R}^{N\times N}$ is constructed with a 1 in the (i,j)th position and zeros elsewhere, and where $\boldsymbol{b} = [b_{1,1},\ldots,b_{N,N}]^{\mathsf{T}} \in \mathbb{R}^{N^2}$. Even at the relatively small patch size of 8×8

(i.e., $N = 64$), the matrix $\boldsymbol{A}$ has dimension $64 \times 262\,144$, and the unknown matrix $\boldsymbol{X}$ has dimension $262\,144 \times L$. The rule-of-thumb $L = 5N \ln N$ [87] then gives $L = 1331$, in which case $\boldsymbol{X}$ contains 3.5×10^8 entries, which leads to difficulties with computation and memory.

3.3 Contributions

In this chapter, we present a novel VAMP-based approach to bilinear recovery. With the aim of computational efficiency, we avoid lifting and instead build on the recently proposed Adaptive VAMP framework from [37]. However, different from [37], which focused on noisy *linear* recovery, we focus on noisy *bilinear* recovery.

In particular, we focus on recovering the unknown parameters $\{b_i\}_{i=1}^Q$ and the unknown random matrix $\boldsymbol{C} \in \mathbb{R}^{N \times L}$ from noisy measurements $\boldsymbol{Y} \in \mathbb{R}^{M \times L}$ of the form

$$\boldsymbol{Y} = \sum_{i=1}^{Q} b_i \boldsymbol{A}_i \boldsymbol{C} + \boldsymbol{W}, \tag{3.8}$$

where $\{\boldsymbol{A}_i\}$ are known and $\boldsymbol{W}$ is additive white Gaussian noise (AWGN). Note that (3.8) is a multiple-measurement vector (MMV) extension of (3.3), and that it covers all of the motivating problems discussed in Sec. 3.1. For example, in self-calibration, where we estimate $\boldsymbol{b}$ and $\boldsymbol{C}$ from $\boldsymbol{Y} = \mathrm{Diag}(\boldsymbol{H}\boldsymbol{b})\boldsymbol{\Psi}\boldsymbol{C} + \boldsymbol{W}$, we can set $\boldsymbol{A}_i = \mathrm{Diag}(\boldsymbol{h}_i)\boldsymbol{\Psi}$, where $\boldsymbol{h}_i$ is the ith column of $\boldsymbol{H}$. Or, in dictionary learning, where we estimate $\boldsymbol{B}$ and $\boldsymbol{C}$ from $\boldsymbol{Y} = \boldsymbol{B}\boldsymbol{C} + \boldsymbol{W}$, we can write $\boldsymbol{B} = \sum_{i=1}^{MN} b_i \boldsymbol{A}_i$ for $\boldsymbol{A}_i = \boldsymbol{e}_{\langle i-1 \rangle_M} \boldsymbol{e}_{\lfloor (i-1)/M \rfloor}^{\mathsf{T}}$, where $\langle i \rangle_M$ denotes i-modulo-M, $\lfloor \cdot \rfloor$ denotes floor, and $\{\boldsymbol{e}_i\}$ is the standard basis.

When deriving[3] the proposed method, we treat $\{b_i\}$ as deterministic unknowns and the entries of $\boldsymbol{C}$ as random variables. The prior distribution on $\boldsymbol{C}$ is assumed to be known up to some (possibly) unknown hyperparameters, which are learned jointly with $\{b_i\}$ and $\boldsymbol{C}$. Also, $\boldsymbol{W}$ is treated as additive white Gaussian noise (AWGN) with an unknown variance that is also learned. More details are provided in the sequel.

We show (empirically) that the proposed Bilinear Adaptive VAMP (BAd-VAMP) method performs as well as the EM-PBiGAMP algorithm from [66], with regard to accuracy and computational complexity, when the underlying matrices are i.i.d., as assumed for the derivation of PBiGAMP. However, we will show that BAd-VAMP outperforms EM-PBiGAMP when the underlying matrices become ill-conditioned. In the ill-conditioned case, we show that BAd-VAMP performs as well as, and sometimes significantly better than, lifted VAMP. However, BAd-VAMP is much more computationally efficient due to its avoidance of lifting. In this sense, the proposed BAd-VAMP is shown to be accurate, robust, and computationally efficient.

3.4 Problem Formulation

In an effort to make our algorithmic development more consistent with our discussion on VAMP in chapter 2, we now make the following notational changes relative to (3.8):

(a) We will use the notation $\boldsymbol{A}(\boldsymbol{b}) \triangleq \sum_i b_i \boldsymbol{A}_i$ to be concise.

(b) The quantities b_i and $\boldsymbol{C}$ in (3.8) will be changed to $\theta_{A,i}$ and $\boldsymbol{X}$ respectively.

[3]Although the derivation treats the entries of $\boldsymbol{C}$ as random variables and the associated denoiser as Bayesian, the final algorithm is more general in that it only requires the denoiser to be Lipschitz.

We can thus state our problem of interest as follows: estimate the matrix $\boldsymbol{X} \in \mathbb{R}^{N\times L}$ and learn the parameters $\boldsymbol{\Theta} \triangleq \{\boldsymbol{\theta}_A, \boldsymbol{\theta}_x, \gamma_w\}$ in the statistical model

$$\boldsymbol{Y} = \boldsymbol{A}(\boldsymbol{\theta}_A)\boldsymbol{X} + \boldsymbol{W} \tag{3.9a}$$

$$\boldsymbol{x}_l \sim p_{\mathbf{x}}(\cdot;\boldsymbol{\theta}_x) \quad \forall l \in \{1,\dots,L\} \tag{3.9b}$$

$$w_{ml} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0,\gamma_w^{-1}), \tag{3.9c}$$

where $\boldsymbol{A}(\cdot)$ is a known matrix-valued linear function, and $p_{\mathbf{x}}(\cdot;\boldsymbol{\theta}_x)$ is a prior density on $\boldsymbol{X}$ parameterized by the unknown vector $\boldsymbol{\theta}_x$. Here, γ_w is the noise precision, i.e., the inverse noise variance.

More precisely, we aim to compute the maximimum-likelihood (ML) estimate of $\boldsymbol{\Theta}$ and, under that estimate, compute the minimum mean-squared error (MMSE) estimate of $\boldsymbol{X}$, i.e.,

$$\widehat{\boldsymbol{\Theta}}_{\mathsf{ML}} = \arg\max_{\boldsymbol{\Theta}} \; p_{\mathbf{Y}}(\boldsymbol{Y};\boldsymbol{\Theta}) \tag{3.10}$$

$$\widehat{\boldsymbol{X}}_{\mathsf{MMSE}} = \mathbb{E}[\boldsymbol{X}|\boldsymbol{Y};\widehat{\boldsymbol{\Theta}}_{\mathsf{ML}}]. \tag{3.11}$$

In (3.10), $p_{\mathbf{Y}}(\boldsymbol{Y};\boldsymbol{\Theta})$ is the likelihood function of $\boldsymbol{\Theta}$, which can be written as

$$p_{\mathbf{Y}}(\boldsymbol{Y};\boldsymbol{\Theta}) = \int p_{\mathbf{X}}(\boldsymbol{X};\boldsymbol{\Theta})\; p_{\mathbf{Y}|\mathbf{X}}(\boldsymbol{Y}|\boldsymbol{X};\boldsymbol{\Theta})\; \mathrm{d}\boldsymbol{X}. \tag{3.12}$$

In (3.11), the expectation is taken over the posterior density

$$p_{\mathbf{X}|\mathbf{Y}}(\boldsymbol{X}|\boldsymbol{Y};\widehat{\boldsymbol{\Theta}}_{\mathsf{ML}}) = \frac{p_{\mathbf{X}}(\boldsymbol{X};\widehat{\boldsymbol{\Theta}}_{\mathsf{ML}})p_{\mathbf{Y}|\mathbf{X}}(\boldsymbol{Y}|\boldsymbol{X};\widehat{\boldsymbol{\Theta}}_{\mathsf{ML}})}{p_{\mathbf{Y}}(\boldsymbol{Y};\widehat{\boldsymbol{\Theta}}_{\mathsf{ML}})}. \tag{3.13}$$

The statistical model (3.9) implies that[4]

$$p_{\mathbf{Y}|\mathbf{X}}(\boldsymbol{Y}|\boldsymbol{X};\boldsymbol{\Theta}) = \prod_{l=1}^{L} p_{\mathbf{y}|\mathbf{x}}(\boldsymbol{y}_l|\boldsymbol{x}_l;\boldsymbol{\Theta}) \tag{3.14}$$

[4]In (3.14)-(3.16), to promote notational simplicity, the left side of the equation is written using $\boldsymbol{\Theta}$ even though the right side depends on a subset of $\boldsymbol{\Theta}$.

where each factor becomes

$$p_{\mathbf{y}|\mathbf{x}}(\boldsymbol{y}|\boldsymbol{x};\boldsymbol{\Theta}) = \mathcal{N}\big(\boldsymbol{y};\boldsymbol{A}(\boldsymbol{\theta}_A)\boldsymbol{x},\boldsymbol{I}/\gamma_w\big). \tag{3.15}$$

Since the columns $\{\boldsymbol{x}_l\}_{l=1}^L$ are independently drawn from density $p_{\mathbf{x}}(\cdot;\boldsymbol{\theta}_x)$, we have

$$p_{\mathbf{X}}(\boldsymbol{X};\boldsymbol{\Theta}) = \prod_{l=1}^{L} p_{\mathbf{x}}(\boldsymbol{x}_l;\boldsymbol{\theta}_x) \tag{3.16}$$

for some density $p_{\mathbf{x}}(\cdot;\boldsymbol{\theta}_x)$ parameterized by $\boldsymbol{\theta}_x$. In this case, the posterior density $p_{\mathbf{X}|\mathbf{Y}}$ decouples across the columns of $\boldsymbol{X}$ and $\boldsymbol{Y}$ as

$$p_{\mathbf{X}|\mathbf{Y}}(\boldsymbol{X}|\boldsymbol{Y};\boldsymbol{\Theta}) \propto \prod_{l=1}^{L} p_{\mathbf{x}}(\boldsymbol{x}_l;\boldsymbol{\Theta}) p_{\mathbf{y}|\mathbf{x}}(\boldsymbol{y}_l|\boldsymbol{x}_l;\boldsymbol{\Theta}). \tag{3.17}$$

3.5 Expectation Maximization

We can use maximum likelihood (ML) estimation to learn the unknown parameters $\boldsymbol{\Theta}$ in (3.9). From (3.10) and (3.12), we have

$$\widehat{\boldsymbol{\Theta}}_{\mathsf{ML}} = \arg\min_{\boldsymbol{\Theta}} -\ln \int p_{\mathbf{X}}(\boldsymbol{X};\boldsymbol{\Theta}) p_{\mathbf{Y}|\mathbf{X}}(\boldsymbol{Y}|\boldsymbol{X};\boldsymbol{\Theta})\,\mathrm{d}\boldsymbol{X}, \tag{3.18}$$

but (3.18) is impractical to optimize directly due to the high dimensional integral.

Expectation-maximization (EM) [29] is a well known iterative approach to ML that alternates between the following two steps:

(a) Minimizing an upper-bound of the negative log-likelihood.

(b) Tightening the upper-bound.

The EM algorithm is usually written as

$$Q(\boldsymbol{\Theta};\widehat{\boldsymbol{\Theta}}^t) \triangleq -\,\mathbb{E}\left[\ln p_{\mathbf{X},\mathbf{Y}}(\boldsymbol{X},\boldsymbol{Y};\boldsymbol{\Theta}) \,\middle|\, \boldsymbol{Y};\widehat{\boldsymbol{\Theta}}^t\right] \tag{3.19a}$$

$$\widehat{\boldsymbol{\Theta}}^{t+1} = \arg\min_{\boldsymbol{\Theta}} Q(\boldsymbol{\Theta};\widehat{\boldsymbol{\Theta}}^t). \tag{3.19b}$$

Letting $q^t = p_{\mathbf{X}|\mathbf{Y}}(\cdot|\boldsymbol{Y};\widehat{\boldsymbol{\Theta}}^t)$, we can write

$$\begin{aligned} Q(\boldsymbol{\Theta};\widehat{\boldsymbol{\Theta}}^t) &= -\,\mathbb{E}\left[\ln p_{\mathbf{X}}(\boldsymbol{X};\boldsymbol{\Theta})\,\middle|\,\boldsymbol{Y};\widehat{\boldsymbol{\Theta}}^t\right] \\ &\quad -\,\mathbb{E}\left[\ln p_{\mathbf{Y}|\mathbf{X}}(\boldsymbol{Y}|\boldsymbol{X};\boldsymbol{\Theta})\,\middle|\,\boldsymbol{Y};\widehat{\boldsymbol{\Theta}}^t\right] && \text{(3.20a)} \\ &= -\,\mathbb{E}\left[\ln p_{\mathbf{X}}(\boldsymbol{X};\boldsymbol{\Theta})\,\middle|\,q^t\right] \\ &\quad -\,\mathbb{E}\left[\ln p_{\mathbf{Y}|\mathbf{X}}(\boldsymbol{Y}|\boldsymbol{X};\boldsymbol{\Theta})\,\middle|\,q^t\right] && \text{(3.20b)} \\ &= J(q^t,q^t,q^t;\boldsymbol{\Theta}) + \text{const.} && \text{(3.20c)} \end{aligned}$$

where J is the Gibbs free energy from (2.13) where there were no unknown parameters. In the presence of unknown parameters $\boldsymbol{\Theta}$, it takes the form

$$J(q_1,q_2,q_3;\boldsymbol{\Theta}) \triangleq D_{\mathsf{KL}}(q_1\|p_{\mathbf{X}}(\cdot,\boldsymbol{\Theta})) + D_{\mathsf{KL}}(q_2\|p_{\mathbf{Y}|\mathbf{X}}(\boldsymbol{Y}|\cdot;\boldsymbol{\Theta})) + H(q_3). \tag{3.21}$$

Thus, (3.19) can also be written as the following two steps [61]:

$$q^t = p_{\mathbf{X}|\mathbf{Y}}(\cdot|\boldsymbol{Y};\widehat{\boldsymbol{\Theta}}^t) \tag{3.22a}$$

$$\widehat{\boldsymbol{\Theta}}^{t+1} = \arg\min_{\boldsymbol{\Theta}} J(q^t,q^t,q^t;\boldsymbol{\Theta}). \tag{3.22b}$$

EM is an type of Majorization-Minimization algorithm as it minimizes an upper bound to the actual cost function that we want to minimize. We can show that $J(q^t,q^t,q^t;\boldsymbol{\Theta})$ is an upper bound on $-\ln p_{\mathbf{Y}}(\boldsymbol{Y};\boldsymbol{\Theta})$ for *any* q^t since

$$\begin{aligned} J(q^t,q^t,q^t;\boldsymbol{\Theta}) &= -\ln p_{\mathbf{Y}}(\boldsymbol{Y};\boldsymbol{\Theta}) + D_{\mathsf{KL}}(q^t\,\|\,p_{\mathbf{Y}|\mathbf{X}}(\boldsymbol{Y}|\cdot;\boldsymbol{\Theta})) && \text{(3.23)} \\ &\overset{\text{(a)}}{\geq} -\ln p_{\mathbf{Y}}(\boldsymbol{Y};\boldsymbol{\Theta}), && \text{(3.24)} \end{aligned}$$

where (a) follows from $D_{\mathsf{KL}} \geq 0$ by construction. Thus, while the specific choice of q^t in (3.22a) yields a tight upper bound in that

$$J(q^t,q^t,q^t;\widehat{\boldsymbol{\Theta}}^t) = -\ln p_{\mathbf{Y}}(\boldsymbol{Y};\widehat{\boldsymbol{\Theta}}^t), \tag{3.25}$$

Algorithm 3 Bilinear EM-VAMP

initialize:
  $\forall l : \boldsymbol{r}_{1,l}^0, \gamma_{1,l}^0, \boldsymbol{\theta}_x^0, \boldsymbol{\theta}_A^0, \gamma_w^0$
for $t = 0, \ldots, T_{\max}$ **do**
  $\forall l : \boldsymbol{x}_{1,l}^t = \boldsymbol{g}_1(\boldsymbol{r}_{1,l}^t, \gamma_{1,l}^t; \boldsymbol{\theta}_x^t)$
  $\forall l : 1/\eta_{1,l}^t = \langle \boldsymbol{g}_1'(\boldsymbol{r}_{1,l}^t, \gamma_{1,l}^t; \boldsymbol{\theta}_x^t) \rangle / \gamma_{1,l}^t$
  $q_1^t(\boldsymbol{X}) \propto \prod_{l=1}^L p_{\mathbf{x}}(\boldsymbol{x}_l; \boldsymbol{\theta}_x^t) e^{-\frac{1}{2}\gamma_{1,l}^t \|\boldsymbol{x}_l - \boldsymbol{r}_{1,l}^t\|^2}$
  $\boldsymbol{\theta}_x^{t+1} = \arg\max_{\boldsymbol{\theta}_x} \mathbb{E}[\ln p_{\mathbf{X}}(\boldsymbol{X}; \boldsymbol{\theta}_x) | q_1^t]$
  $\forall l : \gamma_{2,l}^t = \eta_{1,l}^t - \gamma_{1,l}^t$
  $\forall l : \boldsymbol{r}_{2,l}^t = (\eta_{1,l}^t \boldsymbol{x}_{1,l}^t - \gamma_{1,l}^t \boldsymbol{r}_{1,l}^t)/\gamma_{2,l}^t$
  $\forall l : \boldsymbol{x}_{2,l}^t = \boldsymbol{g}_{2,l}(\boldsymbol{r}_{2,l}^t, \gamma_{2,l}^t; \boldsymbol{\theta}_A^t, \gamma_w^t)$
  $\forall l : 1/\eta_{2,l}^t = \langle \boldsymbol{g}_{2,l}'(\boldsymbol{r}_{2,l}^t, \gamma_{2,l}^t; \boldsymbol{\theta}_A^t, \gamma_w^t) \rangle / \gamma_{2,l}^t$
  $q_2^t(\boldsymbol{X}) \propto \prod_{l=1}^L p_{\mathbf{y}|\mathbf{x}}(\boldsymbol{y}_l | \boldsymbol{x}_l; \boldsymbol{\theta}_A^t, \gamma_w^t) e^{-\frac{1}{2}\gamma_{2,l}^t \|\boldsymbol{x}_l - \boldsymbol{r}_{2,l}^t\|^2}$
  $\boldsymbol{\theta}_A^{t+1} = \arg\max_{\boldsymbol{\theta}_A} \mathbb{E}[\ln p_{\mathbf{Y}|\mathbf{X}}(\boldsymbol{Y}|\boldsymbol{X}; \boldsymbol{\theta}_A, \gamma_w^t) | \boldsymbol{Y}, q_2^t]$
  $\gamma_w^{t+1} = \arg\max_{\gamma_w} \mathbb{E}[\ln p_{\mathbf{Y}|\mathbf{X}}(\boldsymbol{Y}|\boldsymbol{X}; \boldsymbol{\theta}_A^{t+1}, \gamma_w) | \boldsymbol{Y}, q_2^t]$
  $\forall l : \gamma_{1,l}^{t+1} = \eta_{2,l}^t - \gamma_{2,l}^t$
  $\forall l : \boldsymbol{r}_{1,l}^{t+1} = (\eta_{2,l}^t \boldsymbol{x}_{2,l}^t - \gamma_{2,l}^t \boldsymbol{r}_{2,l}^t)/\gamma_{1,l}^{t+1}$
end for

other choices of bounding q^t can also be used in EM [61]. EM is used in many parameter estimation problems in signal processing most notable being training hidden Markov Models in speech [69]. In the next section, we will use the variational interpretation of EM to develop an algorithm to solve the bilinear recovery problem in (3.9).

3.6 Bilinear EM-VAMP

From the descriptions of VAMP in chapter 2 and EM algorithm in Sec. 3.5, we see that both the algorithms minimize the same Gibbs free energy cost $J(q_1, q_2, q_3; \boldsymbol{\Theta})$ from (3.21), but w.r.t. different variables. VAMP minimizes J w.r.t. the beliefs $\{q_1, q_2, q_3\}$ under the first and second moment-matching constraints for a given value of $\boldsymbol{\Theta}$. Meanwhile, EM minimizes J w.r.t. the parameters $\boldsymbol{\Theta}$ for a given beliefs

$\{q_1, q_2, q_3\}$. As a result, the two approaches can be straightforwardly merged for *joint* estimation of $\{q_1, q_2, q_3\}$ and $\boldsymbol{\Theta}$. In doing so, the goal is to solve the following constrained optimization problem

$$\arg \min_{\boldsymbol{\Theta}, q_1} \min_{q_2} \max_{q_3} J(q_1, q_2, q_3; \boldsymbol{\Theta}) \tag{3.26a}$$

$$\text{s.t.} \quad \mathbb{E}[\boldsymbol{x}|q_1] = \mathbb{E}[\boldsymbol{x}|q_2] = \mathbb{E}[\boldsymbol{x}|q_3] \tag{3.26b}$$

$$\text{tr}\{\text{Cov}[\boldsymbol{x}|q_1]\} = \text{tr}\{\text{Cov}[\boldsymbol{x}|q_2]\} = \text{tr}\{\text{Cov}[\boldsymbol{x}|q_3]\}, \tag{3.26c}$$

and the proposed methodology is to "interleave" the VAMP and EM algorithms, as specified in Alg. 3. In lines 3-4 of Alg. 3, the estimation function $\boldsymbol{g}_1$ is similar to that in VAMP (2.17) but with parameters $\boldsymbol{\theta}_x$,

$$\boldsymbol{g}_1(\boldsymbol{r}_{1,l}, \gamma_{1,l}; \boldsymbol{\theta}_x) \triangleq \frac{\int \boldsymbol{x}\, p_{\mathbf{x}}(\boldsymbol{x}; \boldsymbol{\theta}_x) \mathcal{N}(\boldsymbol{x}; \boldsymbol{r}_{1,l}, \boldsymbol{I}/\gamma_{1,l})\, \mathrm{d}\boldsymbol{x}}{\int p_{\mathbf{x}}(\boldsymbol{x}; \boldsymbol{\theta}_x) \mathcal{N}(\boldsymbol{x}; \boldsymbol{r}_{1,l}, \boldsymbol{I}/\gamma_{1,l})\, \mathrm{d}\boldsymbol{x}}. \tag{3.27}$$

And in lines 9-10, the estimation function $\boldsymbol{g}_{2,l}$ is the MMV version of the linear estimator $\boldsymbol{g}_2$ from (2.18) and it is defined as

$$\boldsymbol{g}_{2,l}(\boldsymbol{r}_{2,l}, \gamma_{2,l}; \boldsymbol{\theta}_A, \gamma_w) \triangleq \frac{\int \boldsymbol{x}\, p_{\mathbf{y}|\mathbf{x}}(\boldsymbol{y}_l|\boldsymbol{x}; \boldsymbol{\theta}_A, \gamma_w) \mathcal{N}(\boldsymbol{x}; \boldsymbol{r}_{2,l}, \boldsymbol{I}/\gamma_{2,l})\, \mathrm{d}\boldsymbol{x}}{\int p_{\mathbf{y}|\mathbf{x}}(\boldsymbol{y}_l|\boldsymbol{x}; \boldsymbol{\theta}_A, \gamma_w) \mathcal{N}(\boldsymbol{x}; \boldsymbol{r}_{2,l}, \boldsymbol{I}/\gamma_{2,l})\, \mathrm{d}\boldsymbol{x}}. \tag{3.28}$$

It is the LMMSE estimator applied on each column $\boldsymbol{x}_l$ in the model

$$\boldsymbol{y}_l = \boldsymbol{A}(\boldsymbol{\theta}_A)\boldsymbol{x}_l + \boldsymbol{w}_l \tag{3.29a}$$

$$\boldsymbol{x}_l \sim \mathcal{N}(\boldsymbol{r}_{2,l}, \boldsymbol{I}/\gamma_{2,l}) \tag{3.29b}$$

$$\boldsymbol{w}_l \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma_w). \tag{3.29c}$$

The other lines in Alg. 3 will be detailed in sections from 3.7.1 to 3.7.3.

3.7 Bilinear Adaptive VAMP

Assume at every iteration t, the following conditions are satisfied in BAdVAMP

i) The matrix $\boldsymbol{A}(\boldsymbol{\theta}_A^t)$ is infinitely large and right-rotationally invariant.

ii) The estimation functions $\boldsymbol{g}_1$ and $\{\boldsymbol{g}_{2,l}\}_{l=1}^L$ are "matched" (i.e., MMSE) for the statistical model generating $(\boldsymbol{X}, \boldsymbol{Y})$.

Then, the VAMP state-evolution from (2.34) shows that at every iteration t, the VAMP quantities $\{(\boldsymbol{r}_{1,l}^t, \gamma_{1,l}^t, \boldsymbol{r}_{2,l}^t, \gamma_{2,l}^t)\}_{l=1}^L$ obey the statistical model

$$\boldsymbol{r}_{1,l}^t = \boldsymbol{x}_l + \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma_{1,l}^t) \; \forall l \tag{3.30a}$$

$$\boldsymbol{x}_l = \boldsymbol{r}_{2,l}^t + \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma_{2,l}^t) \; \forall l, \tag{3.30b}$$

where $\boldsymbol{x}_l$ is the lth column of the true signal realization $\boldsymbol{X}$ that we aim to recover. That is, $\boldsymbol{r}_{1,l}^t$ is an AWGN-corrupted version of the true signal $\boldsymbol{x}_l$ with *known* AWGN precision $\gamma_{1,l}^t$, and the true signal $\boldsymbol{x}_l$ is an AWGN-corrupted version of $\boldsymbol{r}_{2,l}^t$ with *known* AWGN precision $\gamma_{2,l}^t$. In the context of EM-VAMP under (3.9), this "matched" condition requires that $\boldsymbol{\theta}_A^t$, $\boldsymbol{\theta}_x^t$, and γ_w^t are all perfect estimates. When $\boldsymbol{\theta}_A^t$, $\boldsymbol{\theta}_x^t$, or γ_w^t are *not* perfect, so that $\boldsymbol{g}_1$ and $\boldsymbol{g}_{2,l}$ are mismatched, the VAMP state-evolution shows that $\boldsymbol{r}_{1,l}^t$ is still an AWGN corrupted version of $\boldsymbol{x}_l$, but with an AWGN precision *different* than $\gamma_{1,l}^t$.

In the mismatched case, the impact on EM-VAMP is the following. While the algorithm is trying to learn $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_A, \boldsymbol{\theta}_x, \gamma_w\}$, the value of $\gamma_{i,l}^t$ does *not* correctly characterize the noise precision in $\boldsymbol{r}_{i,l}^t$. As a result, the beliefs q_1^t and q_2^t in lines 5 and 11 of Alg. 3 become mismatched, which compromises the EM updates of $\boldsymbol{\Theta}^t$.

To remedy this situation, it was proposed in [45] (in the context of EM-GAMP [97]) to explicitly estimate the precision of the AWGN corruption on $\boldsymbol{r}_{1,l}^t$ and $\boldsymbol{r}_{2,l}^t$ and use it in place of the AMP-supplied estimates $\gamma_{1,l}^t$ and $\gamma_{2,l}^t$. This approach was coined "Adaptive" GAMP in [45] and later extended to (linear) Adaptive VAMP in [37].

For Bilinear Adaptive VAMP, the first goal is to replace the estimation of $\boldsymbol{\theta}_x^t$ in line 6 of Alg. 3 with the joint ML estimation

$$(\boldsymbol{\theta}_x^t, \boldsymbol{\gamma}_1^t) = \arg\max_{\boldsymbol{\theta}_x, \boldsymbol{\gamma}_1} p(\boldsymbol{R}_1^t; \boldsymbol{\gamma}_1, \boldsymbol{\theta}_x) \tag{3.31}$$

under the following statistical model

$$\boldsymbol{r}_{1,l}^t = \boldsymbol{x}_l + \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma_{1,l}) \; \forall l \tag{3.32}$$

$$\boldsymbol{x}_l \sim p_{\mathbf{x}}(\cdot; \boldsymbol{\theta}_x) \; \forall l, \tag{3.33}$$

with independence across $l = 1, \ldots, L$. For this subproblem, we propose to use (inner) EM iterations indexed by τ, i.e.,

$$(\boldsymbol{\theta}_x^{\tau+1}, \boldsymbol{\gamma}_1^{\tau+1}) = \arg\max_{\boldsymbol{\theta}_x, \boldsymbol{\gamma}_1} \mathbb{E}\left[\ln p(\boldsymbol{X}, \boldsymbol{R}_1^t; \boldsymbol{\gamma}_1, \boldsymbol{\theta}_x) \,\middle|\, \boldsymbol{R}_1^t; \boldsymbol{\gamma}_1^\tau, \boldsymbol{\theta}_x^\tau\right] \tag{3.34}$$

$$= \arg\max_{\boldsymbol{\theta}_x, \boldsymbol{\gamma}_1} \Big\{ \mathbb{E}\left[\ln p(\boldsymbol{X}; \boldsymbol{\theta}_x) \,\middle|\, \boldsymbol{R}_1^t; \boldsymbol{\gamma}_1^\tau, \boldsymbol{\theta}_x^\tau\right] + \mathbb{E}\left[\ln p(\boldsymbol{R}_1^t | \boldsymbol{X}; \boldsymbol{\gamma}_1) \,\middle|\, \boldsymbol{R}_1^t; \boldsymbol{\gamma}_1^\tau, \boldsymbol{\theta}_x^\tau\right] \Big\}. \tag{3.35}$$

From (3.32), we can write the following

$$p(\boldsymbol{R}_1^t | \boldsymbol{X}; \boldsymbol{\gamma}_1) = \prod_{l=1}^L p(\boldsymbol{r}_{1,l}^t | \boldsymbol{x}_l; \gamma_{1,l}) \tag{3.36}$$

$$= \prod_{l=1}^L \mathcal{N}(\boldsymbol{r}_{1,l}^t; \boldsymbol{x}_l, \boldsymbol{I}/\gamma_{1,l}). \tag{3.37}$$

The optimization problem in (3.34) decouples into

$$\boldsymbol{\theta}_x^{\tau+1} = \arg\max_{\boldsymbol{\theta}_x} \mathbb{E}\left[\ln p(\boldsymbol{X}; \boldsymbol{\theta}_x) \,\middle|\, \boldsymbol{R}_1^t; \boldsymbol{\gamma}_1^\tau, \boldsymbol{\theta}_x^\tau\right] \tag{3.38}$$

and

$$\boldsymbol{\gamma}_1^{\tau+1} = \arg\max_{\boldsymbol{\gamma}_1} \mathbb{E}\left[\ln p(\boldsymbol{R}_1^t|\boldsymbol{X};\boldsymbol{\gamma}_1)\,\middle|\,\boldsymbol{R}_1^t;\boldsymbol{\gamma}_1^\tau,\boldsymbol{\theta}_x^\tau\right] \tag{3.39}$$

$$= \arg\max_{\boldsymbol{\gamma}_1} \sum_{l=1}^{L} \mathbb{E}\left[\ln p(\boldsymbol{r}_{1,l}^t|\boldsymbol{x}_l;\gamma_{1,l})\,\middle|\,\boldsymbol{r}_{1,l}^t;\gamma_{1,l}^\tau,\boldsymbol{\theta}_x^\tau\right], \tag{3.40}$$

where the latter optimization decouples further into

$$\gamma_{1,l}^{\tau+1} = \arg\max_{\gamma_{1,l}} \left\{ \frac{N}{2}\ln\gamma_{1,l} - \frac{\gamma_{1,l}}{2}\mathbb{E}\left[\|\boldsymbol{x}_l - \boldsymbol{r}_{1,l}^t\|_2^2\,\middle|\,\boldsymbol{r}_{1,l}^t;\gamma_{1,l}^\tau,\boldsymbol{\theta}_x^\tau\right]\right\} \tag{3.41}$$

$$= N\left\{\mathbb{E}\left[\|\boldsymbol{x}_l - \boldsymbol{r}_{1,l}^t\|_2^2\,\middle|\,\boldsymbol{r}_{1,l}^t;\gamma_{1,l}^\tau,\boldsymbol{\theta}_x^\tau\right]\right\}^{-1} \tag{3.42}$$

$$= \left\{\frac{1}{N}\sum_{n=1}^{N}\mathbb{E}\left[(x_{nl} - r_{1,nl}^t)^2\,\middle|\,\boldsymbol{r}_{1,l}^t;\gamma_{1,l}^\tau,\boldsymbol{\theta}_x^\tau\right]\right\}^{-1} \tag{3.43}$$

$$= \left\{\frac{1}{N}\|\boldsymbol{x}_{1,l}^\tau - \boldsymbol{r}_{1,l}^t\|^2 + \frac{1}{\eta_{1,l}^\tau}\right\}^{-1}, \tag{3.44}$$

for $l = 1,\ldots,L$ and

$$\boldsymbol{x}_{1,l}^\tau \triangleq \mathbb{E}\left[\boldsymbol{x}_l\middle|\boldsymbol{r}_{1,l}^t;\gamma_{1,l}^\tau,\boldsymbol{\theta}_x^\tau\right] \tag{3.45}$$

$$= \boldsymbol{g}_1(\boldsymbol{r}_{1,l}^t,\gamma_{1,l}^\tau;\boldsymbol{\theta}_x^\tau) \tag{3.46}$$

$$1/\eta_{1,l}^\tau \triangleq \mathrm{tr}\left\{\,\mathrm{Cov}\left[\boldsymbol{x}_l\middle|\boldsymbol{r}_{1,l}^t;\gamma_{1,l}^\tau,\boldsymbol{\theta}_x^\tau\right]\right\}/N \tag{3.47}$$

$$= \langle \boldsymbol{g}_1'(\boldsymbol{r}_{1,l}^t,\gamma_{1,l}^\tau;\boldsymbol{\theta}_x^\tau)\rangle/\gamma_{1,l}^\tau. \tag{3.48}$$

Above, we detailed the re-estimation of $\boldsymbol{\gamma}_1^t$. A similar procedure can be used for re-estimation of the precisions $\boldsymbol{\gamma}_2^t$. The resulting Bilinear Adaptive VAMP (BAd-VAMP) is summarized in Alg. 4 using $\tau_{1,\max}$ EM iterations for the first inner loop and $\tau_{2,\max}$ EM iterations for the second inner loop. To avoid the complications of a dual-index notation (i.e., t and τ), we use only the single index t in Alg. 4 and over-write the quantities in each inner loop. Note that, when $\tau_{1,\max} = \tau_{2,\max} = 0$, BAd-VAMP (i.e., Alg. 4) reduces to bilinear EM-VAMP (i.e., Alg. 3). In practice, we

noticed that re-estimating the precision $\boldsymbol{\gamma}_1$ is important especially in problems like dictionary learning and often a single EM iteration for $\boldsymbol{\gamma}_1$ (i.e., $\tau_{1,\max} = 1$) suffices. Re-estimating the precisions $\boldsymbol{\gamma}_2$ in VAMP can be very computationally expensive as it involves a matrix inversion, however, in many applications we noticed that $\boldsymbol{\gamma}_2$ is often accurate and we can skip it.

Algorithm 4 Bilinear Adaptive VAMP

initialize:
  $\forall l : \boldsymbol{r}_{1,l}^0, \gamma_{1,l}^0, \boldsymbol{\theta}_x^0, \boldsymbol{\theta}_A^0, \gamma_w^0$
for $t = 0, \dots, T_{\max}$ **do**
  for $\tau = 0, \dots, \tau_{1,\max}$ **do**
    $\forall l : \boldsymbol{x}_{1,l}^t \leftarrow \boldsymbol{g}_1(\boldsymbol{r}_{1,l}^t, \gamma_{1,l}^t; \boldsymbol{\theta}_x^t)$
    $\forall l : 1/\eta_{1,l}^t \leftarrow \langle \boldsymbol{g}_1'(\boldsymbol{r}_{1,l}^t, \gamma_{1,l}^t; \boldsymbol{\theta}_x^t) \rangle / \gamma_{1,l}^t$
    $\forall l : 1/\gamma_{1,l}^t \leftarrow \frac{1}{N} \|\boldsymbol{x}_{1,l}^t - \boldsymbol{r}_{1,l}^t\|^2 + 1/\eta_{1,l}^t$
    $q_1^t(\boldsymbol{X}) \propto \prod_{l=1}^L p_{\mathbf{x}}(\boldsymbol{x}_l; \boldsymbol{\theta}_x^t) e^{-\frac{1}{2}\gamma_{1,l}^t \|\boldsymbol{x}_l - \boldsymbol{r}_{1,l}^t\|^2}$
    $\boldsymbol{\theta}_x^t \leftarrow \arg\max_{\boldsymbol{\theta}_x} \mathbb{E}[\ln p_{\mathbf{X}}(\boldsymbol{X}; \boldsymbol{\theta}_x) | q_1^t]$
  end for
  $\boldsymbol{\theta}_x^{t+1} = \boldsymbol{\theta}_x^t$
  $\forall l : \gamma_{2,l}^t = \eta_{1,l}^t - \gamma_{1,l}^t$
  $\forall l : \boldsymbol{r}_{2,l}^t = (\eta_{1,l}^t \boldsymbol{x}_{1,l}^t - \gamma_{1,l}^t \boldsymbol{r}_{1,l}^t)/\gamma_{2,l}^t$
  for $\tau = 0, \dots, \tau_{2,\max}$ **do**
    $\forall l : \boldsymbol{x}_{2,l}^t \leftarrow \boldsymbol{g}_{2,l}(\boldsymbol{r}_{2,l}^t, \gamma_{2,l}^t; \boldsymbol{\theta}_A^t, \gamma_w^t)$
    $\forall l : 1/\eta_{2,l}^t \leftarrow \langle \boldsymbol{g}_{2,l}'(\boldsymbol{r}_{2,l}^t, \gamma_{2,l}^t; \boldsymbol{\theta}_A^t, \gamma_w^t) \rangle / \gamma_{2,l}^t$
    $\forall l : 1/\gamma_{2,l}^t \leftarrow \frac{1}{N} \|\boldsymbol{x}_{2,l}^t - \boldsymbol{r}_{2,l}^t\|^2 + 1/\eta_{2,l}^t$
    $q_2^t(\boldsymbol{X}) \propto \prod_l p_{\mathbf{y}|\mathbf{x}}(\boldsymbol{y}_l | \boldsymbol{x}_l; \boldsymbol{\theta}_A^t, \gamma_w^t) e^{-\frac{1}{2}\gamma_{2,l}^t \|\boldsymbol{x}_l - \boldsymbol{r}_{2,l}^t\|^2}$
    $\boldsymbol{\theta}_A^t \leftarrow \arg\max_{\boldsymbol{\theta}_A} \mathbb{E}[\ln p_{\mathbf{Y}|\mathbf{X}}(\boldsymbol{Y} | \boldsymbol{X}; \boldsymbol{\theta}_A, \gamma_w^t) | \boldsymbol{Y}, q_2^t]$
    $\gamma_w^t \leftarrow \arg\max_{\gamma_w} \mathbb{E}[\ln p_{\mathbf{Y}|\mathbf{X}}(\boldsymbol{Y} | \boldsymbol{X}; \boldsymbol{\theta}_A^t, \gamma_w) | \boldsymbol{Y}, q_2^t]$
  end for
  $\boldsymbol{\theta}_A^{t+1} = \boldsymbol{\theta}_A^t$
  $\gamma_w^{t+1} = \gamma_w^t$
  $\forall l : \gamma_{1,l}^{t+1} = \eta_{2,l}^t - \gamma_{2,l}^t$
  $\forall l : \boldsymbol{r}_{1,l}^{t+1} = (\eta_{2,l}^t \boldsymbol{x}_{2,l}^t - \gamma_{2,l}^t \boldsymbol{r}_{2,l}^t)/\gamma_{1,l}^{t+1}$
end for

We now provide additional details on the steps in Alg. 4.

3.7.1 Estimating the Matrix $\boldsymbol{X}$

Recalling the definition of $\boldsymbol{g}_{2,l}$ in (3.28), the form of $p_{\mathbf{y}|\mathbf{x}}$ in (3.15) implies that

$$\boldsymbol{g}_{2,l}(\boldsymbol{r}_{2,l}^t, \gamma_{2,l}^t; \boldsymbol{\theta}_A^t, \gamma_w^t) = \boldsymbol{C}_l^t\big(\gamma_{2,l}^t \boldsymbol{r}_{2,l}^t + \gamma_w^t \boldsymbol{A}(\boldsymbol{\theta}_A^t)^{\mathsf{T}} \boldsymbol{y}_l\big) \tag{3.49}$$

$$\langle \boldsymbol{g}'_{2,l}(\boldsymbol{r}_{2,l}^t, \gamma_{2,l}^t; \boldsymbol{\theta}_A^t, \gamma_w^t)\rangle = \gamma_{2,l}^t \operatorname{tr}\left\{\boldsymbol{C}_l^t\right\}/N \tag{3.50}$$

for the covariance matrix

$$\boldsymbol{C}_l^t \triangleq \big(\gamma_{2,l}^t \boldsymbol{I}_N + \gamma_w^t \boldsymbol{A}(\boldsymbol{\theta}_A^t)^{\mathsf{T}} \boldsymbol{A}(\boldsymbol{\theta}_A^t)\big)^{-1}. \tag{3.51}$$

To avoid computing a separate matrix inverse (3.51) for each column $l = 1, \ldots, L$, one could instead compute the eigenvalue decomposition

$$\boldsymbol{A}(\boldsymbol{\theta}_A^t)^{\mathsf{T}} \boldsymbol{A}(\boldsymbol{\theta}_A^t) = \boldsymbol{U}^t \operatorname{Diag}(\boldsymbol{s}^t) \boldsymbol{U}^{t\mathsf{T}}, \tag{3.52}$$

and then leverage the fact that

$$\boldsymbol{C}_l^t = \big(\gamma_{2,l}^t \boldsymbol{I} + \gamma_w^t \boldsymbol{U}^t \operatorname{Diag}(\boldsymbol{s}^t) \boldsymbol{U}^{t\mathsf{T}}\big)^{-1} \tag{3.53}$$

$$= \big(\boldsymbol{U}^t \left(\gamma_{2,l}^t \boldsymbol{I} + \gamma_w^t \operatorname{Diag}(\boldsymbol{s}^t)\right) \boldsymbol{U}^{t\mathsf{T}}\big)^{-1} \tag{3.54}$$

$$= \boldsymbol{U}^t \operatorname{Diag}(\gamma_{2,l}^t \mathbf{1} + \gamma_w^t \boldsymbol{s}^t)^{-1} \boldsymbol{U}^{t\mathsf{T}}, \tag{3.55}$$

which reduces to the inversion of a diagonal matrix for each $l = 1, \ldots, L$.

3.7.2 Learning the Parameters $\boldsymbol{\theta}_A$

We now provide details on the update of $\boldsymbol{\theta}_A$ line 18 of Alg. 4. Given the form of $p_{\mathbf{Y}|\mathbf{X}}$ in (3.14)-(3.15), we have that

$$\ln p_{\mathbf{Y}|\mathbf{X}}(\boldsymbol{Y}|\boldsymbol{X};\boldsymbol{\theta}_A,\gamma_w) = \tfrac{ML}{2}\ln\gamma_w - \tfrac{\gamma_w}{2}\|\boldsymbol{Y}-\boldsymbol{A}(\boldsymbol{\theta}_A)\boldsymbol{X}\|_F^2 + \text{const} \tag{3.56}$$

$$= \tfrac{ML}{2}\ln\gamma_w - \tfrac{\gamma_w}{2}\big(\operatorname{tr}\{\boldsymbol{Y}\boldsymbol{Y}^{\mathsf{T}}\} - 2\operatorname{tr}\{\boldsymbol{A}(\boldsymbol{\theta}_A)\boldsymbol{X}\boldsymbol{Y}^{\mathsf{T}}\} + \operatorname{tr}\{\boldsymbol{A}(\boldsymbol{\theta}_A)\boldsymbol{X}\boldsymbol{X}^{\mathsf{T}}\boldsymbol{A}(\boldsymbol{\theta}_A)^{\mathsf{T}}\}\big) + \text{const}. \tag{3.57}$$

Since the first and second moments of belief q_2^t are the following

$$\mathbb{E}\left[\boldsymbol{X}\middle|q_2^t\right] = \boldsymbol{X}_2^t \tag{3.58}$$

$$\mathbb{E}\left[\boldsymbol{X}\boldsymbol{X}^{\mathsf{T}}\middle|q_2^t\right] = \sum_{l=1}^{L}\mathbb{E}\left[\boldsymbol{x}_l\boldsymbol{x}_l^{\mathsf{T}}\middle|q_{2,l}^t\right] = \boldsymbol{X}_2^t\boldsymbol{X}_2^{t\mathsf{T}} + \underbrace{\sum_{l=1}^{L}\boldsymbol{C}_l^t}_{\triangleq\,\boldsymbol{C}^t}, \tag{3.59}$$

we have that

$$\begin{aligned}
&\mathbb{E}\left[\ln p_{\mathbf{Y}|\mathbf{X}}(\boldsymbol{Y}|\boldsymbol{X};\boldsymbol{\theta}_A,\gamma_w)\middle|\boldsymbol{Y},q_2^t\right] \\
&= \frac{ML}{2}\ln\gamma_w - \frac{\gamma_w}{2}\Big(\operatorname{tr}\left\{\boldsymbol{Y}\boldsymbol{Y}^{\mathsf{T}}\right\} - 2\operatorname{tr}\left\{\boldsymbol{A}(\boldsymbol{\theta}_A)\boldsymbol{X}_2^t\boldsymbol{Y}^{\mathsf{T}}\right\} \\
&\quad + \operatorname{tr}\left\{\boldsymbol{A}(\boldsymbol{\theta}_A)\boldsymbol{X}_2^t\boldsymbol{X}_2^{t\mathsf{T}}\boldsymbol{A}(\boldsymbol{\theta}_A)^{\mathsf{T}}\right\} + \operatorname{tr}\left\{\boldsymbol{A}(\boldsymbol{\theta}_A)\boldsymbol{C}^t\boldsymbol{A}(\boldsymbol{\theta}_A)^{\mathsf{T}}\right\}\Big) \qquad (3.60)\\
&= \frac{ML}{2}\ln\gamma_w - \frac{\gamma_w}{2}\Big(\|\boldsymbol{Y}-\boldsymbol{A}(\boldsymbol{\theta}_A)\boldsymbol{X}_2^t\|_F^2 \\
&\quad + \operatorname{tr}\left\{\boldsymbol{A}(\boldsymbol{\theta}_A)\boldsymbol{C}^t\boldsymbol{A}(\boldsymbol{\theta}_A)^{\mathsf{T}}\right\}\Big) + \text{const}. \qquad (3.61)
\end{aligned}$$

To maximize (3.61) over $\boldsymbol{\theta}_A = [\theta_{A,1},\ldots,\theta_{A,Q}]$ with fixed γ_w, we consider the affine-linear model

$$\boldsymbol{A}(\boldsymbol{\theta}_A) = \boldsymbol{A}_0 + \sum_{i=1}^{Q}\theta_{A,i}\boldsymbol{A}_i, \tag{3.62}$$

noting that non-linear models could be handled using similar techniques. Plugging (3.62) into (3.60), we get

$$\begin{aligned}
&\mathbb{E}\left[\ln p_{\mathbf{Y}|\mathbf{X}}(\boldsymbol{Y}|\boldsymbol{X};\boldsymbol{\theta}_A,\gamma_w)\middle|\boldsymbol{Y},q_2^t\right] \\
&= \text{const} - \frac{\gamma_w}{2}\sum_{i=1}^{Q}\sum_{j=1}^{Q}\theta_{A,i}\,\text{tr}\{\boldsymbol{A}_i(\boldsymbol{C}^t+\boldsymbol{X}_2^t\boldsymbol{X}_2^{t\mathsf{T}})\boldsymbol{A}_j^{\mathsf{T}}\}\theta_{A,j} \\
&\quad - \gamma_w\sum_{i=1}^{Q}\theta_{A,i}\big(\text{tr}\{\boldsymbol{A}_i(\boldsymbol{C}^t+\boldsymbol{X}_2^t\boldsymbol{X}_2^{t\mathsf{T}})\boldsymbol{A}_0^{\mathsf{T}}\}-\text{tr}\{\boldsymbol{A}_i\boldsymbol{X}_2^t\boldsymbol{Y}^{\mathsf{T}}\}\big) \qquad (3.63)\\
&= -\frac{\gamma_w}{2}\Big(\boldsymbol{\theta}_A^{\mathsf{T}}\boldsymbol{H}^t\boldsymbol{\theta}_A - 2\boldsymbol{\theta}_A^{\mathsf{T}}\boldsymbol{\beta}^t\Big)+\text{const} \qquad (3.64)
\end{aligned}$$

for the quantities $\boldsymbol{H}^t$ and $\boldsymbol{\beta}^t$ defined as

$$\begin{aligned}
[\boldsymbol{H}^t]_{ij} &\triangleq \text{tr}\left\{\boldsymbol{A}_i(\boldsymbol{C}^t+\boldsymbol{X}_2^t\boldsymbol{X}_2^{t\mathsf{T}})\boldsymbol{A}_j^{\mathsf{T}}\right\} &(3.65)\\
&= \text{tr}\left\{\boldsymbol{A}_j^{\mathsf{T}}\boldsymbol{A}_i(\boldsymbol{C}^t+\boldsymbol{X}_2^t\boldsymbol{X}_2^{t\mathsf{T}})\right\} &(3.66)\\
[\boldsymbol{\beta}^t]_i &\triangleq \text{tr}\left\{\boldsymbol{A}_i\boldsymbol{X}_2^t\boldsymbol{Y}^{\mathsf{T}}\right\}-\text{tr}\left\{\boldsymbol{A}_i(\boldsymbol{C}^t+\boldsymbol{X}_2^t\boldsymbol{X}_2^{t\mathsf{T}})\boldsymbol{A}_0^{\mathsf{T}}\right\} &(3.67)\\
&= \text{tr}\left\{\boldsymbol{Y}^{\mathsf{T}}\boldsymbol{A}_i\boldsymbol{X}_2^t\right\}-\text{tr}\left\{\boldsymbol{A}_0^{\mathsf{T}}\boldsymbol{A}_i(\boldsymbol{C}^t+\boldsymbol{X}_2^t\boldsymbol{X}_2^{t\mathsf{T}})\right\}, &(3.68)
\end{aligned}$$

where $\boldsymbol{A}_j^{\mathsf{T}}\boldsymbol{A}_i$ and $\boldsymbol{Y}^{\mathsf{T}}\boldsymbol{A}_i$ can be pre-computed to speed up the algorithm. We can find the maximizer of (3.64) by computing its gradient w.r.t. $\boldsymbol{\theta}_A$ and setting it to zero. We then get the update $\boldsymbol{\theta}_A^{t+1}$ as

$$\begin{aligned}
\boldsymbol{H}^t\boldsymbol{\theta}_A^{t+1} &= \boldsymbol{\beta}^t &(3.69)\\
\Rightarrow \boldsymbol{\theta}_A^{t+1} &= (\boldsymbol{H}^t)^{-1}\boldsymbol{\beta}^t. &(3.70)
\end{aligned}$$

A special case of (3.62) is where $\boldsymbol{A}(\cdot)$ has no structure, i.e.,

$$\boldsymbol{A}(\boldsymbol{\theta}_A)=\sum_{m=1}^{M}\sum_{n=1}^{N}\theta_{A,m,n}\boldsymbol{e}_m\boldsymbol{e}_n^{\mathsf{T}}. \qquad (3.71)$$

where $\boldsymbol{e}_m$ denotes the mth standard basis vector. In this case, it can be shown that

$$\boldsymbol{A}(\boldsymbol{\theta}_A^{t+1})=\boldsymbol{Y}\boldsymbol{X}_2^{t\mathsf{T}}\big(\boldsymbol{C}^t+\boldsymbol{X}_2^t\boldsymbol{X}_2^{t\mathsf{T}}\big)^{-1}. \qquad (3.72)$$

3.7.3 Learning the Noise Precision γ_w

We now provide details on the update of the noise precision γ_w in line 19 of Alg. 4. To maximize (3.61) over γ_w with fixed $\boldsymbol{\theta}_A = \boldsymbol{\theta}_A^{t+1}$, we search for the values of γ_w that zero the derivative of (3.61). The unique solution is straightforwardly shown to be

$$1/\gamma_w^{t+1} = \frac{1}{ML}\Big(\|\boldsymbol{Y} - \boldsymbol{A}(\boldsymbol{\theta}_A^{t+1})\boldsymbol{X}_2^t\|_F^2 + \operatorname{tr}\left\{\boldsymbol{A}(\boldsymbol{\theta}_A^{t+1})\boldsymbol{C}^t\boldsymbol{A}(\boldsymbol{\theta}_A^{t+1})^{\mathsf{T}}\right\}\Big). \tag{3.73}$$

In Alg. 5, BAd-VAMP is rewritten with detailed expressions for the updates of $\boldsymbol{x}_{2,l}^t$, $\eta_{2,l}^t$, $\boldsymbol{\theta}_A^t$, and γ_w^t.

Algorithm 5 Bilinear Adaptive VAMP (Detailed)

initialize:
  $\forall l : \boldsymbol{r}_{1,l}^0, \gamma_{1,l}^0, \boldsymbol{\theta}_x^0, \boldsymbol{\theta}_A^0, \gamma_w^0$
for $t = 0, \ldots, T_{\max}$ **do**
  for $\tau = 0, \ldots, \tau_{1,\max}$ **do**
    $\forall l : \boldsymbol{x}_{1,l}^t \leftarrow \boldsymbol{g}_1(\boldsymbol{r}_{1,l}^t, \gamma_{1,l}^t; \boldsymbol{\theta}_x^t)$
    $\forall l : 1/\eta_{1,l}^t \leftarrow \langle \boldsymbol{g}_1'(\boldsymbol{r}_{1,l}^t, \gamma_{1,l}^t; \boldsymbol{\theta}_x^t) \rangle / \gamma_{1,l}^t$
    $\forall l : 1/\gamma_{1,l}^t \leftarrow \frac{1}{N} \|\boldsymbol{x}_{1,l}^t - \boldsymbol{r}_{1,l}^t\|^2 + 1/\eta_{1,l}^t$
    $q_1^t(\boldsymbol{X}) \propto \prod_{l=1}^L p_{\mathbf{x}}(\boldsymbol{x}_l; \boldsymbol{\theta}_x^t) e^{-\frac{1}{2}\gamma_{1,l}^t \|\boldsymbol{x}_l - \boldsymbol{r}_{1,l}^t\|^2}$
    $\boldsymbol{\theta}_x^t \leftarrow \arg\max_{\boldsymbol{\theta}_x} \mathbb{E}[\ln p_{\mathbf{X}}(\boldsymbol{X}; \boldsymbol{\theta}_x) | q_1^t]$
  end for
  $\boldsymbol{\theta}_x^{t+1} = \boldsymbol{\theta}_x^t$
  $\forall l : \gamma_{2,l}^t = \eta_{1,l}^t - \gamma_{1,l}^t$
  $\forall l : \boldsymbol{r}_{2,l}^t = (\eta_{1,l}^t \boldsymbol{x}_{1,l}^t - \gamma_{1,l}^t \boldsymbol{r}_{1,l}^t)/\gamma_{2,l}^t$
  for $\tau = 0, \ldots, \tau_{2,\max}$ **do**
    $\forall l : \boldsymbol{C}_l^t \leftarrow \big(\gamma_{2,l}^t \boldsymbol{I}_N + \gamma_w^t \boldsymbol{A}(\boldsymbol{\theta}_A^t)^{\mathsf{T}} \boldsymbol{A}(\boldsymbol{\theta}_A^t)\big)^{-1}$
    $\forall l : \boldsymbol{x}_{2,l}^t \leftarrow \boldsymbol{C}_l^t \big(\gamma_{2,l}^t \boldsymbol{r}_{2,l}^t + \gamma_w^t \boldsymbol{A}(\boldsymbol{\theta}_A^t)^{\mathsf{T}} \boldsymbol{y}_l\big)$
    $\forall l : 1/\eta_{2,l}^t \leftarrow \mathrm{tr}\{\boldsymbol{C}_l^t\}/N$
    $\boldsymbol{C}^t \leftarrow \sum_{l=1}^L \boldsymbol{C}_l^t$
    $\forall i, j : [\boldsymbol{H}^t]_{ij} \leftarrow \mathrm{tr}\big\{\boldsymbol{A}_j^{\mathsf{T}} \boldsymbol{A}_i (\boldsymbol{C}^t + \boldsymbol{X}_2^t \boldsymbol{X}_2^{t\mathsf{T}})\big\}$
    $\forall i : [\boldsymbol{\beta}^t]_i \leftarrow \mathrm{tr}\big\{\boldsymbol{Y}^{\mathsf{T}} \boldsymbol{A}_i \boldsymbol{X}_2^t\big\} - \mathrm{tr}\big\{\boldsymbol{A}_0^{\mathsf{T}} \boldsymbol{A}_i (\boldsymbol{C}^t + \boldsymbol{X}_2^t \boldsymbol{X}_2^{t\mathsf{T}})\big\}$
    $\boldsymbol{\theta}_A^t \leftarrow (\boldsymbol{H}^t)^{-1} \boldsymbol{\beta}^t$
    $1/\gamma_w^t \leftarrow \frac{1}{ML}\big(\|\boldsymbol{Y} - \boldsymbol{A}(\boldsymbol{\theta}^t)\boldsymbol{X}_2^t\|_F^2 + \mathrm{tr}\big\{\boldsymbol{A}(\boldsymbol{\theta}_A^t) \boldsymbol{C}^t \boldsymbol{A}(\boldsymbol{\theta}_A^t)^{\mathsf{T}}\big\}\big)$
  end for
  $\boldsymbol{\theta}_A^{t+1} = \boldsymbol{\theta}_A^t$
  $\gamma_w^{t+1} = \gamma_w^t$
  $\forall l : \gamma_{1,l}^{t+1} = \eta_{2,l}^t - \gamma_{2,l}^t$
  $\forall l : \boldsymbol{r}_{1,l}^{t+1} = (\eta_{2,l}^t \boldsymbol{x}_{2,l}^t - \gamma_{2,l}^t \boldsymbol{r}_{2,l}^t)/\gamma_{1,l}^{t+1}$
end for

3.7.4 Algorithm Enhancements

We now propose several enhancements to the BAd-VAMP algorithm presented in Alg. 4 and detailed in Alg. 5.

Damping

For fixed $\boldsymbol{\Theta}^t$ and infinitely large right-rotationally invariant $\boldsymbol{A}(\boldsymbol{\theta}_A^t)$, the state-evolution of VAMP guarantees its convergence. But when $\boldsymbol{A}(\boldsymbol{\theta}_A^t)$ deviates from this assumption, damping the VAMP iterations can help maintain convergence [75]. With damping, lines 25-26 of Alg. 5 (or lines 23-24 of Alg. 4) would be replaced by

$$\gamma_{1,l}^{t+1} = (1-\zeta)\gamma_{1,l}^t + \zeta(\eta_{2,l}^t - \gamma_{2,l}^t) \tag{3.74}$$

$$\boldsymbol{r}_{1,l}^{t+1} = (1-\zeta)\boldsymbol{r}_{1,l}^t + \zeta(\eta_{2,l}^t \boldsymbol{x}_{2,l}^t - \gamma_{2,l}^t \boldsymbol{r}_{2,l}^t)/(\eta_{2,l}^t - \gamma_{2,l}^t) \tag{3.75}$$

for some $\zeta \in (0,1)$. We can perform a similar damping on $\{(\boldsymbol{r}_{2,l}^t, \gamma_{2,l}^t)\}_{l=1}^L$ and replace lines 11-12 of Alg. 5 (or lines 11-12 of Alg. 4) by the following,

$$\gamma_{2,l}^t = (1-\zeta)\gamma_{1,l}^{t-1} + \zeta(\eta_{1,l}^t - \gamma_{1,l}^t) \tag{3.76}$$

$$\boldsymbol{r}_{1,l}^t = (1-\zeta)\boldsymbol{r}_{1,l}^{t-1} + \zeta(\eta_{1,l}^t \boldsymbol{x}_{1,l}^t - \gamma_{1,l}^t \boldsymbol{r}_{1,l}^t)/(\eta_{1,l}^t - \gamma_{1,l}^t). \tag{3.77}$$

The case $\zeta = 1$ corresponds to no damping.

Negative precisions

Sometimes the precisions $\{\gamma_{1,l}, \gamma_{2,l}\}_{l=1}^L$ can be negative. To avoid the precisions from being negative, we suggest to restrict them to the interval $[\gamma_{\min}, \infty)$, for very small $\gamma_{\min} > 0$, in lines 11 and 25 of Alg. 5 (or lines 11 and 23 of Alg. 4).

Restarts

Due to the non-convex nature of the bilinear inference problem, the algorithm may get stuck at local minima or slowed by saddle points. To mitigate these issues, it sometimes helps to restart the algorithm. For each restart, we suggest to initialize $\boldsymbol{\theta}_A^0$ at the final estimate of $\boldsymbol{\theta}_A$ returned by the previous run.

3.8 Relation to Previous Work

The proposed Bilinear Adaptive VAMP algorithm extends the (linear) Adaptive VAMP algorithm of [37] from the case where $\boldsymbol{A}(\boldsymbol{\theta}_A)$ is known to the case where $\boldsymbol{A}(\boldsymbol{\theta}_A)$ is unknown. In the known-$\boldsymbol{A}(\boldsymbol{\theta}_A)$ setting, where $\boldsymbol{A}(\boldsymbol{\theta}_A)$ is infinitely large and right-rotationally invariant, it was rigorously established in [37] that Adaptive VAMP obeys a state-evolution similar to that of VAMP, and that its estimates of $\{\boldsymbol{\theta}_x, \gamma_w\}$ are asymptotically consistent if certain identifiability conditions are met, i.e., they converge to the true values as $t \to \infty$. As future work, it would be interesting to understand whether Bilinear Adaptive VAMP also obeys a state evolution for certain classes of $\boldsymbol{A}(\cdot)$.

The proposed BAd-VAMP algorithm targets the same class of bilinear recovery problems as the EM-PBiGAMP algorithm from [66], and both leverage EM for automated hyperparameter tuning. The BiGAMP algorithm [67]- [68] is a special case of the PBiGAMP algorithm [66]. BiGAMP applies to the recovery of $\boldsymbol{A}$ and $\boldsymbol{X}$ from measurements of the form $\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{X} + \boldsymbol{W}$, whereas PBiGAMP applies to the recovery of $\boldsymbol{b}$ and $\boldsymbol{X}$ from $\boldsymbol{Y} = \boldsymbol{A}(\boldsymbol{b})\boldsymbol{X} + \boldsymbol{W}$ under known affine linear $\boldsymbol{A}(\cdot)$. Both can be combined with EM for hyperparameter learning. Note that the "AMP" aspects of these algorithms are fundamentally different. PBiGAMP treats the vectors $\{\boldsymbol{a}_{i,j}\}$

in (3.2) as i.i.d. Gaussian for its derivation, whereas BAd-VAMP treats the matrix $\boldsymbol{A}(\boldsymbol{b}) = \sum_{i=1}^{Q} b_i \boldsymbol{A}_i$ as right rotationally-invariant for its derivation. The latter allows more freedom in the singular values of $\boldsymbol{A}(\boldsymbol{b})$, which leads to increased robustness in practice, as demonstrated by the numerical experiments in Sec. 3.9.

BAd-VAMP and lifted VAMP both leverage the VAMP approach from [75] to solve bilinear inference problems. However, they do so in very different ways. As discussed in Sec. 3.1, lifted VAMP "lifts" the bilinear problem into a higher-dimensional linear problem, and then uses non-separable denoising to jointly estimate $\boldsymbol{b}$ and $\boldsymbol{c}$ in (3.5). An unfortunate consequence of lifting is a possibly significant increase in computational complexity and memory. In contrast, BAd-VAMP avoids lifting, and it employs EM to estimate $\boldsymbol{b}$ and VAMP to estimate $\boldsymbol{c}$. Interestingly, Sec. 3.9 shows BAd-VAMP performing equal or better to lifted VAMP in all experiments.

3.9 Numerical Experiments

In this section, we present the results of numerical experiments that study the behavior of the BAd-VAMP algorithm from Alg. 5, in comparison to other state-of-the-art algorithms, on several bilinear recovery problems. In all cases, we ran BAd-VAMP with $\tau_{1,\max} = 1$ and $\tau_{2,\max} = 0$ inner EM iterations and we assumed that the signal prior $p_{\mathbf{x}}$ is fully known (i.e., $\boldsymbol{\theta}_x$ is known). We nominally used a damping coefficient of $\zeta = 0.8$ and minimum precision of $\gamma_{\min} = 10^{-6}$. We initialized BAd-VAMP by $\gamma_w^0 = 0.1$, $\gamma_{1,l}^0 = 10^{-3}$ $\forall l$, and we set $\boldsymbol{r}_{1,l}^0$ and $\boldsymbol{\theta}_A^0$ to random vectors drawn i.i.d. from $\mathcal{N}(0, 10)$ and $\mathcal{N}(0, 1)$ respectively. Unless otherwise noted, no restarts were used in that experiment.

3.9.1 CS with Matrix Uncertainty

We consider the compressive sensing (CS) with matrix uncertainty [105] problem that we discussed in Sec. 2.5.2. We describe the problem formulation here for convenience. The goal is to recover the K-sparse signal $\boldsymbol{c} \in \mathbb{R}^N$ and the uncertainty parameters $\boldsymbol{b} \in \mathbb{R}^Q$ from length-M measurements $\boldsymbol{y}$,

$$\boldsymbol{y} = \boldsymbol{A}(\boldsymbol{b})\boldsymbol{c} + \boldsymbol{w} \tag{3.78a}$$

$$\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma_w) \tag{3.78b}$$

$$\boldsymbol{A}(\boldsymbol{b}) = \boldsymbol{A}_0 + \sum_{i=1}^{Q} b_i \boldsymbol{A}_i, \tag{3.78c}$$

where $\{\boldsymbol{A}_i\}_{i=0}^Q$ are known matrices. For our experiments, we used $Q = 10$, $N = 256$ and $K = 10$. We selected the noise precision γ_w so that the SNR as defined is 40 dB,

$$\mathsf{SNR} \triangleq \frac{\mathbb{E}[\|\boldsymbol{A}(\boldsymbol{b})\boldsymbol{c}\|^2]}{\mathbb{E}[\|\boldsymbol{w}\|^2]} \tag{3.79}$$

$$= \frac{\gamma_w}{M}\,\mathbb{E}[\|\boldsymbol{A}(\boldsymbol{b})\boldsymbol{c}\|^2]. \tag{3.80}$$

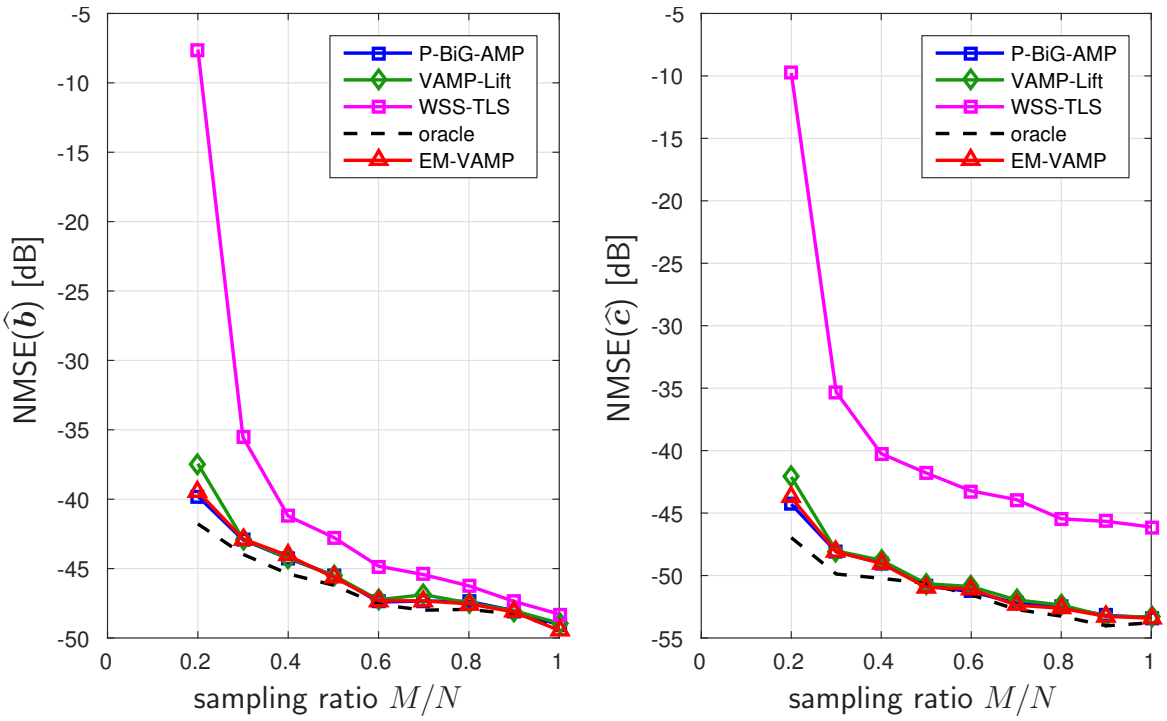

Figure 3.1: CS with matrix uncertainty: Median NMSE (over 50 trials) on signal $\boldsymbol{c}$ and uncertainty parameters $\boldsymbol{b}$ versus sampling ratio M/N.

Also, the uncertainty parameters $\boldsymbol{b}$ were drawn $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, and $\boldsymbol{c}$ was drawn with uniformly random support and with K non-zero elements from $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. We measured performance of our algorithm using the following metrics

$$\text{NMSE}(\widehat{\boldsymbol{b}}) \triangleq \frac{\|\widehat{\boldsymbol{b}} - \boldsymbol{b}\|^2}{\|\boldsymbol{b}\|^2} \tag{3.81}$$

$$\text{NMSE}(\widehat{\boldsymbol{c}}) \triangleq \frac{\|\widehat{\boldsymbol{c}} - \boldsymbol{c}\|^2}{\|\boldsymbol{c}\|^2}. \tag{3.82}$$

As a reference, we considered two oracle estimators: the MMSE estimator for $\boldsymbol{b}$ assuming $\boldsymbol{c}$ is known, and the MMSE estimator for $\boldsymbol{c}$ assuming $\boldsymbol{b}$ and the support of $\boldsymbol{c}$ are known.

For our first experiment, the elements of $\{\boldsymbol{A}_i\}_{i=1}^Q$ were drawn i.i.d. $\mathcal{N}(0,1)$ and the elements of $\boldsymbol{A}_0$ were drawn $\mathcal{N}(0,20)$. BAd-VAMP was run for a maximum of 200 iterations with a maximum of 2 restarts and damping $\zeta = 0.86$.

Figure 3.1 shows that the AMP-based algorithms gave near-oracle performance for the tested range of M/N, although lifted VAMP performed slightly worse than the others when $M/N = 0.2$. In contrast, the performance of WSS-TLS from the award-winning paper [105] was significantly worse than the AMP approaches. WSS-TLS aims to solve the non-convex optimization problem

$$(\widehat{\boldsymbol{b}}, \widehat{\boldsymbol{c}}) = \arg\min_{\boldsymbol{b},\boldsymbol{c}} \left\| \left(\boldsymbol{A}_0 + \sum_{i=1}^{Q} b_i \boldsymbol{A}_i \right) \boldsymbol{c} - \boldsymbol{y} \right\|^2 + \|\boldsymbol{b}\|^2/\gamma_w + \lambda \|\boldsymbol{c}\|_1 \tag{3.83}$$

using alternating minimization. For WSS-TLS, we used oracle knowledge of γ_w, oracle tuning of the regularization parameter λ, and code from the authors' website.

For our second experiment, we tested algorithm's robustness to non-zero mean in $\boldsymbol{A}(\boldsymbol{b})$[5], since this is a known issue with many AMP algorithms [18, 74, 96]. For this, we fixed the sampling ratio at $M/N = 0.6$, drew the elements of $\{\boldsymbol{A}_i\}_{i=1}^Q$ from i.i.d. $\mathcal{N}(\mu,1)$, and drew the elements of $\boldsymbol{A}_0$ from i.i.d. $\mathcal{N}(\mu,20)$. Figure 3.2 reports the median NMSE versus mean μ, and shows that BAd-VAMP is much more robust to $\mu > 0$ than the other tested AMP algorithms as well as WSS-TLS.

Figure 3.3 shows the runtime of the algorithms in Figure 3.1. Our implementation used MATLAB (R2015b) on an RHEL workstation with an 8-core Intel i7 processor. Although, for WSS-TLS, we used a grid-search to optimize λ in (3.83), Figure 3.3

[5]For the simpler case where $\boldsymbol{b}$ is known and the objective is to recover $\boldsymbol{c}$ from $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{c} + \boldsymbol{w}$, modifications of AMP that temporarily remove the mean from $\boldsymbol{A}$ have been proposed [96]. However, it is not clear how to extend this approach to the bilinear problem of recovering $\boldsymbol{b}$ and $\boldsymbol{c}$ from $\boldsymbol{y} = \boldsymbol{A}(\boldsymbol{b})\boldsymbol{c} + \boldsymbol{w}$.

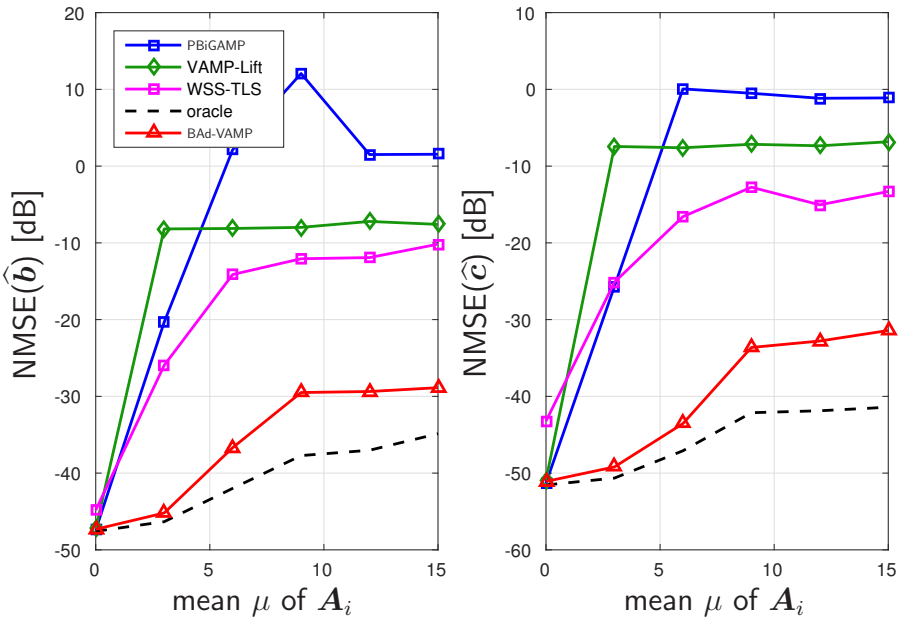

Figure 3.2: CS with matrix uncertainty: Median NMSE (over 50 trials) on signal $\boldsymbol{c}$ and uncertainty parameters $\boldsymbol{b}$ versus mean of matrices $\boldsymbol{A}_i$ at $M/N = 0.6$.

only shows the runtime of WSS-TLS after λ was chosen. Figure 3.3 shows BAd-VAMP running much faster than lifted VAMP and WSS-TLS, and slightly slower than PBiGAMP.

3.9.2 Self-Calibration

We consider the self-calibration problem [51] that we also discussed earlier in Sec. 2.5.2. We present the problem formulation here for convenience. The goal is to recover the K-sparse signal vector $\boldsymbol{c}$ and the calibration parameters $\boldsymbol{b}$ from measurements of the form $\boldsymbol{y} = \text{Diag}(\boldsymbol{H}\boldsymbol{b})\boldsymbol{\Psi}\boldsymbol{c}$ with known $\boldsymbol{H} \in \mathbb{R}^{M\times Q}$ and $\boldsymbol{\Psi} \in \mathbb{R}^{M\times N}$. Here, $\boldsymbol{H}\boldsymbol{b}$ represents an unknown vector of gains on the measurements, where the gain vector is believed to lie in the Q-dimensional subspace spanned by the columns of $\boldsymbol{H}$. For our experiment, $M = 128$, $N = 256$, $\boldsymbol{\Psi}$ and $\boldsymbol{b}$ where drawn i.i.d. $\mathcal{N}(0, 1)$, $\boldsymbol{H}$ was

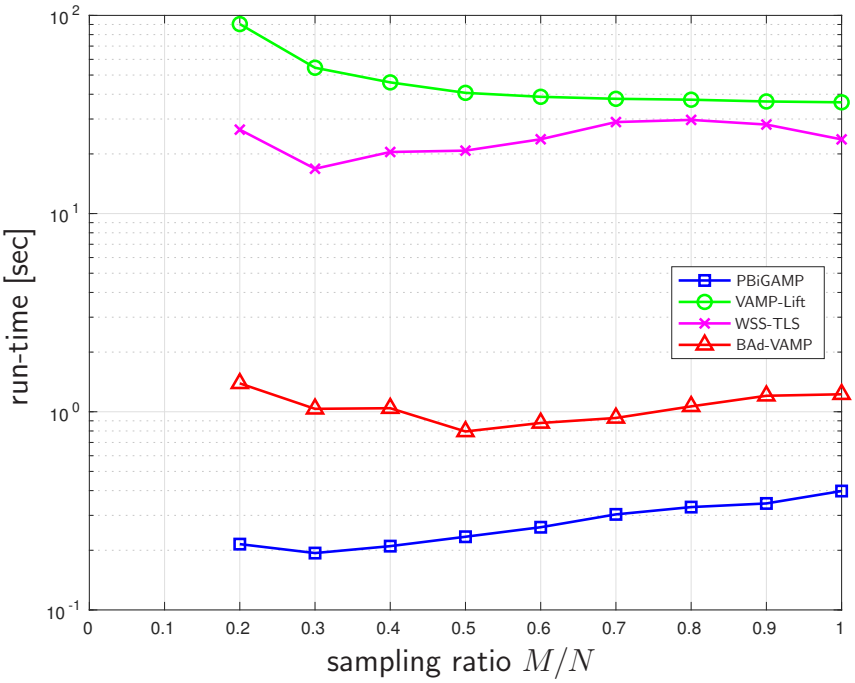

Figure 3.3: CS with matrix uncertainty: Median run-time in seconds (over 10 trials) versus sampling ratio M/N.

constructed using Q randomly selected columns of the Hadamard matrix, and $\boldsymbol{c}$ was drawn with uniformly random support and with K non-zero elements from $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$.

Figure 3.4 shows the rate of successful recovery versus subspace dimension Q and sparsity K for several algorithms. A recovery $(\widehat{\boldsymbol{b}}, \widehat{\boldsymbol{c}})$ was considered "successful" when $\|\widehat{\boldsymbol{b}}\widehat{\boldsymbol{c}}^{\mathsf{T}} - \boldsymbol{b}\boldsymbol{c}^{\mathsf{T}}\|_F^2 / \|\boldsymbol{b}\boldsymbol{c}^{\mathsf{T}}\|_F^2 \leq -50$ dB. From the figure, we see that the performance of BAd-VAMP is similar to that of EM-PBiGAMP, and even slightly better when Q is small and K is large. Meanwhile, BAd-VAMP appears significantly better than both lifted VAMP and SparseLift from [51]. SparseLift is a convex relaxation with provable guarantees [51]. For computational reasons (recall the discussion in Sec. 3.2), it was difficult to simulate lifted VAMP for $Q \geq 10$.

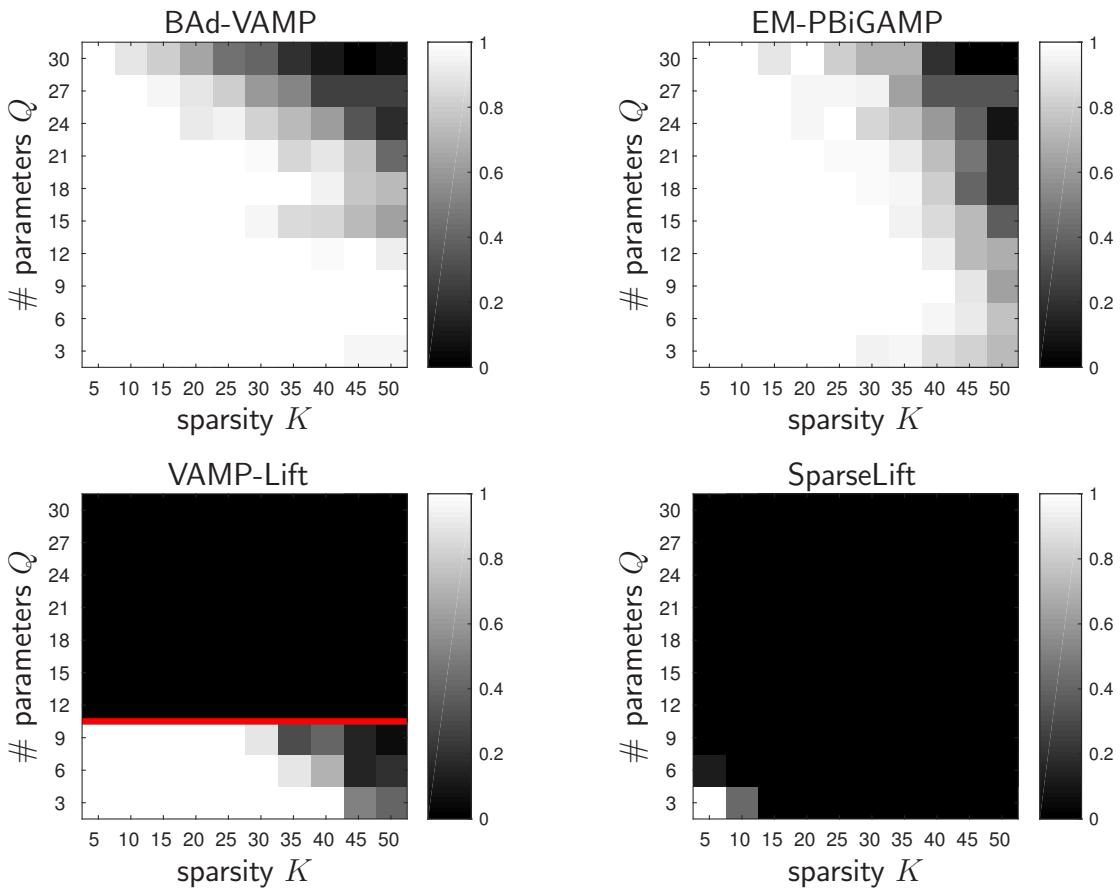

Figure 3.4: Self-calibration: Empirical success rate (over 50 trials) for several algorithms versus number of calibration parameters Q and sparsity K. For computational reasons, VAMP-Lift was simulated only for $Q < 10$.

3.9.3 Calibration in Tomography

We consider the problem of reconstructing an image from a sequence of tomographic projections, where the projections along each direction are scaled by an unknown calibration gain. In particular, let $\boldsymbol{\Psi}_\omega$ be the tomographic projection matrix corresponding to angle $\omega \in [0, \pi]$. We used the matrix form of the Radon transform instead of the operator form (i.e., Radon transform) to avoid numerical error when implementing the adjoint. Our goal is to reconstruct the image $\boldsymbol{x}$ from measurements

$$\boldsymbol{y} = \begin{bmatrix} b_1 \boldsymbol{\Psi}_{\omega_1} \\ \vdots \\ b_K \boldsymbol{\Psi}_{\omega_K} \end{bmatrix} \boldsymbol{x} + \boldsymbol{w} \tag{3.84a}$$

$$\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma_w), \tag{3.84b}$$

where $b_k \sim \mathcal{N}(1, \sigma_b^2)$ are the unknown gains unknown. Note that, by defining

$$\boldsymbol{A}_k = \begin{bmatrix} \boldsymbol{0} \\ \vdots \\ \boldsymbol{\Psi}_{\omega_k} \\ \vdots \\ \boldsymbol{0} \end{bmatrix} \quad \text{for} \quad k = 1, \dots, K \tag{3.85}$$

we can write $\boldsymbol{y} = \boldsymbol{A}(\boldsymbol{b})\boldsymbol{x} + \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma_w)$ for $\boldsymbol{A}(\boldsymbol{b}) = \sum_{k=1}^K b_k \boldsymbol{A}_k$, which matches (3.8).

In an attempt to solve the above problem, we used BAd-VAMP to recover the image $\boldsymbol{x}$ while simultaneously learning the calibration gains $\boldsymbol{b}$. For this, we used BAd-VAMP in "plug-and-play" mode briefly described in Sec. 1.1, where the BM3D image denoiser [26] was used to implement the $\boldsymbol{g}_1(\cdot)$ function in Alg. 1. There is an inherent scaling ambiguity of the problem, i.e., if $(\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{b}})$ is a solution then so is $(\alpha\widehat{\boldsymbol{x}}, \alpha^{-1}\widehat{\boldsymbol{b}})$ for any $\alpha > 0$. To avoid the scaling problem, we scaled the image estimate $\widehat{\boldsymbol{x}}$ by the α that minimized $\|\boldsymbol{x} - \alpha\widehat{\boldsymbol{x}}\|$ before computing the PSNR.

As baselines, we also tested the VAMP [75], total variation (TV) [10, 82] and regularization-by-denoising (RED) [76,79] approaches (see descriptions below). These

approaches all assume a noisy *linear* data model of the form $\boldsymbol{y} = \widehat{\boldsymbol{A}}\boldsymbol{x} + \mathcal{N}(0, \gamma_w^{-1}\boldsymbol{I})$ with known $\widehat{\boldsymbol{A}}$. To apply them to (3.84b), we considered two cases: the *genie-calibrated* (GC) case, where a genie supplies the true gains $\boldsymbol{b}$ and the algorithm uses $\widehat{\boldsymbol{A}} = \boldsymbol{A}(\boldsymbol{b})$, and the *un-calibrated* (UC) case, where $\boldsymbol{b}$ is unknown and the algorithms assume $\widehat{\boldsymbol{A}} = \boldsymbol{A}(\boldsymbol{1})$. In the latter case, the $\widehat{\boldsymbol{A}}$-based model is mismatched to the data-generation model (3.84b). For fair comparison, we scaled the GC and UC image estimates $\widehat{\boldsymbol{x}}$ by the α that minimized $\|\boldsymbol{x} - \alpha\widehat{\boldsymbol{x}}\|_2$ before computing the PSNR.

We now provide additional details on the experimental setup. For $\boldsymbol{x}$, we used the modified Shepp-Logan phantom of size 64×64, shown in the top-left panel of Fig. 3.5. For $\boldsymbol{A}(\cdot)$, we used $K = 25$ projections spaced uniformly in $\omega \in [0, \pi]$. The calibration gains $\boldsymbol{b} \overset{\text{iid}}{\sim} \mathcal{N}(1, \sigma_b^2)$ were generated using $\sigma_b = 0.06$, and the noise precision γ_w was set to achieve an SNR of $\mathbb{E}[\|\boldsymbol{A}(\boldsymbol{b})\boldsymbol{x}\|^2]/\,\mathbb{E}[\|\boldsymbol{w}\|^2] = 40$ dB. The TV method [82] computes

$$\widehat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \left\{ \frac{1}{2}\|\boldsymbol{y} - \widehat{\boldsymbol{A}}\boldsymbol{x}\|_2^2 + \lambda_t \|\nabla \boldsymbol{x}\|_{2,1} \right\} \tag{3.86}$$

for the isotropic TV operator

$$\|\nabla \boldsymbol{x}\|_{2,1} = \sum_{i,j} \sqrt{(x_{i,j} - x_{i,j-1})^2 + (x_{i,j} - x_{i-1,j})^2}. \tag{3.87}$$

We solved (3.86) using FASTA [38], and tuned λ_t to maximize the PSNR. RED [76,79] solves the fixed-point equation

$$\widehat{\boldsymbol{A}}^{\mathsf{H}}(\widehat{\boldsymbol{A}}\widehat{\boldsymbol{x}} - \boldsymbol{y}) + \lambda_r(\widehat{\boldsymbol{x}} - \boldsymbol{\rho}(\widehat{\boldsymbol{x}}, \tau)) = \boldsymbol{0} \tag{3.88}$$

for $\widehat{\boldsymbol{x}}$, where $\boldsymbol{\rho}(\cdot, \tau)$ is an image denoising algorithm with noise-variance τ. For our experiment, we used BM3D for $\boldsymbol{\rho}(\cdot, \tau)$, solved (3.88) using the ADMM method from [79] with 200 iterations, and tuned both λ_r and τ to maximize PSNR. For BAd-VAMP, we initialized $\boldsymbol{b}$ to $\boldsymbol{1}$, used damping $\zeta = 0.1$, assumed known noise precision

Table 3.1: PSNR (dB) in the tomography experiment

measurements	BAd-VAMP	VAMP	RED	TV
genie calibrated (GC)	—	**39.57**	36.56	33.27
un-calibrated (UC)	**38.27**	31.62	31.24	26.79

γ_w, and used at most 100 iterations. For VAMP, we initialized $\gamma_1^0 = 10^{-4}$ and used at most 100 iterations.

Table 3.1 reports the median PSNR achieved by each algorithm across 10 random draws of $\boldsymbol{b}$ and $\boldsymbol{w}$, Fig. 3.5 shows example image recoveries, and Fig. 3.6 shows the corresponding error images. From Table 3.1, we see that the PSNR performance of BAd-VAMP (which does not know $\boldsymbol{b}$) is nearly as good as genie-calibrated VAMP, and 1.7 dB better than genie-calibrated RED. Furthermore, the PSNR performance of BAd-VAMP is more than 6.6 dB better than un-calibrated VAMP and 7 dB better than un-calibrated RED. The uncalibrated VAMP, RED, and TV recoveries in Fig. 3.5 are plagued by either streaking artifacts and/or loss of detail (e.g., note the disappearance of the small white dots in uncalibrated TV). But the BAdVAMP image recovery in Fig. 3.5 shows no streaking artifacts and a high level of detail. Likewise, Fig. 3.6 shows that TV has trouble correctly recovering the white outer ellipse, RED has trouble in the interior region, but BAd-VAMP does well throughout.

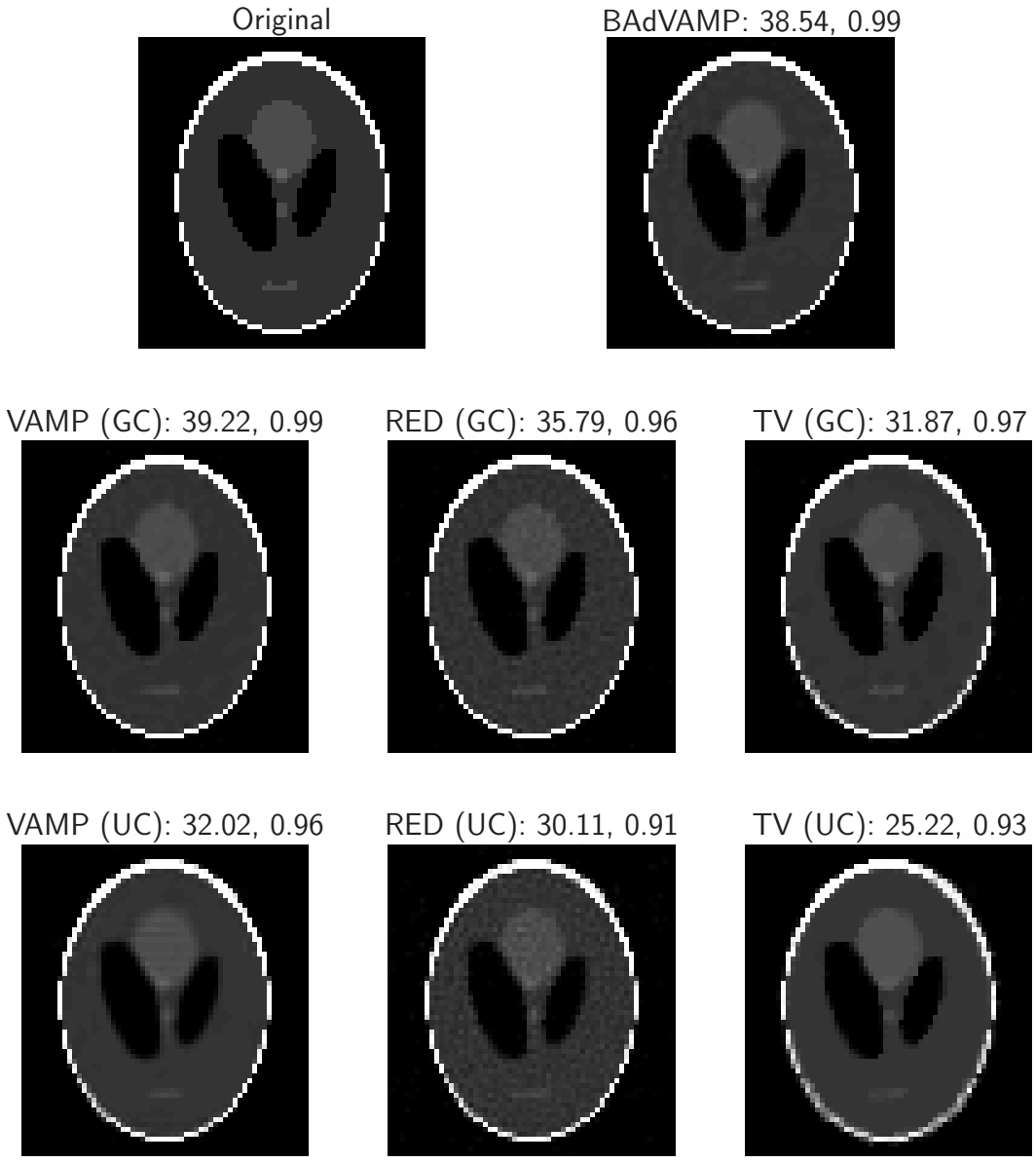

Figure 3.5: Calibration in tomography: Reconstruction PSNR in dB and SSIM of 64×64 Shepp-Logan phantom from 25 equally spaced tomographic projections. In the genie-calibrated (GC) case, $\widehat{\boldsymbol{A}} = \boldsymbol{A}(\boldsymbol{b})$, while in the un-calibrated (UC) case, $\widehat{\boldsymbol{A}} = \boldsymbol{A}(\boldsymbol{1})$.

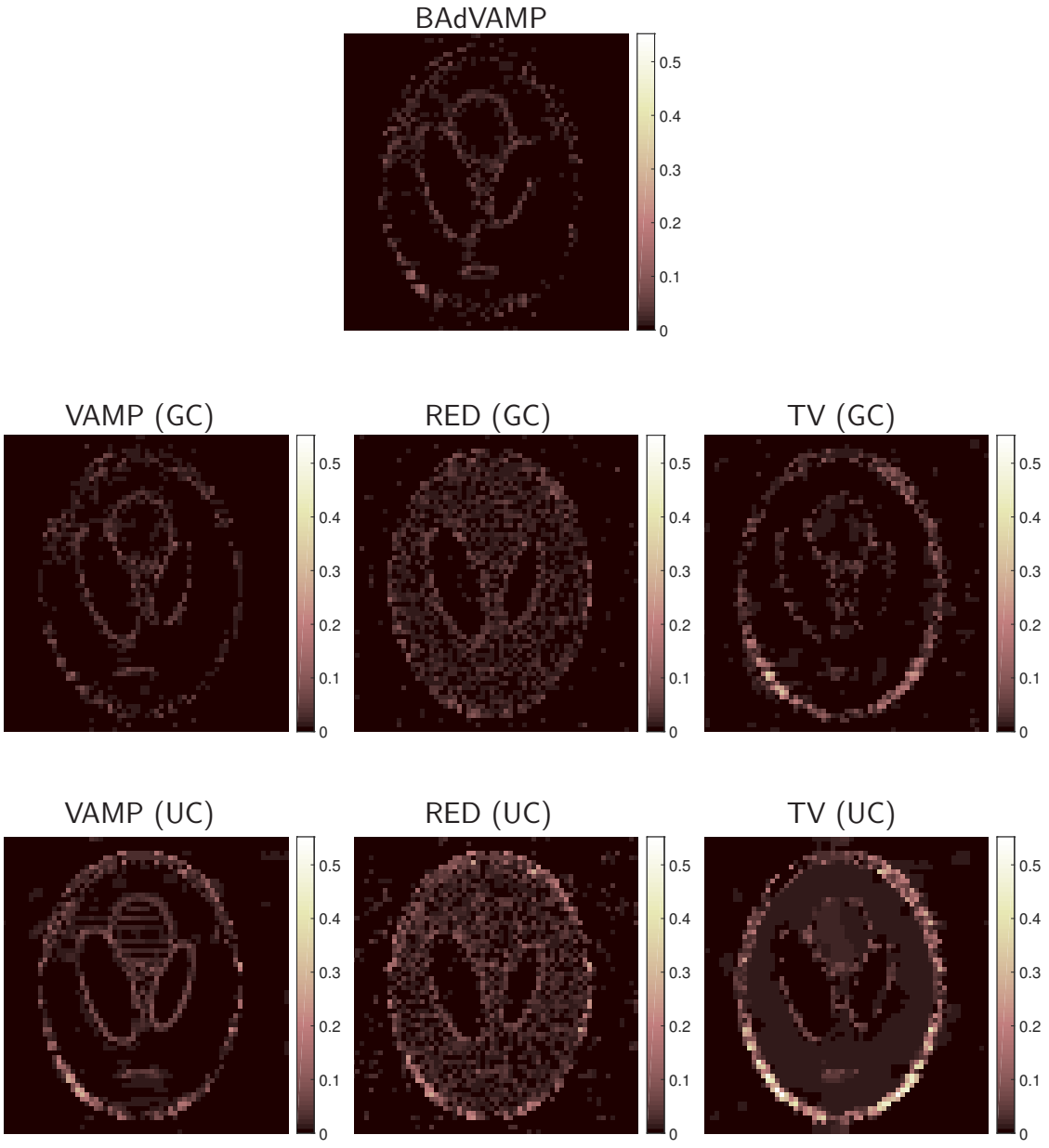

Figure 3.6: Calibration in tomography: Error images for the reconstructions shown in Fig. 3.5.

3.9.4 Noiseless Dictionary Learning

In dictionary learning (DL) [81], the goal is to find a dictionary matrix $\boldsymbol{A} \in \mathbb{R}^{M\times N}$ and a sparse matrix $\boldsymbol{X} \in \mathbb{R}^{N\times L}$ such that a given matrix $\boldsymbol{Y} \in \mathbb{R}^{M\times L}$ can be approximately factored as $\boldsymbol{Y} \approx \boldsymbol{A}\boldsymbol{X}$. In this section, we test the proposed BAd-VAMP algorithm for DL by generating $\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{X}$ such that $\boldsymbol{X}$ has K-sparse columns, and measuring the NMSE on the resulting estimates of $\boldsymbol{A}$ and $\boldsymbol{X}$.

We consider two cases: i) where the true $\boldsymbol{A}$ is structured as $\boldsymbol{A} = \sum_{i=1}^{Q} b_i \boldsymbol{A}_i$ with known $\{\boldsymbol{A}_i\}_{i=1}^{Q}$ (recall (3.62)), and ii) where the true $\boldsymbol{A}$ is unstructured (recall (3.71)). In either case, the pair $(\boldsymbol{A}, \boldsymbol{X})$ is recoverable only up to an ambiguity: a scalar ambiguity in the structured case and a generalized permutation ambiguity in the unstructured case. Thus, when measuring reconstruction quality, we consider

$$\mathrm{NMSE}(\widehat{\boldsymbol{A}}) \triangleq \min_{\lambda\in\mathbb{R}} \frac{\|\boldsymbol{A} - \lambda\widehat{\boldsymbol{A}}\|_F^2}{\|\boldsymbol{A}\|_F^2} \tag{3.89}$$

in the structured case and

$$\mathrm{NMSE}(\widehat{\boldsymbol{A}}) \triangleq \min_{\boldsymbol{P}\in\mathcal{P}} \frac{\|\boldsymbol{A} - \widehat{\boldsymbol{A}}\boldsymbol{P}\|_F^2}{\|\boldsymbol{A}\|_F^2} \tag{3.90}$$

in the unstructured case, where $\mathcal{P}$ denotes the set of generalized permutation matrices.[6] For our experiments, we drew the coefficients of $\{\boldsymbol{A}_i\}_{i=1}^{Q}$ and $\boldsymbol{b}$ as i.i.d. $\mathcal{N}(0,1)$ with $Q = N$ in the structured case, and we drew the coefficients of $\boldsymbol{A}$ as i.i.d. $\mathcal{N}(0,1)$ in the unstructured case.

In our first experiment, we fixed the sparsity rate at $K/N = 0.2$ and we varied both the dictionary dimension N and the training length L. The top-right panel of Fig. 3.7 suggests that, as the dimension N grows, a *fixed* training length L is sufficient

[6] If $\boldsymbol{P}$ is a generalized permutation matrix then $\boldsymbol{P} = \boldsymbol{\Pi}\boldsymbol{D}$, where $\boldsymbol{\Pi}$ is a permutation matrix and $\boldsymbol{D}$ is a diagonal matrix.

to successfully recover $\boldsymbol{A}$ in the structured case with $Q = N$. By "successfully recover," we mean that NMSE($\widehat{\boldsymbol{A}}$) ≤ -50 dB. Note that this latter prescription for L is consistent with the theoretical analysis in [87]. In the unstructured case, the bottom panel of Figure 3.7 shows the median NMSE($\widehat{\boldsymbol{A}}$) versys N when $L = 6N \ln N$. Together, the top-left and bottom panels of Fig. 3.7 suggest that a training length of $L = O(N \ln N)$ suffices to successfully recover $\boldsymbol{A}$.

In our second experiment, we focused on the unstructured case, fixed the training length at $L = 5N \ln N$, and varied both the dictionary dimension N and the sparsity K in the columns of $\boldsymbol{X}$. Figure 3.8 shows that BAd-VAMP performed similarly to EM-BiGAMP [68] for all but very small N, and much better than K-SVD [2] and SPAMS [54]. The advantage of BAd-VAMP over EM-BiGAMP for DL will be illustrated in the sequel.

3.9.5 Noisy, Ill-Conditioned Dictionary Learning

In this section, we show the robustness of BAd-VAMP over EM-BiGAMP [67] when learning ill-conditioned dictionaries from noisy measurements. To do so, we generated the measurements as $\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{X} + \boldsymbol{W}$ and tested the algorithms in recovering $\boldsymbol{A}$ and $\boldsymbol{X}$ (up to appropriate ambiguities). The elements of $\boldsymbol{W}$ were drawn i.i.d. $\mathcal{N}(0, 1/\gamma_w)$ with γ_w chosen to achieve SNR $\triangleq \mathbb{E}[\|\boldsymbol{A}\boldsymbol{X}\|_F^2] / \mathbb{E}[\|\boldsymbol{W}\|_F^2] = 40$ dB. The true dictionary was generated as $\boldsymbol{A} = \boldsymbol{U}\,\mathrm{Diag}(\boldsymbol{s})\boldsymbol{V}^{\mathsf{T}}$, where $\boldsymbol{U}$ and $\boldsymbol{V}$ were drawn uniformly over the group of orthogonal matrices, and where the singular values in $\boldsymbol{s}$ were chosen so that $s_i/s_{i-1} = \rho \; \forall i$. The values of s_0 and ρ were selected to obtain a desired condition number $\kappa(\boldsymbol{A})$ while also ensuring $\|\boldsymbol{A}\|_F^2 = N$.

Figure 3.9 reports median NMSE($\widehat{\boldsymbol{A}}$) and NMSE($\widehat{\boldsymbol{X}}$) versus condition number $\kappa(\boldsymbol{A})$ for the recovery of $\boldsymbol{A} \in \mathbb{R}^{N\times N}$ and K-sparse $\boldsymbol{X} \in \mathbb{R}^{N\times L}$ from noisy measurements $\boldsymbol{Y}$. For this figure, we used $K = 13$, $N = 64$, $L = 5N \ln N$, the unstructured definition of NMSE($\widehat{\boldsymbol{A}}$) from (3.90), and a similar definition for NMSE($\widehat{\boldsymbol{X}}$). In addition to showing the performance of BAd-VAMP and EM-PBiGAMP, the figure shows the performance of the known-$\boldsymbol{X}$ oracle for the estimation of $\boldsymbol{A}$, as well as the known-$\boldsymbol{A}$ and known-support oracle for the estimation of $\boldsymbol{X}$. Figure 3.9 shows that EM-BiGAMP gave near-oracle NMSE for $\kappa(\boldsymbol{A}) \leq 40$, but its performance degraded significantly for larger $\kappa(\boldsymbol{A})$. In contrast, BAd-VAMP gave near-oracle NMSE for $\kappa(\boldsymbol{A}) \leq 110$, which suggests increased robustness to ill-conditioned dictionaries $\boldsymbol{A}$.

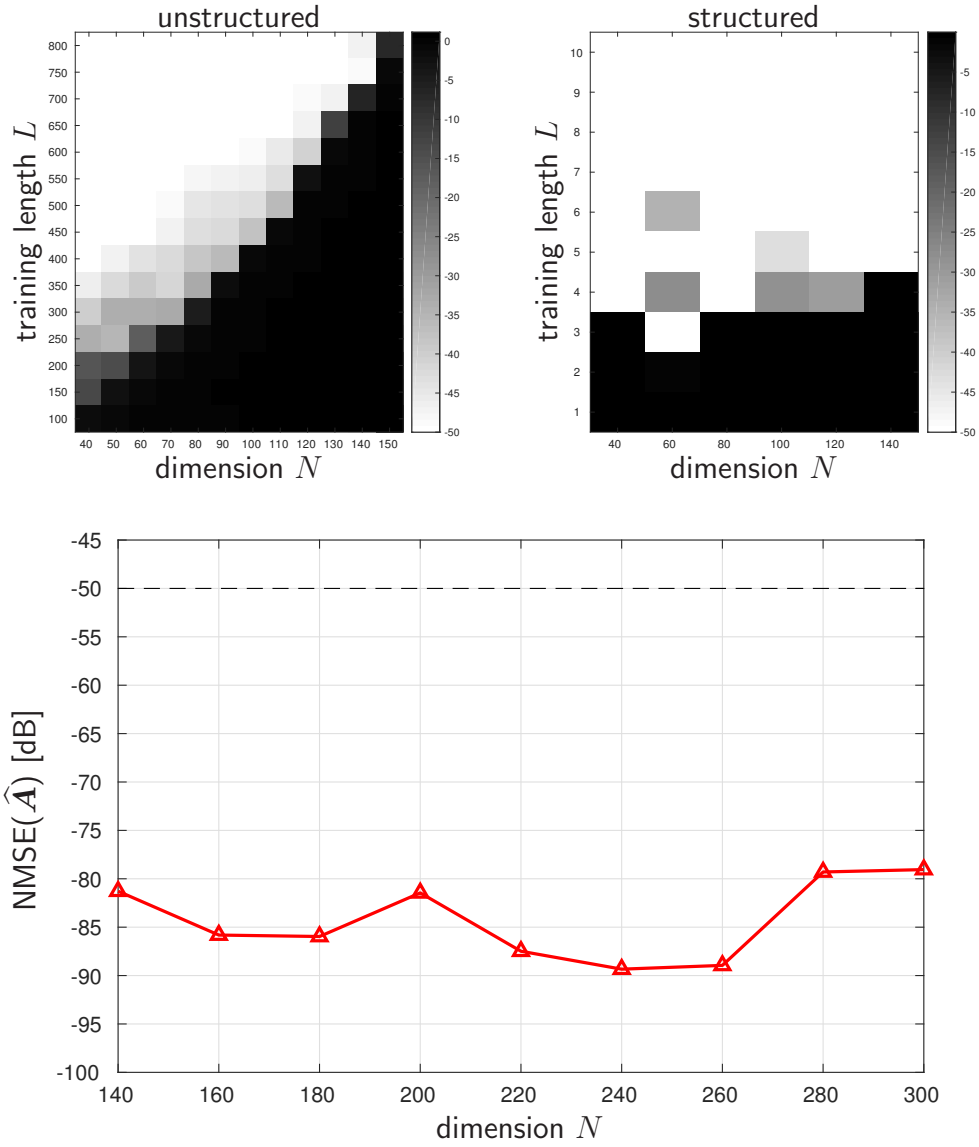

Figure 3.7: BAd-VAMP dictionary learning with $N \times N$ dictionary $\boldsymbol{A}$ and $N \times L$ code matrix $\boldsymbol{X}$ with sparsity rate $K/N = 0.2$. Top left: success-rate for unstructured dictionary versus N and L. Top right: success-rate for structured dictionary with $Q = N$ free parameters. Bottom: Median NMSE($\widehat{\boldsymbol{A}}$) over 20 trials versus N for an unstructured dictionary with $L = 6N \ln N$.

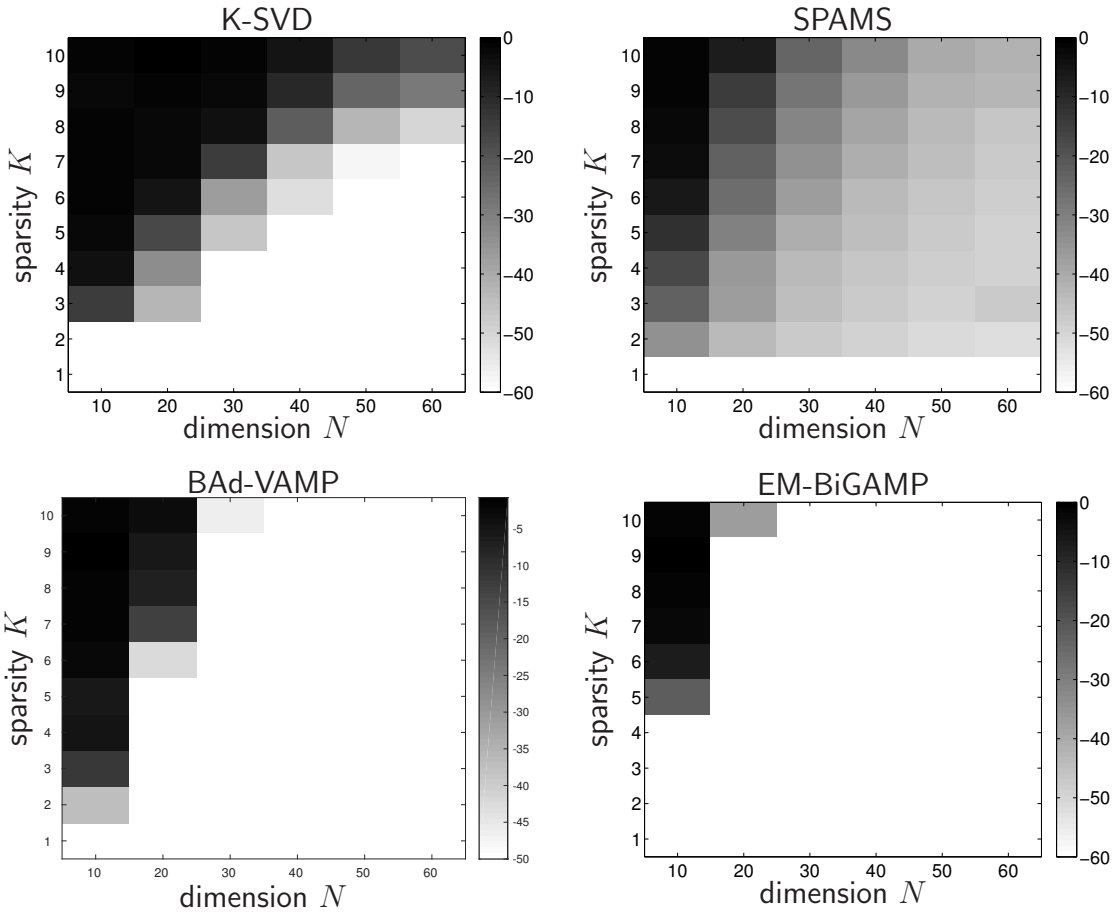

Figure 3.8: Noiseless dictionary learning: Median NMSE($\widehat{\boldsymbol{A}}$) in dB (over 20 trials) versus dictionary dimension N and sparsity K with unstructured dictionary $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ and training length $L = 5N \ln N$.

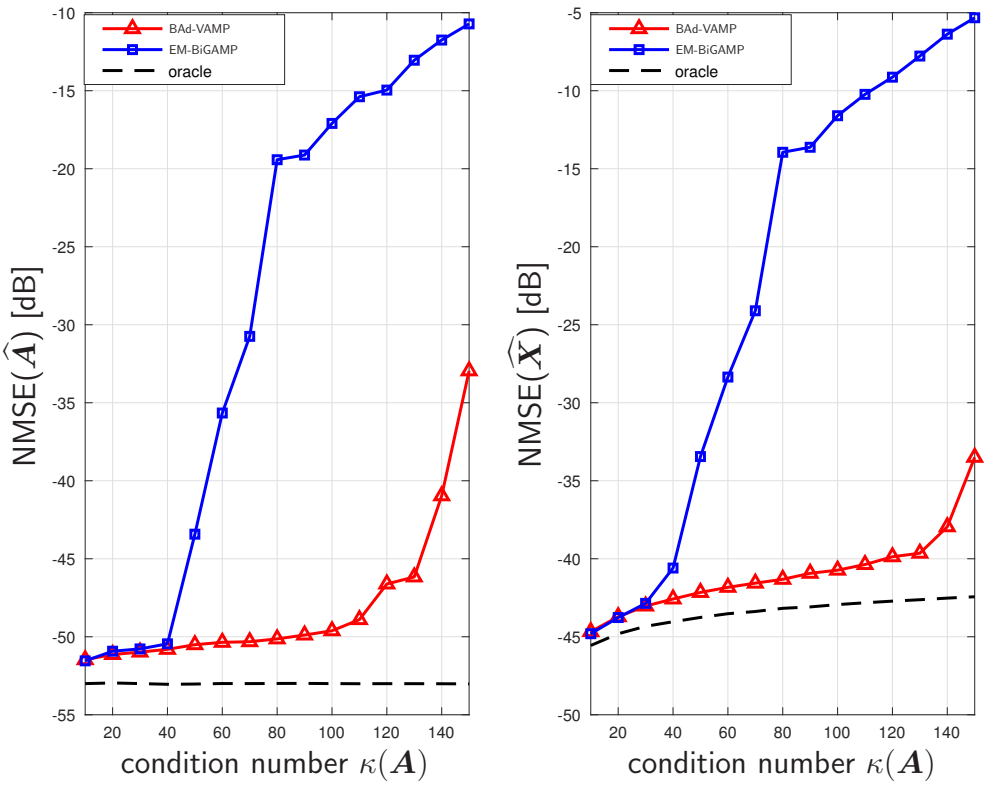

Figure 3.9: Noisy dictionary learning: Median NMSE in dB (over 50 trials) versus condition number $\kappa(\boldsymbol{A})$ for unstructured $\boldsymbol{A} \in \mathbb{R}^{N\times N}$ with $N = 64$, sparsity $K = 13$, training length $L = 5N \ln N$, and SNR $= 40$ dB.

Chapter 4: Image Reconstruction in Magnetic Resonance Imaging

4.1 Introduction

Magnetic Resonance Imaging (MRI) is a non-invasive technique used in radiology to generate internal images of the body. It uses a combination of strong static magnetic field, gradient magnetic field, and a sequence of RF pulses to capture measurements. The underlying physics of MRI is a complex quantum mechanical phenomenon. In classical physics, a static magnetic field magnetizes the atomic nucleus which causes all the nuclei to spin coherently at an angular frequency proportional to the external magnetic field. When an RF pulse of the same frequency is applied, the nuclei get excited to a higher energy state and upon relaxation emit signals measured by receiver coils in the MRI scanner. The gradient magnetic fields are used to spatially encode the signal. Since our body has an abundance of Hydrogen atoms, MRI mostly maps the location of water molecules and fat tissues in the body.

Unlike CT, which uses high energy ionizing X-rays, MRI is considered to be much safer. MRI provides better contrast between fat, water, muscle and other soft tissue structures, making it the most preferred imaging method for knee, brain, etc.

There are several challenges in MRI. Data acquisition is a very slow process, e.g., an MRI scan of the pelvic area can take more than 60 minutes and the patient has to remain still in a long narrow tube for the entire duration. Any small motion can cause undesirable artifacts in the image. For this reason, there has been a lot of research in accelerating MRI. The MRI scanner measures the image in the spatial Fourier domain, also known as the "k-space". In the ideal case, if we were to measure the entire k-space, then we could reconstruct the image exactly by taking the inverse Fourier transform of the measurements. However, in that case, the data acquisition would be extremely slow. In accelerated MRI, we acquire a relatively small number of samples in k-space and reconstruct the image from these few measurements. To increase the number of measurements without compromising the acquisition time, a popular technique is parallel MRI, in which k-space data is simultaneously acquired using multiple receiver coils. Each receiver coil has a different sensitivity map, which acts as a spatial encoding.

In this chapter, we will develop algorithms to reconstruct images in accelerated MRI using the Approximate Message Passing framework, for both single-coil and multicoil MRI. We will assume that the coil sensitivity maps are perfectly known. In practice, they can be estimated with techniques like ESPIRIT [94].

4.2 Measurement Model

With known coil maps, we can formulate MRI image reconstruction as a linear inverse problem. The measurements are noisy, subsampled, k-space data. In single-coil MRI, the measurements acquired by the coil can be mathematically modeled as

$$\boldsymbol{y} = \underbrace{\boldsymbol{SF}}_{\boldsymbol{A}}\boldsymbol{x} + \boldsymbol{w}, \quad \boldsymbol{w} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma_w) \tag{4.1a}$$

$$\Rightarrow \boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{w}, \tag{4.1b}$$

where $\boldsymbol{F}$ is the $N \times N$ 2D or 3D DFT matrix, $\boldsymbol{S} \in \mathbb{R}^{M \times N}$ is a sampling mask constructed by selecting M random rows from the $N \times N$ identity matrix, and $\boldsymbol{w} \in \mathbb{C}^M$ is AWGN measurement noise. The ratio $R \triangleq N/M$ is called the *acceleration rate*.

In parallel MRI, we have $K > 1$ receiver coils acquiring subsampled k-space data points simultaneously. Each coil applies a spatial encoding represented by a diagonal coil sensitivity matrix $\boldsymbol{C}_i$ and all the coils have the same sampling mask $\boldsymbol{S}$. The measurement $\boldsymbol{y}_i$ from the i^{th} coil can be modeled as

$$\boldsymbol{y}_i = \boldsymbol{SFC}_i\boldsymbol{x} + \boldsymbol{w}_i \tag{4.2a}$$

$$\boldsymbol{w}_i \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma_w). \tag{4.2b}$$

We then stack all the coil measurements $\{\boldsymbol{y}_i\}_{i=1}^K$ into a single column vector $\boldsymbol{y}$ which yields

$$\underbrace{\begin{bmatrix} \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_K \end{bmatrix}}_{\boldsymbol{y}} = \underbrace{\begin{bmatrix} \boldsymbol{SFC}_1 \\ \vdots \\ \boldsymbol{SFC}_K \end{bmatrix}}_{\boldsymbol{A}} \boldsymbol{x} + \underbrace{\begin{bmatrix} \boldsymbol{w}_1 \\ \vdots \\ \boldsymbol{w}_K \end{bmatrix}}_{\boldsymbol{w}} \tag{4.3a}$$

$$\Rightarrow \boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{w} \in \mathbb{C}^{MK}. \tag{4.3b}$$

The sampling mask in k-space can take several forms; random uniformly chosen points, non-uniform variable density [57], or random vertical or horizontal lines in k-space with a fully sampled central region. The uniform and variable density sampling masks make $\boldsymbol{A}$ in (4.3b) "incoherent", which is helpful for compressive sensing

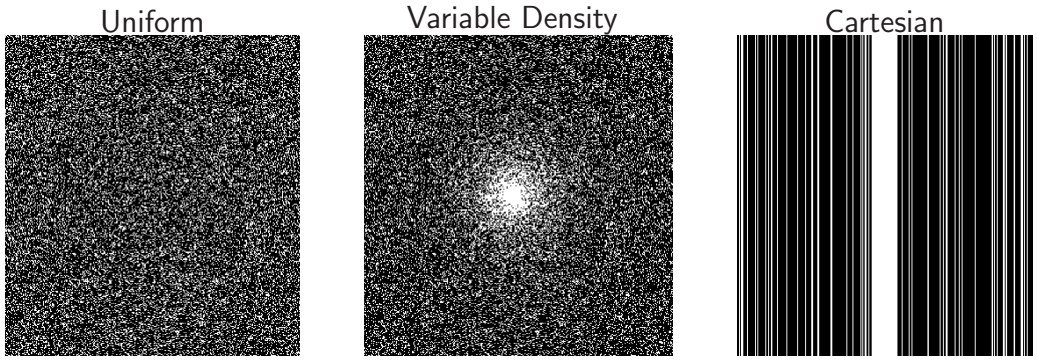

Figure 4.1: Example sampling masks for acceleration rate $R = 4$

based methods. Taking vertical lines in k-space sampling is called "Cartesian sampling" and is the most dominant method in clinical practice, mainly due its robustness to various system defects.

4.3 Signal Denoiser

From (4.1) and (4.3b), we see that image reconstruction in MRI is a linear inverse problem. Figure 4.2 shows that for high acceleration rates, the matrix $\boldsymbol{A}$ is ill-conditioned, so using prior knowledge of MR images will lead to better reconstruction performance. If p_{x} is a prior density over $\boldsymbol{x}$, the *maximum a posteriori* (MAP) estimate of $\boldsymbol{x}$ is

$$\widehat{\boldsymbol{x}}_{\mathsf{MAP}} \triangleq \arg\min_{\boldsymbol{x}} -\ln p_{\mathsf{x}|\mathsf{y}}(\boldsymbol{x}|\boldsymbol{y}) \tag{4.4a}$$

$$= \arg\min_{\boldsymbol{x}} -\ln p_{\mathsf{y}|\mathsf{x}}(\boldsymbol{y}|\boldsymbol{x}) - \ln p_{\mathsf{x}}(\boldsymbol{x}). \tag{4.4b}$$

The likelihood function function from (4.3b) is

$$p_{\mathsf{y}|\mathsf{x}}(\boldsymbol{y}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{y}; \boldsymbol{A}\boldsymbol{x}, \boldsymbol{I}/\gamma_w), \tag{4.5}$$

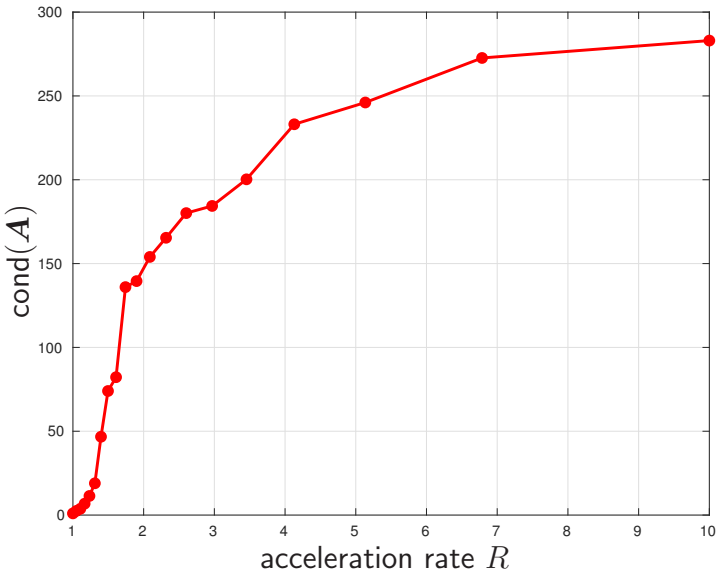

Figure 4.2: Condition number of $\boldsymbol{A}$ vs acceleration rate R in multicoil MRI for Cartesian sampling and 4 receiver coils.

in which case Equation (4.4b) simplifies to

$$\widehat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \frac{\gamma_w}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2 + \lambda\rho(\boldsymbol{x}), \tag{4.6}$$

where $\rho(\boldsymbol{x}) \triangleq -\frac{1}{\lambda}\ln p_{\mathbf{x}}(\boldsymbol{x})$ and $\lambda > 0$. In CS based reconstruction methods [53], sparsity of transform coefficients for some transform $\boldsymbol{\Phi}^{\mathsf{H}}$, e.g. a wavelet transform, is used as the prior, in which case $\rho(\cdot)$ typically takes the form

$$\rho(\boldsymbol{x}) = \|\boldsymbol{\Phi}^{\mathsf{H}}\boldsymbol{x}\|_1. \tag{4.7}$$

The ℓ_1 norm yields a convex regularization that encourages solutions with sparse transform coefficients.

As we increase the acceleration rate, we need more powerful priors to accurately reconstruct the MR image. Designing a prior distribution $p_{\mathbf{x}}$ for images is difficult in practice. However, most iterative optimization algorithms don't require an explicit

knowledge of p_{x}. Rather, they use a signal denoiser, which implicitly uses some prior p_{x} and which is much easier to design in practice. A Gaussian denoiser is defined as a function that recovers $\boldsymbol{x}$ from its AWGN corrupted version $\boldsymbol{r}$:

$$\boldsymbol{r} = \boldsymbol{x} + \boldsymbol{q}, \quad \boldsymbol{q} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma). \tag{4.8}$$

There have been many papers on image denoising, e.g., [15, 26, 52, 92, 104], some of which solve a MAP optimization problem of the form (4.6) with $\boldsymbol{A} = \boldsymbol{I}$, whereas some are algorithmic [26, 52, 92] or deep learning based [52, 104]. In the next section, we will describe in detail how a denoiser is used to solve linear inverse problems[7].

4.4 Plug-and-Play Algorithms

To solve convex problems of the form (4.6), a commonly used algorithm is alternating direction method of multipliers (ADMM) [14] which converts (4.6) into a constrained optimization problem using the variable splitting technique,

$$\min_{\boldsymbol{x},\boldsymbol{v}} \frac{\gamma_w}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2 + \rho(\boldsymbol{v}) \tag{4.9a}$$

$$\text{subject to} \;\; \boldsymbol{x} = \boldsymbol{v}. \tag{4.9b}$$

and solves it using the augmented Lagrangian

$$\min_{\boldsymbol{x},\boldsymbol{v}} \max_{\boldsymbol{u}} \Big\{ \frac{\gamma_w}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2 + \lambda\rho(\boldsymbol{v}) + \mathrm{Re}\{\boldsymbol{u}^{\mathsf{H}}(\boldsymbol{x} - \boldsymbol{v})\} + \frac{c}{2}\|\boldsymbol{x} - \boldsymbol{v}\|^2 \Big\}, \tag{4.10}$$

[7] We saw an example of this earlier in Sec. 2.5

where $\boldsymbol{u}$ is a Lagrange multiplier and $c > 0$ is a tuning parameter that does not affect the fixed point. ADMM solves the following sequence of sub-problems at iteration t,

$$\boldsymbol{x}^{t+1} = \arg\min_{\boldsymbol{x}} \frac{\gamma_w}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2 + \frac{c}{2}\|\boldsymbol{x} - \boldsymbol{v}^t + \widetilde{\boldsymbol{u}}^t\|^2 \tag{4.11a}$$

$$\boldsymbol{v}^{t+1} = \arg\min_{\boldsymbol{v}} \lambda\rho(\boldsymbol{v}) + \frac{c}{2}\|\boldsymbol{v} - \boldsymbol{x}^{t+1} - \widetilde{\boldsymbol{u}}^t\|^2 \tag{4.11b}$$

$$\widetilde{\boldsymbol{u}}^{t+1} = \widetilde{\boldsymbol{u}}^t + \left(\boldsymbol{x}^{t+1} - \boldsymbol{v}^{t+1}\right). \tag{4.11c}$$

where $\widetilde{\boldsymbol{u}}^t \triangleq (1/c)\boldsymbol{u}^t$. Under mild conditions on convex ρ, it can be shown that ADMM converges to the solution of (4.6). Using the definition of the proximal operator

$$\text{prox}_{\phi}(\boldsymbol{r}) \triangleq \arg\min_{\boldsymbol{x}} \phi(\boldsymbol{x}) + \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{r}\|^2, \tag{4.12}$$

we can rewrite the ADMM iterations in (4.11) as

$$\boldsymbol{x}^{t+1} = \text{prox}_{\frac{\gamma_w}{c} f}(\boldsymbol{v}^t - \widetilde{\boldsymbol{u}}^t) \tag{4.13a}$$

$$\boldsymbol{v}^{t+1} = \text{prox}_{\frac{\lambda}{c}\rho}(\boldsymbol{x}^{t+1} + \widetilde{\boldsymbol{u}}^t) \tag{4.13b}$$

$$\widetilde{\boldsymbol{u}}^{t+1} = \widetilde{\boldsymbol{u}}^t + \left(\boldsymbol{x}^{t+1} - \boldsymbol{v}^{t+1}\right), \tag{4.13c}$$

where $f(\boldsymbol{x}) \triangleq \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2$ is the data fidelity term.

We can interpret the proximal operator $\text{prox}_{\frac{\lambda}{c}\rho}$ as a Gaussian denoiser. For example, it is the MAP estimator of $\boldsymbol{x}$ from the AWGN corrupted measurement $\boldsymbol{r} = \boldsymbol{x} + \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma)$ with prior density $p_{\mathbf{x}}(\boldsymbol{x}) \propto e^{-\lambda\rho(\boldsymbol{x})}$. Using this interpretation, Bouman et. al. [95] suggested to replace (4.13b) with any suitable image denoiser, $\boldsymbol{g}(\boldsymbol{r}, \gamma)$. In this way, ADMM can be used with denoisers that are neither optimization based nor have any statistical interpretation, e.g., block-matching and 3D filtering (BM3D) [26], non-local means (NLM) [15], denoising convolutional neural network (DnCNN) [104], etc.

The variable splitting technique in ADMM decouples the prior image knowledge from data consistency. It solves two subproblems at every iteration; (4.13a) enforcing data consistency and (4.13b) enforcing the prior knowledge. Many other iterative solvers for linear inverse problems also decouple the two stages using some variable splitting method. Though the authors in [95] originally used ADMM, their approach can be easily generalized to other iterative algorithms, which led to the development of so called *plug-and-play* (PnP) algorithms. PnP-FISTA [44], RED [76, 79], PnP-PDS [62] are examples of some other popular PnP algorithms.

Another advantage of any PnP algorithm is that the denoiser is not tied to any particular choice of $\boldsymbol{A}$. This is very beneficial in computational imaging, where the forward operator $\boldsymbol{A}$ depends on the application. We use this PnP approach to develop our reconstruction algorithm for MRI using the AMP framework.

4.5 Our Approach

In most PnP algorithms, the denoiser input variance (often adjustable) is fixed over all iterations. But the denoiser variance affects their fixed point and convergence speed [25]. AMP based algorithms don't have this issue, because the denoiser input noise variance is automatically estimated by the algorithm at every iteration, leading to fast convergence and accurate solutions with large random $\boldsymbol{A}$. However, AMP based algorithms are not designed to use the structured $\boldsymbol{A}$ typically used in MRI. Thus, their noise-variance estimates may not be accurate and the algorithm may even diverge.

Some existing AMP based algorithms for MRI are Variable Density AMP (VDAMP) [57], which uses soft-thresholding of the wavelet coefficients for denoising, and BM3D-AMP-MRI [33] which uses a modified version of the BM3D [26] denoiser to handle complex-valued images. VDAMP neither supports multicoil MRI nor Cartesian sampling mask and BM3D-AMP-MRI is for single-coil MRI. To the best of our knowledge, we are the first to propose an AMP based algorithm using a generic denoiser for image reconstruction in multicoil MRI using generic k-space sampling masks.

4.5.1 AMP for MRI

In this section, we discuss problems with the conventional AMP algorithm [32] for MRI and propose steps to rectify them. AMP recovers $\boldsymbol{x} \in \mathbb{C}^N$ from measurements $\boldsymbol{y} \in \mathbb{C}^M$ in (4.6) using the following iterations

$$\boldsymbol{v}^t = \boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}^t + \frac{N}{M}\boldsymbol{v}^{t-1}\left\langle \boldsymbol{g}^{'t}(\boldsymbol{r}^{t-1})\right\rangle \tag{4.14a}$$

$$\boldsymbol{r}^t = \boldsymbol{x}^t + \beta\boldsymbol{A}^{\mathsf{H}}\boldsymbol{v}^t \tag{4.14b}$$

$$\boldsymbol{x}^{t+1} = \boldsymbol{g}^t(\boldsymbol{r}^t) \tag{4.14c}$$

where $\boldsymbol{v}^{-1} = \boldsymbol{0}$ and for i.i.d. Gaussian zero-mean $\boldsymbol{A}$ the scalar β is set to $\frac{N}{\|\boldsymbol{A}\|_F^2}$. We later describe the effect of β on AMP. The denoiser $\boldsymbol{g}^t$ may depend on iteration t, and its divergence $\left\langle \boldsymbol{g}^{'t}(\boldsymbol{r})\right\rangle$ is the average of diagonal elements of the Jacobian of $\boldsymbol{g}^t$, that can be approximated using the Monte-Carlo method [70]

$$\left\langle \boldsymbol{g}^{'t}(\boldsymbol{r})\right\rangle \approx \frac{1}{N\epsilon}\boldsymbol{q}^{\mathsf{H}}\left[\boldsymbol{g}^t(\boldsymbol{r}+\epsilon\boldsymbol{q}) - \boldsymbol{g}^t(\boldsymbol{r})\right], \tag{4.15}$$

using small $\epsilon > 0$ and random $\boldsymbol{q}$ such that $\mathbb{E}[\boldsymbol{q}\boldsymbol{q}^{\mathsf{H}}] = \boldsymbol{I}$.

The rightmost term in (4.14a) is known as the *Onsager correction term* from statistical physics, which ensures that for zero-mean i.i.d. Gaussian $\boldsymbol{A}$, as the dimensions M, N grow to infinity (fixed M/N), the denoiser input error satisfies the AWGN property,

$$\boldsymbol{r}^t = \boldsymbol{x}_0 + \boldsymbol{e}^t, \quad \boldsymbol{e}^t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}/\gamma^t). \tag{4.16}$$

Additionally, the errors $\{\boldsymbol{e}^t\}$ are independent across iterations t. See [9] for details.

The version of AMP in (4.14) is usually seen in the literature and it expects $\|\boldsymbol{A}\|_F^2 \approx N$. When it is not the case, the scalar β in (4.14) compensates for the mismatch. To understand how this happens, let $\boldsymbol{A}$ be drawn from i.i.d. $\mathcal{N}(0, \sigma_a^2)$ and $\boldsymbol{x}_0$ be the true vector, and consider the following error terms in AMP:

$$\boldsymbol{e}^t \triangleq \boldsymbol{r}^t - \boldsymbol{x}_0 \quad \text{(denoiser input error)} \tag{4.17a}$$

$$\boldsymbol{\epsilon}^t \triangleq \boldsymbol{x}^t - \boldsymbol{x}_0 \quad \text{(denoiser output error)}, \tag{4.17b}$$

which evolve according to

$$\boldsymbol{\epsilon}^{t+1} = \boldsymbol{g}^t(\boldsymbol{x}_0 + \boldsymbol{e}^t) - \boldsymbol{x}_0 \tag{4.18a}$$

$$\boldsymbol{e}^{t+1} = (\boldsymbol{I} - \beta \boldsymbol{A}^{\mathsf{H}}\boldsymbol{A})\boldsymbol{\epsilon}^{t+1} + \beta \frac{N}{M} \boldsymbol{A}^{\mathsf{H}} \boldsymbol{v}^t \left\langle \boldsymbol{g}'^t(\boldsymbol{x}_0 + \boldsymbol{e}^t) \right\rangle + \beta \boldsymbol{A}^{\mathsf{H}} \boldsymbol{w}. \tag{4.18b}$$

In the first iteration, $t = 0$, since we initialize using $\boldsymbol{v}^{-1} = 0$, we see that the denoiser input error is

$$\boldsymbol{e}^0 = \left(\boldsymbol{I} - \beta \boldsymbol{A}^{\mathsf{H}}\boldsymbol{A}\right) \boldsymbol{\epsilon}^0 + \beta \boldsymbol{A}^{\mathsf{H}} \boldsymbol{w}. \tag{4.19}$$

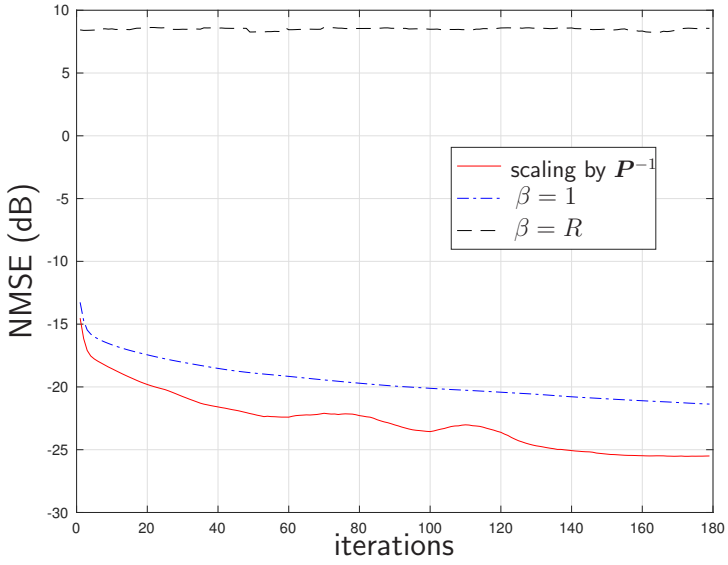

Figure 4.3: NMSE (dB) of AMP in MRI for different type of scaling in k-space

If we assume $\boldsymbol{\epsilon}^0$ is independent of $\{\boldsymbol{A}, \boldsymbol{w}\}$ and we take the expectation conditioned on $\boldsymbol{\epsilon}^0$, then we get

$$\mathbb{E}[\boldsymbol{e}^0 \mid \boldsymbol{\epsilon}^0] = \mathbb{E}\left[\left(\boldsymbol{I} - \beta \boldsymbol{A}^{\mathsf{H}}\boldsymbol{A}\right)\boldsymbol{\epsilon}^0 \mid \boldsymbol{\epsilon}^0\right] + \beta \underbrace{\mathbb{E}\left[\boldsymbol{A}^{\mathsf{H}}\boldsymbol{w}\right]}_{\mathbf{0}} \tag{4.20a}$$

$$\overset{\text{(a)}}{=} \mathbb{E}\left[\left(\boldsymbol{I} - \frac{1}{\sigma_a^2 M}\boldsymbol{A}^{\mathsf{H}}\boldsymbol{A}\right)\right]\boldsymbol{\epsilon}^0 \tag{4.20b}$$

$$= \left(\boldsymbol{I} - \frac{1}{\sigma_a^2 M}\sigma_a^2 M \boldsymbol{I}\right)\boldsymbol{\epsilon}^0 \tag{4.20c}$$

$$= \mathbf{0}, \tag{4.20d}$$

where in (a) we used $\beta = \frac{N}{\|\boldsymbol{A}\|_F^2} = \frac{1}{\sigma_a^2 M}$. Equation (4.20) shows that this choice of the scalar β is necessary for the denoiser input to be an unbiased estimate of $\boldsymbol{x}_0$.

In MRI, $\boldsymbol{A}$ is not i.i.d. Gaussian, so if we run AMP with $\beta = \frac{N}{\|\boldsymbol{A}\|_F^2} = R \triangleq \frac{N}{M}$, it doesn't work well and often diverges. An existing method BM3D-AMP-MRI [33] sets $\beta = 1$, which empirically helps to stabilize AMP but its final solution is not quite accurate. For single-coil knee MRI, Cartesian sampling of $R = 4$ and SNR=40 dB,

Fig. 4.3 shows the NMSE of AMP for $\beta = 1$, $\beta = R$, and another type of scaling in k-space that we propose next.

Scaling in k-space

It is not clear how to achieve (4.16) in MRI, but if one scales the k-space measurements in a certain way, one can still have the unbiased property satisfied. We are going to assume that the coil sensitivities satisfy:

$$\sum_{i=1}^{K} \boldsymbol{C}_i^{\mathsf{H}} \boldsymbol{C}_i = \boldsymbol{I}. \tag{4.21}$$

Coil sensitivities estimated by ESPIRIT [94] guarantees the above scaling.

Consider generating the sampling mask $\boldsymbol{S}$ by randomly selecting rows from the $N \times N$ identity matrix such that the probability of selecting the i^{th} row is p_i, where $p_i > 0 \; \forall i$. Let $\boldsymbol{P} \in \mathbb{R}^{N \times N}$ be a diagonal matrix with the probabilities $\{p_i\}_{i=1}^{N}$ along its diagonal, i.e., $\boldsymbol{P} \triangleq \text{Diag}(\boldsymbol{p})$. In the conventional AMP iterations, instead of scaling the k-space residual $\boldsymbol{v}^t$ in line (4.14b) by the scalar β, we propose to scale it by $\boldsymbol{S}\boldsymbol{P}^{-1}\boldsymbol{S}^{\mathsf{T}}$. In other words, we replace line (4.14b) by

$$\boldsymbol{r}^t = \boldsymbol{x}^t + \boldsymbol{A}^{\mathsf{H}}(\boldsymbol{S}\boldsymbol{P}^{-1}\boldsymbol{S}^{\mathsf{T}}\boldsymbol{v}^t). \tag{4.22}$$

In this case, at iteration $t = 0$, the denoiser input error becomes

$$\boldsymbol{e}^0 = \left(\boldsymbol{I} - \boldsymbol{A}^{\mathsf{H}}\boldsymbol{S}\boldsymbol{P}^{-1}\boldsymbol{S}^{\mathsf{T}}\boldsymbol{A}\right)\boldsymbol{\epsilon}^0 + \boldsymbol{A}^{\mathsf{H}}\boldsymbol{S}\boldsymbol{P}^{-1}\boldsymbol{S}^{\mathsf{T}}\boldsymbol{w}. \tag{4.23}$$

And the expected error is

$$\mathbb{E}[\boldsymbol{e}^0 \mid \boldsymbol{\epsilon}^0] = \mathbb{E}\left[\boldsymbol{I} - \boldsymbol{A}^{\mathsf{H}}\boldsymbol{S}\boldsymbol{P}^{-1}\boldsymbol{S}^{\mathsf{T}}\boldsymbol{A}\right]\boldsymbol{\epsilon}^0 + \mathbb{E}\left[\boldsymbol{A}^{\mathsf{H}}\boldsymbol{S}\boldsymbol{P}^{-1}\boldsymbol{S}^{\mathsf{T}}\boldsymbol{w}\right] \tag{4.24a}$$

$$= \mathbb{E}\left[\boldsymbol{I} - \sum_{i=1}^{K}\boldsymbol{C}_i^{\mathsf{H}}\boldsymbol{F}^{\mathsf{H}}\boldsymbol{S}^{\mathsf{T}}\boldsymbol{S}\boldsymbol{P}^{-1}\boldsymbol{S}^{\mathsf{T}}\boldsymbol{S}\boldsymbol{F}\boldsymbol{C}_i\right]\boldsymbol{\epsilon}^0 + \mathbb{E}\left[\boldsymbol{A}^{\mathsf{H}}\boldsymbol{S}\boldsymbol{P}^{-1}\boldsymbol{S}^{\mathsf{T}}\right]\underbrace{\mathbb{E}[\boldsymbol{w}]}_{\mathbf{0}} \tag{4.24b}$$

$$= \left(\boldsymbol{I} - \sum_{i=1}^{K}\boldsymbol{C}_i^{\mathsf{H}}\boldsymbol{F}^{\mathsf{H}}\,\mathbb{E}\left[\boldsymbol{S}^{\mathsf{T}}\boldsymbol{S}\boldsymbol{P}^{-1}\boldsymbol{S}^{\mathsf{T}}\boldsymbol{S}\right]\boldsymbol{F}\boldsymbol{C}_i\right)\boldsymbol{\epsilon}^0 \tag{4.24c}$$

$$= \left(\boldsymbol{I} - \sum_{i=1}^{K}\boldsymbol{C}_i^{\mathsf{H}}\boldsymbol{F}^{\mathsf{H}}\,\mathbb{E}\left[\overline{\boldsymbol{S}}\boldsymbol{P}^{-1}\overline{\boldsymbol{S}}\right]\boldsymbol{F}\boldsymbol{C}_i\right)\boldsymbol{\epsilon}^0 \quad \text{where} \quad \overline{\boldsymbol{S}} \triangleq \boldsymbol{S}^{\mathsf{T}}\boldsymbol{S} \tag{4.24d}$$

$$= \left(\boldsymbol{I} - \sum_{i=1}^{K}\boldsymbol{C}_i^{\mathsf{H}}\boldsymbol{F}^{\mathsf{H}}\,\mathbb{E}\left[\overline{\boldsymbol{S}}\right]\boldsymbol{P}^{-1}\boldsymbol{F}\boldsymbol{C}_i\right)\boldsymbol{\epsilon}^0 \tag{4.24e}$$

$$\overset{\text{(a)}}{=} \left(\boldsymbol{I} - \sum_{i=1}^{K}\boldsymbol{C}_i^{\mathsf{H}}\boldsymbol{F}^{\mathsf{H}}\boldsymbol{P}\boldsymbol{P}^{-1}\boldsymbol{F}\boldsymbol{C}_i\right)\boldsymbol{\epsilon}^0 \tag{4.24f}$$

$$= \left(\boldsymbol{I} - \sum_{i=1}^{K}\boldsymbol{C}_i^{\mathsf{H}}\boldsymbol{C}_i\right)\boldsymbol{\epsilon}^0 \tag{4.24g}$$

$$\overset{\text{(b)}}{=} \mathbf{0}, \tag{4.24h}$$

where (a) follows from $\mathbb{E}\left[\overline{\boldsymbol{S}}\right] = \boldsymbol{P}$ and (b) follows from the property (4.21). Our proposed algorithm, Debiased Denoising AMP (DD-AMP) is summarized in Algorithm 6.

Another way to derive Alg. 6 is to left multiply (4.1) by $\boldsymbol{P}^{-\frac{1}{2}}\boldsymbol{S}^{\mathsf{T}}$, in which case we get the following equivalent measurement model,

$$\widetilde{\boldsymbol{y}} = \widetilde{\boldsymbol{A}}\boldsymbol{x} + \widetilde{\boldsymbol{w}}, \tag{4.25}$$

Algorithm 6 DD-AMP (Debiased Denoising AMP) for MRI

1: **define:**
   $\boldsymbol{B} = \boldsymbol{S}\boldsymbol{P}^{-1}\boldsymbol{S}^{\mathsf{T}}$
2: **initialize:**
   $\boldsymbol{x}^0, \boldsymbol{v}^{-1} = \boldsymbol{0}$
3: **for** $t = 0, \dots, T_{\max}$ **do**
4: $\quad \boldsymbol{v}^t = \boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}^t + \frac{N}{M}\boldsymbol{v}^{t-1}\left\langle \boldsymbol{g}'^t(\boldsymbol{r}^{t-1})\right\rangle$
5: $\quad \boldsymbol{r}^t = \boldsymbol{x}^t + \boldsymbol{A}^{\mathsf{H}}\boldsymbol{B}\boldsymbol{v}^t$
6: $\quad \boldsymbol{x}^{t+1} = \boldsymbol{g}^t(\boldsymbol{r}^t)$
7: **end for**

where $\widetilde{\boldsymbol{y}} \triangleq \boldsymbol{P}^{-\frac{1}{2}}\boldsymbol{S}^{\mathsf{T}}\boldsymbol{y}$, $\widetilde{\boldsymbol{A}} \triangleq \boldsymbol{P}^{-\frac{1}{2}}\boldsymbol{S}^{\mathsf{T}}\boldsymbol{A}$ and $\widetilde{\boldsymbol{w}} \triangleq \boldsymbol{P}^{-\frac{1}{2}}\boldsymbol{S}^{\mathsf{T}}\boldsymbol{w}$. Taking expectation over the random quantities $\{\boldsymbol{S}, \boldsymbol{w}\}$, the noise in (4.25) is additive white noise,

$$\mathbb{E}\left[\widetilde{\boldsymbol{w}}\widetilde{\boldsymbol{w}}^{\mathsf{H}}\right] = \mathbb{E}\left[\boldsymbol{P}^{-\frac{1}{2}}\boldsymbol{S}^{\mathsf{T}}\boldsymbol{w}\boldsymbol{w}^{\mathsf{H}}\boldsymbol{S}\boldsymbol{P}^{-\frac{1}{2}}\right] \tag{4.26a}$$

$$= \mathbb{E}\left[\boldsymbol{P}^{-\frac{1}{2}}\boldsymbol{S}^{\mathsf{T}}\,\mathbb{E}[\boldsymbol{w}\boldsymbol{w}^{\mathsf{H}}]\boldsymbol{S}\boldsymbol{P}^{-\frac{1}{2}}\right] \tag{4.26b}$$

$$= \frac{1}{\gamma_w}\,\mathbb{E}\left[\boldsymbol{P}^{-\frac{1}{2}}\boldsymbol{S}^{\mathsf{T}}\boldsymbol{S}\boldsymbol{P}^{-\frac{1}{2}}\right] \tag{4.26c}$$

$$= \frac{1}{\gamma_w}\,\mathbb{E}\left[\boldsymbol{P}^{-\frac{1}{2}}\overline{\boldsymbol{S}}\boldsymbol{P}^{-\frac{1}{2}}\right], \quad \text{where } \overline{\boldsymbol{S}} \triangleq \boldsymbol{S}^{\mathsf{T}}\boldsymbol{S} \tag{4.26d}$$

$$= \frac{1}{\gamma_w}\,\mathbb{E}\left[\overline{\boldsymbol{S}}\boldsymbol{P}^{-1}\right] \tag{4.26e}$$

$$= \boldsymbol{I}/\gamma_w. \tag{4.26f}$$

If we run the traditional AMP algorithm (4.14) using $\beta = 1$ on the scaled measurement model (4.25), we get Algorithm 6 after a variable change.

VDAMP (Alg. 7) [57] also uses a scaling by $\boldsymbol{P}^{-1}$, but they do not apply it to the AMP or VAMP algorithms. They recover the wavelet coefficients of the image. In Alg. 7, $\boldsymbol{g}_{\text{soft-thresh}}$ is the soft-thresholding function, and $\odot$, $\oslash$, denote element-wise product and element-wise division respectively. The operator $\mathcal{B}$ performs block-wise averaging, where one block is a wavelet band. In Alg. 7, line 5 is a linear estimator similar to AMP, whereas line 12 is an Onsager correction step similar to VAMP. Their

Algorithm 7 VDAMP from [57]

define:
  $\boldsymbol{\Psi}$: wavelet transform
  $\boldsymbol{H} = |\boldsymbol{F}\boldsymbol{\Psi}^{\mathsf{H}}|^2$ (element-wise squared)
  $\mathcal{B}, \mathcal{B}^*$: block averaging operator, and its adjoint
initialize:
  $\widetilde{\boldsymbol{r}}^0$
for $t = 0, \ldots, T_{\max}$ **do**
  $\boldsymbol{z}^t = \boldsymbol{y} - \boldsymbol{S}\boldsymbol{F}\boldsymbol{\Psi}^{\mathsf{H}}\widetilde{\boldsymbol{r}}^t$
  $\boldsymbol{r}^t = \widetilde{\boldsymbol{r}}^t + \boldsymbol{\Psi}\boldsymbol{F}^{\mathsf{H}}\boldsymbol{P}^{-1}\boldsymbol{S}^{\mathsf{T}}\boldsymbol{z}^t$
  $\boldsymbol{\tau}^t = \boldsymbol{H}^{\mathsf{T}}\boldsymbol{S}^{\mathsf{T}}\boldsymbol{S}\boldsymbol{P}^{-1}\left[\left(\boldsymbol{P}^{-1} - \boldsymbol{I}\right)\boldsymbol{S}^{\mathsf{T}}|\boldsymbol{z}^t|^2 + \mathbf{1}/\gamma_w\right]$
  $\overline{\boldsymbol{\tau}}^t = \mathcal{B}(\boldsymbol{\tau}^t)$
  $\boldsymbol{w}^t = \boldsymbol{g}_{\mathsf{soft\text{-}thresh}}(\boldsymbol{r}^t, \overline{\boldsymbol{\tau}}^t)$
  $\overline{\boldsymbol{\alpha}}^t = \mathcal{B}\left(\operatorname{diag}\left\{\partial \boldsymbol{g}'_{\mathsf{soft\text{-}thresh}}(\boldsymbol{r}^t, \overline{\boldsymbol{\tau}}^t)/\partial \boldsymbol{r}\right\}\right)$
  $\boldsymbol{\alpha}^t = \mathcal{B}^*(\overline{\boldsymbol{\alpha}}^t)$
  $\boldsymbol{c}^t = \mathbf{1} \oslash (\mathbf{1} - \boldsymbol{\alpha}^t)$
  $\widetilde{\boldsymbol{r}}^t = \boldsymbol{c}^t \odot (\boldsymbol{w}^t - \boldsymbol{\alpha}^t \odot \boldsymbol{r}^t)$
  $\boldsymbol{x}^t = \boldsymbol{\Psi}^{\mathsf{H}}\boldsymbol{w}^t + \boldsymbol{F}^{\mathsf{H}}\boldsymbol{S}^{\mathsf{T}}\left(\boldsymbol{y} - \boldsymbol{S}\boldsymbol{F}\boldsymbol{\Psi}^{\mathsf{H}}\boldsymbol{w}^t\right)$
end for

algorithm is a hybrid of AMP and VAMP, and does not apply to the multicoil case. We show in numerical experiments that our approach achieves superior results even for the single-coil case.

4.5.2 VAMP for MRI

VAMP has provable convergence guarantees when $\boldsymbol{A}$ is right rotationally invariant (RRI) and the denoiser $\boldsymbol{g}_1$ is separable [73] and non-separable (chapter 2). For arbitrary $\boldsymbol{A}$ and non-expansive denoiser $\boldsymbol{g}_1$, VAMP is known to converge with appropriate damping [36]. In MRI, the RRI property does not hold, and empirically damping helps but we have no proof.

A damping scheme was proposed in [36] under which VAMP is guaranteed to converge with any non-expansive denoiser. They fixed the precisions in VAMP $(\gamma_1^t, \gamma_2^t, \eta_1^t, \eta_2^t) = (\gamma_1, \gamma_2, \gamma_1+\gamma_2, \gamma_1+\gamma_2)$ for arbitrary γ_1,$\gamma_2 > 0$, and then the damping factor ζ was calculated such that each iteration of damped VAMP was a contraction mapping. The expression for ζ is

$$\zeta = \frac{2}{1 + \max\left\{\frac{\gamma_1}{\gamma_2}, \frac{\gamma_2}{\gamma_1}\right\}}. \tag{4.27}$$

Though the damping factor was computed for fixed precisions, in practice ζ^t is calculated using the precisions at iteration t. Algorithm 8 shows the damped version of VAMP from [36]. However, this damping scheme doesn't work in some cases, especially in MRI with Cartesian sampling mask. VAMP can be unstable and diverge for high acceleration rates R and challenging sampling masks.

We propose a different damping scheme to stabilize VAMP and make it more robust. In Alg. 8, the precisions (γ_1, γ_2) control how much the denoiser $\boldsymbol{g}_1$ and linear estimator $\boldsymbol{g}_2$ act on their corresponding inputs in lines 10 and 5 respectively. VAMP usually becomes unstable when the precisions change rapidly between iterations. Thus it is important to damp the precisions too. The precisions γ_1 and γ_2 are calculated using the divergences α_1 and α_2 in lines 8 and 15 respectively. The divergences

(α_1, α_2) are computed using the Monte-Carlo method as shown in (4.15). Since $\boldsymbol{g}_2$ is linear, its divergence α_2 is quite accurate, whereas the divergence of $\boldsymbol{g}_1$ is not. To fix this, one could perform multiple Monte-Carlo draws to get a better estimate of α_1 or use a damping on α_1. In practice, multiple Monte-Carlo draws are expensive, so we choose to damp α_1.

Finally, it is important to note that the pseudo-measurements $(\boldsymbol{r}_1, \boldsymbol{r}_2)$ are amplitudes, the precisions (γ_1, γ_2) are inverse variances and the divergences (α_1, α_2) are variance scalings. This should be taken into account when damping, i.e.,

$$\boldsymbol{r}_2^{t+1} \leftarrow \zeta^t \boldsymbol{r}_2^{t+1} + (1-\zeta^t)\boldsymbol{r}_2^t \tag{4.28a}$$

$$\gamma_2^{t+1} \leftarrow \left\{ \frac{\zeta^t}{\sqrt{\gamma_2^{t+1}}} + \frac{1-\zeta^t}{\sqrt{\gamma_2^t}} \right\}^{-2} \tag{4.28b}$$

$$\alpha_1^t \leftarrow \left\{ \theta(\alpha_1^t)^{\frac{1}{2}} + (1-\theta)(\alpha_1^{t-1})^{\frac{1}{2}} \right\}^2, \tag{4.28c}$$

where ζ^t is the damping factor computed similar to (4.27) using the precisions at iteration t, that can also be written in terms of the divergences as,

$$\zeta^t = \frac{2}{1 + \max\left\{ \frac{\gamma_1^t}{\gamma_2^t}, \frac{\gamma_2^t}{\gamma_1^t} \right\}} \tag{4.29a}$$

$$\overset{(a)}{=} \frac{2}{1 + \max\left\{ \frac{1-\alpha_2^t}{\alpha_2^t}, \frac{1-\alpha_1^t}{\alpha_1^t} \right\}} \tag{4.29b}$$

$$= \frac{2}{\max\left\{ \frac{1}{\alpha_2^t}, \frac{1}{\alpha_1^t} \right\}} \tag{4.29c}$$

$$= 2\min\left\{ \alpha_1^t, \alpha_2^t \right\}, \tag{4.29d}$$

where (a) follows from lines 15 and 8 in Alg. 8.

Algorithm 8 Damped VAMP (dGEC) from [36]

require:
  damping factor: $\zeta \in (0, 1]$
initialize:
  $\boldsymbol{r}_2^0, \gamma_2^0$
for $t = 0, \dots, T_{\max}$ **do**
//Linear Stage
  $\boldsymbol{x}_2^t = \boldsymbol{g}_2(\boldsymbol{r}_2^t, \gamma_2^t)$
  $\alpha_2^t = \langle \boldsymbol{g}_2'(\boldsymbol{r}_2^t, \gamma_2^t) \rangle$
  $\boldsymbol{r}_1^t = (\boldsymbol{x}_2^t - \alpha_2^t \boldsymbol{r}_2^t)/(1 - \alpha_2^t)$
  $\gamma_1^t = \gamma_2^t(1 - \alpha_2^t)/\alpha_2^t$
//Denoising
  $\boldsymbol{x}_1^t = \boldsymbol{g}_1(\boldsymbol{r}_1^t, \gamma_1^t)$
  $\alpha_1^t = \langle \boldsymbol{g}_1(\boldsymbol{r}_1^t, \gamma_1^t) \rangle$
  $\zeta^t = 2 \min\{\alpha_1^t, \alpha_2^t\}$
  $\overline{\boldsymbol{r}}_2^{t+1} = (\boldsymbol{x}_1^t - \alpha_1^t \boldsymbol{r}_1^t)/(1 - \alpha_1^t)$
  $\boldsymbol{r}_2^{t+1} = \zeta^t \overline{\boldsymbol{r}}_2^{t+1} + (1 - \zeta^t)\boldsymbol{r}_2^t$
  $\gamma_2^{t+1} = \gamma_1^t(1 - \alpha_1^t)/\alpha_1^t$
end for

Using fast solvers in the Linear Stage

Recall that, in VAMP, the linear estimator $\boldsymbol{g}_2$ solves the following problem

$$\arg\min_{\boldsymbol{x}} \frac{\gamma_w}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2 + \frac{\gamma_1^t}{2}\|\boldsymbol{x} - \boldsymbol{r}_1^t\|^2. \tag{4.30}$$

Its closed-form solution in (2.18) involves a matrix inverse that is expensive in high dimensions. So one can use a fast iterative solver such as LSQR [65] or conjugate gradient [7] to solve (4.30) approximately, as mentioned in [84]. Let $\boldsymbol{g}_{\mathsf{linear\text{-}solver}}(\boldsymbol{r}_1^t, \gamma_1^t)$ denote the solution of a fast iterative solver for (4.30). Our proposed algorithm, Damped Denoising VAMP (DD-VAMP), is summarized in Alg. 9.

Algorithm 9 Damped Denoising-VAMP (DD-VAMP)

require:
  damping factor: $\theta = 0.1$
  $\zeta_{\min} = 0.2$, $\zeta_{\max} = 0.3$
initialize:
  $\boldsymbol{r}_2^0, \gamma_2^0$
for $t = 0, \ldots, T_{\max}$ **do**
//Linear Stage
  $\boldsymbol{x}_2^t = \boldsymbol{g}_{\mathsf{linear\text{-}solver}}(\boldsymbol{r}_2^t, \gamma_2^t)$
  $\alpha_2^t = \langle \boldsymbol{g}'_{\mathsf{linear\text{-}solver}}(\boldsymbol{r}_2^t, \gamma_2^t) \rangle$
  $\boldsymbol{r}_1^t = (\boldsymbol{x}_2^t - \alpha_2^t \boldsymbol{r}_2^t)/(1 - \alpha_2^t)$
  $\gamma_1^t = \gamma_2^t (1 - \alpha_2^t)/\alpha_2^t$
//Denoising
  $\boldsymbol{x}_1^t = \boldsymbol{g}_1(\boldsymbol{r}_1^t, \gamma_1^t)$
  $\overline{\alpha}_1^t = \langle \boldsymbol{g}'_1(\boldsymbol{r}_1^t, \gamma_1^t)$
  $\alpha_1^t = \{\theta(\overline{\alpha}_1^t)^{\frac{1}{2}} + (1-\theta)(\alpha_1^{t-1})^{\frac{1}{2}}\}^2$
  $\overline{\zeta}^t = 2\min\{\alpha_1^t, \alpha_2^t\}$
  $\zeta^t = \max\{\zeta_{\min}, \min\{\zeta_{\max}, \overline{\zeta}^t\}\}$
  $\overline{\boldsymbol{r}}_2^{t+1} = (\boldsymbol{x}_1^t - \alpha_1^t \boldsymbol{r}_1^t)/(1 - \alpha_1^t)$
  $\boldsymbol{r}_2^{t+1} = \zeta^t \overline{\boldsymbol{r}}_2^{t+1} + (1 - \zeta^t)\boldsymbol{r}_2^t$
  $\overline{\gamma}_2^{t+1} = \gamma_1^t (1 - \alpha_1^t)/\alpha_1^t$
  $\gamma_2^{t+1} = \{\zeta^t (\overline{\gamma}_2^{t+1})^{-\frac{1}{2}} + (1 - \zeta^t)(\gamma_2^t)^{-\frac{1}{2}}\}^{-2}$
end for

Figure 4.4: Ground truth, and Cartesian sampling and variable density sampling masks of acceleration $R = 4$

4.6 Numerical Experiments

In this section, we present the results of numerical experiments that study the behavior of our proposed algorithms, DD-AMP (Alg. 6) and DD-VAMP (Alg. 9), in comparison to PnP-ADMM [95], VDAMP [57] and BM3D-AMP-MRI [33] algorithms on image reconstruction in MRI. We considered both single-coil and multicoil MRI, and used real-valued images from the fastMRI [103] knee dataset. We considered real-valued images due to the lack of a good denoiser for complex valued images. As a future work, we aim to test our algorithms on the more challenging complex valued images.

For our experiments, we used a non fat-suppressed mid-slice knee MR image of size 128×128, and Cartesian and variable density sampling masks of acceleration $R = 4$, shown in Fig. 4.4. The noise precision γ_w was set to achieve SNR=40 dB. We tuned PnP-ADMM to maximize the PSNR.

4.6.1 Single-Coil MRI

We first tested our algorithms on single-coil MRI, where the measurements $\boldsymbol{y}$ are generated according to (4.1). Figure 4.5 shows the reconstruction PSNR and SSIM [99] of different plug-and-play algorithms for Cartesian sampling mask. Here, we used the BM3D [26] denoiser in DD-AMP, DD-VAMP and PnP-ADMM algorithms. VDAMP was designed for variable density sampling masks (Fig. 4.1) and doesn't work well with Cartesian masks. In Fig. 4.6, we used the DnCNN [104] denoiser included in the D-AMP toolbox[8]. Using a more powerful denoiser increased the accuracy of the algorithms, as expected. In Fig. 4.7, we used the DnCNN denoiser and variable density sampling mask. From Figures 4.5, 4.6 and 4.7, we see that the performance of DD-VAMP and PnP-ADMM is very similar and that DD-AMP performed a little worse.

[8] github.com/ricedsp/D-AMP_Toolbox

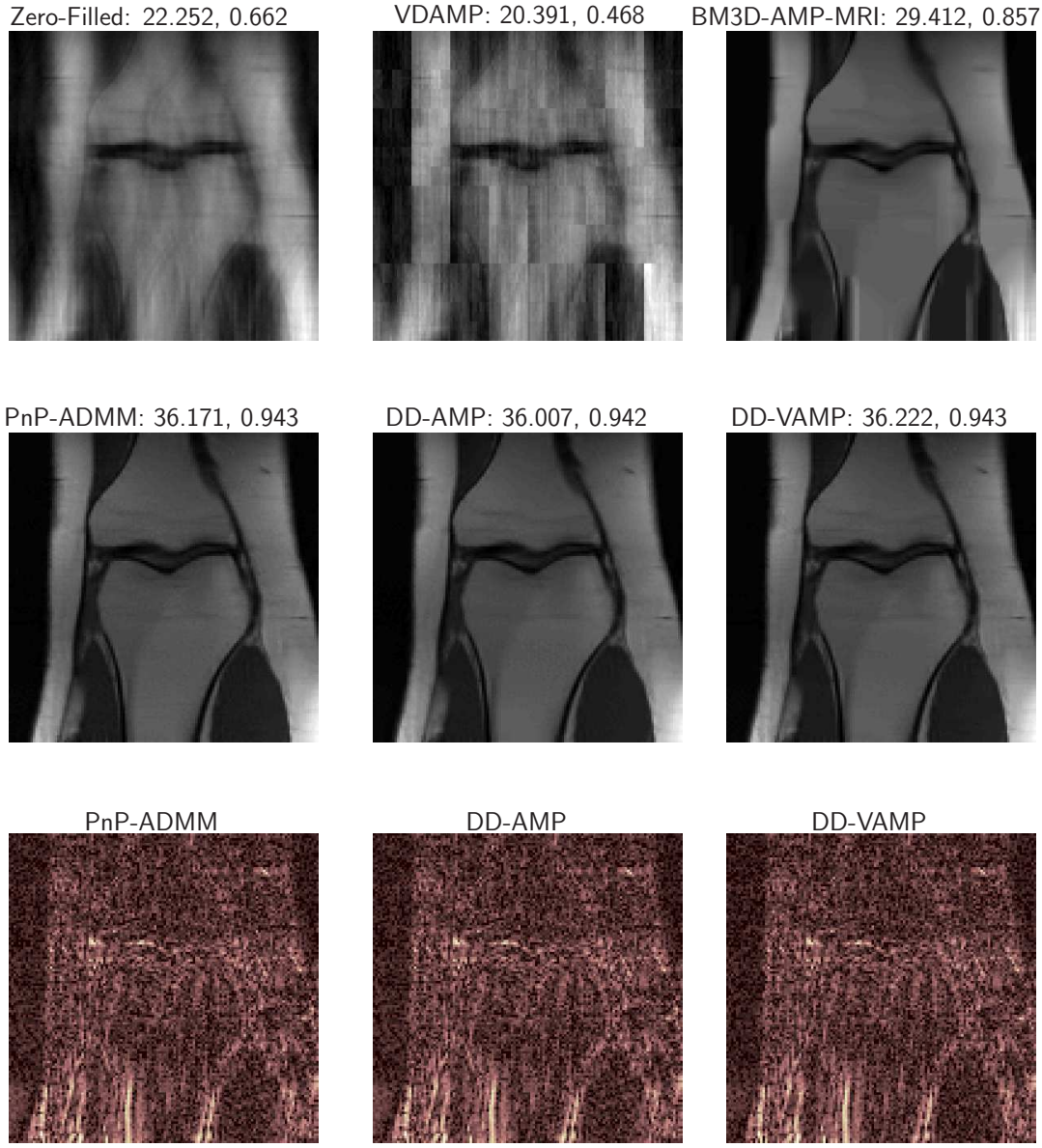

Figure 4.5: Reconstruction PSNR (dB) and SSIM of single-coil knee MRI for Cartesian sampling mask of acceleration $R = 4$. Using BM3D denoiser in DD-AMP, DD-VAMP and PnP-ADMM algorithms. Figures in last row are the error images.

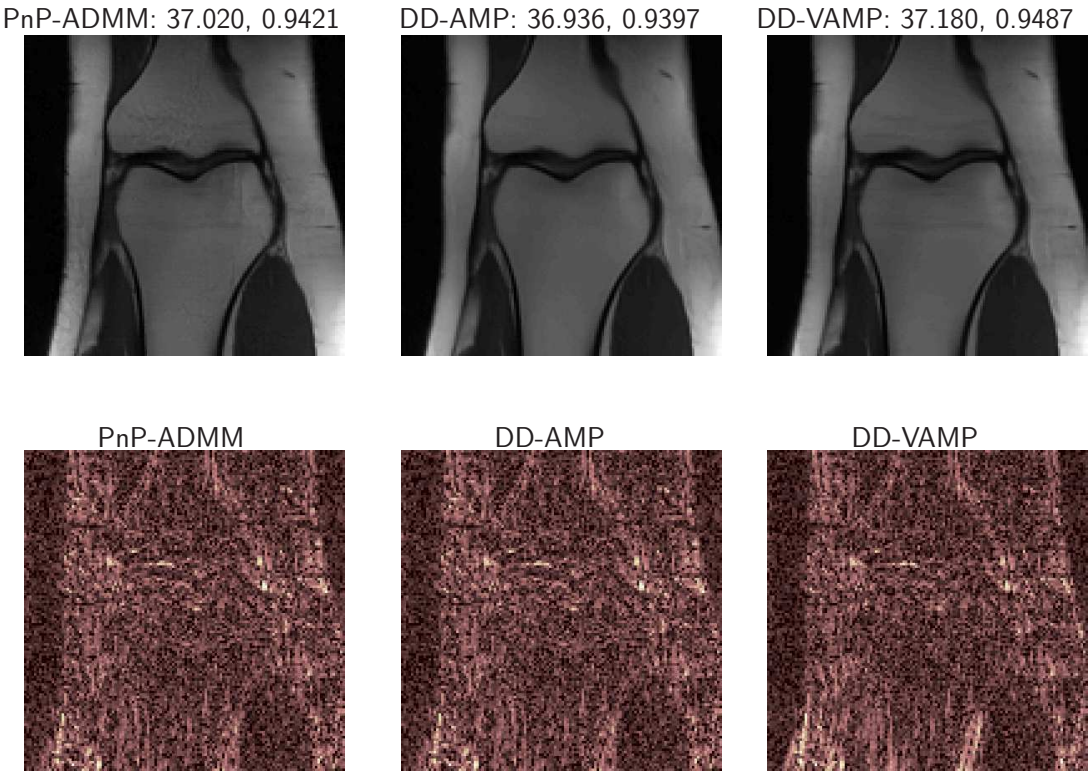

Figure 4.6: Reconstruction PSNR (dB) and SSIM of single-coil knee MRI for Cartesian sampling mask of acceleration $R = 4$. Using DnCNN denoiser in DD-AMP, DD-VAMP and PnP-ADMM algorithms. Figures in last row are error images.

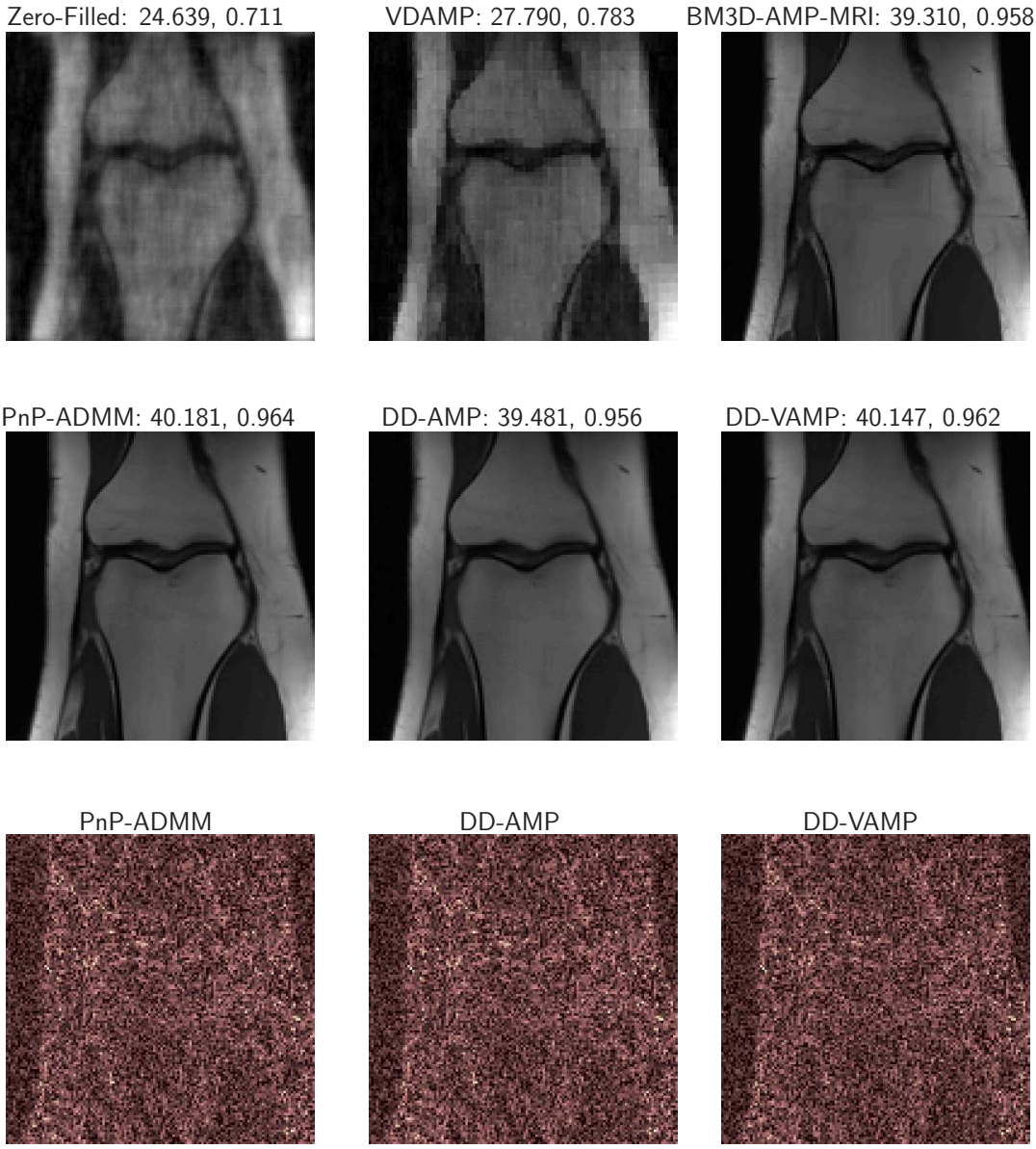

Figure 4.7: Reconstruction PSNR (dB) and SSIM of single-coil knee MRI using variable density sampling mask of acceleration $R = 4$. Using DnCNN denoiser in DD-AMP, DD-VAMP and PnP-ADMM algorithms. Figures in last row are the error images.

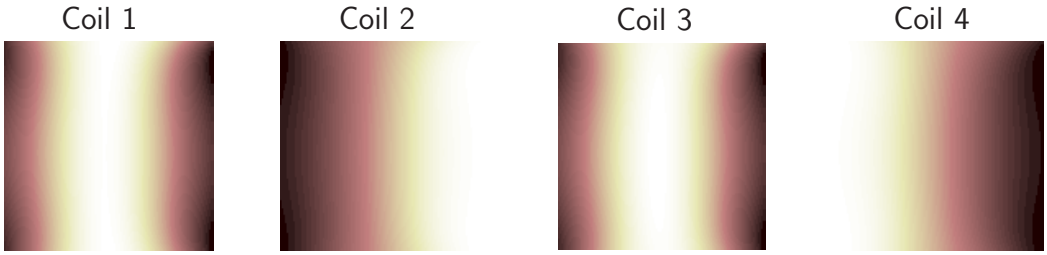

Figure 4.8: Coil sensitivities in multicoil MRI

4.6.2 Multicoil MRI

For our next experiment, we used $K = 4$ receiver coils and generated the coil sensitivity maps shown in Fig. 4.8 using the Biot-Savart law. VDAMP and BM3D-MRI-AMP are not applicable in this case, so we compared our algorithms against PnP-ADMM. We used Cartesian sampling and the BM3D denoiser in Fig. 4.9, and the DnCNN denoiser included in the D-AMP toolbox[9] in Fig. 4.10. In Fig. 4.11, we used the variable density sampling and DnCNN denoiser. Similar to the single-coil case, the performance of DD-VAMP and PnP-ADMM is similar and that DD-AMP performed a little worse.

[9]github.com/ricedsp/D-AMP_Toolbox

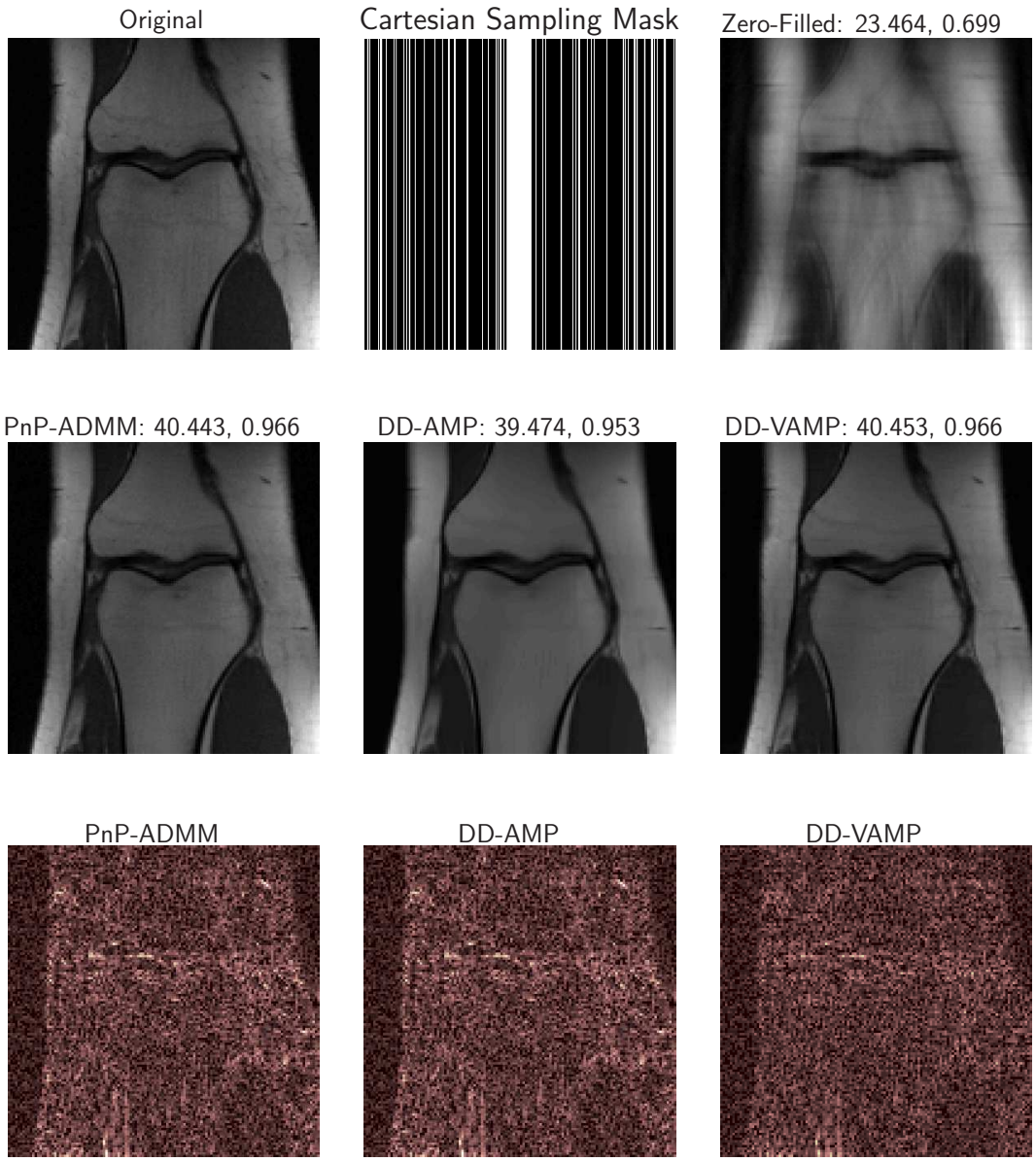

Figure 4.9: Reconstruction PSNR (dB) and SSIM of multicoil knee MRI for 4 receiver coils and Cartesian sampling of acceleration $R = 4$. Using BM3D denoiser in DD-AMP, DD-VAMP and PnP-ADMM. Figures in last row are the error images.

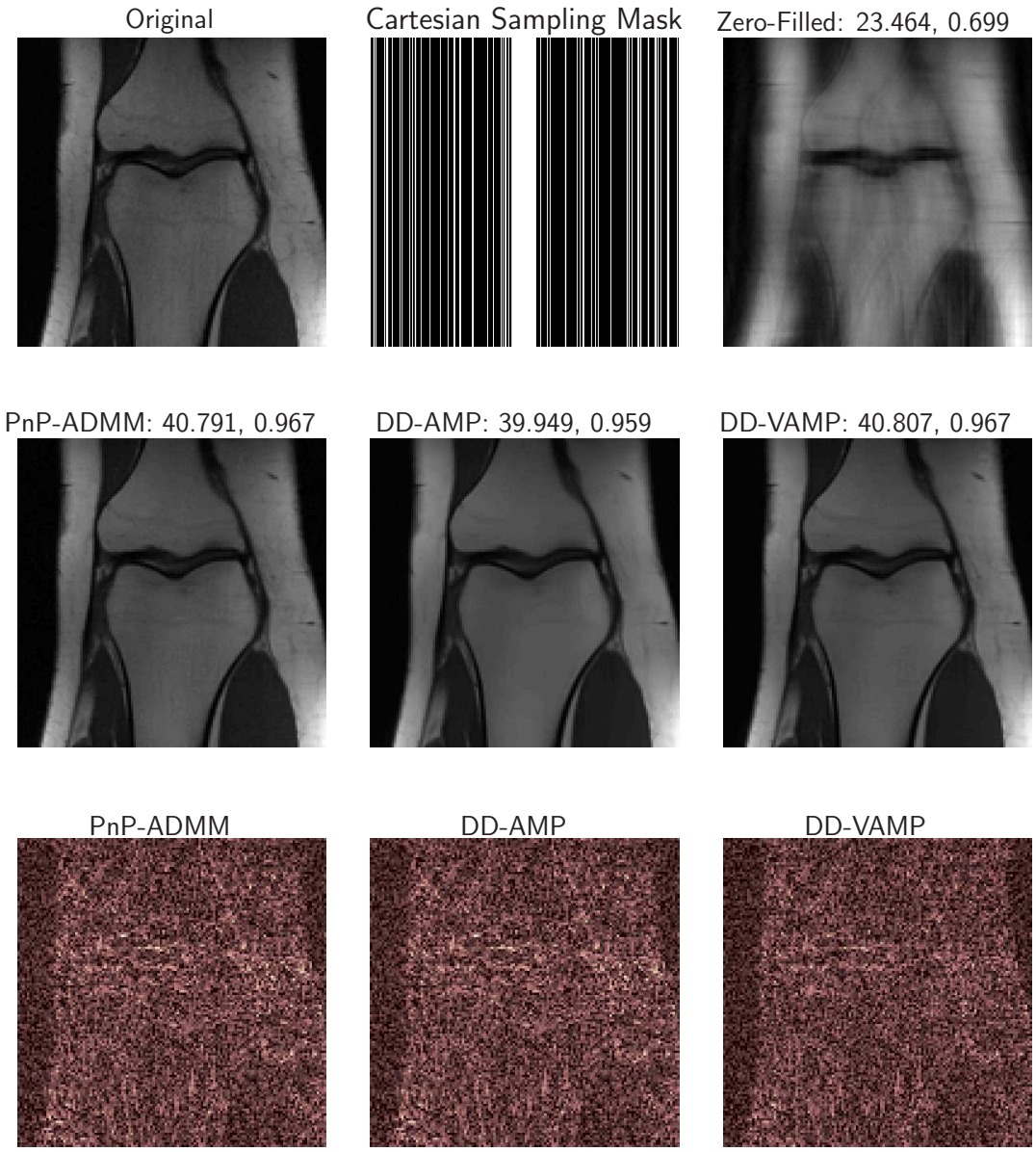

Figure 4.10: Reconstruction PSNR (dB) and SSIM of multicoil knee MRI for 4 receiver coils and Cartesian sampling of acceleration $R = 4$. Using DnCNN denoiser in DD-AMP, DD-VAMP and PnP-ADMM algorithms. Figures in last row are the error images.

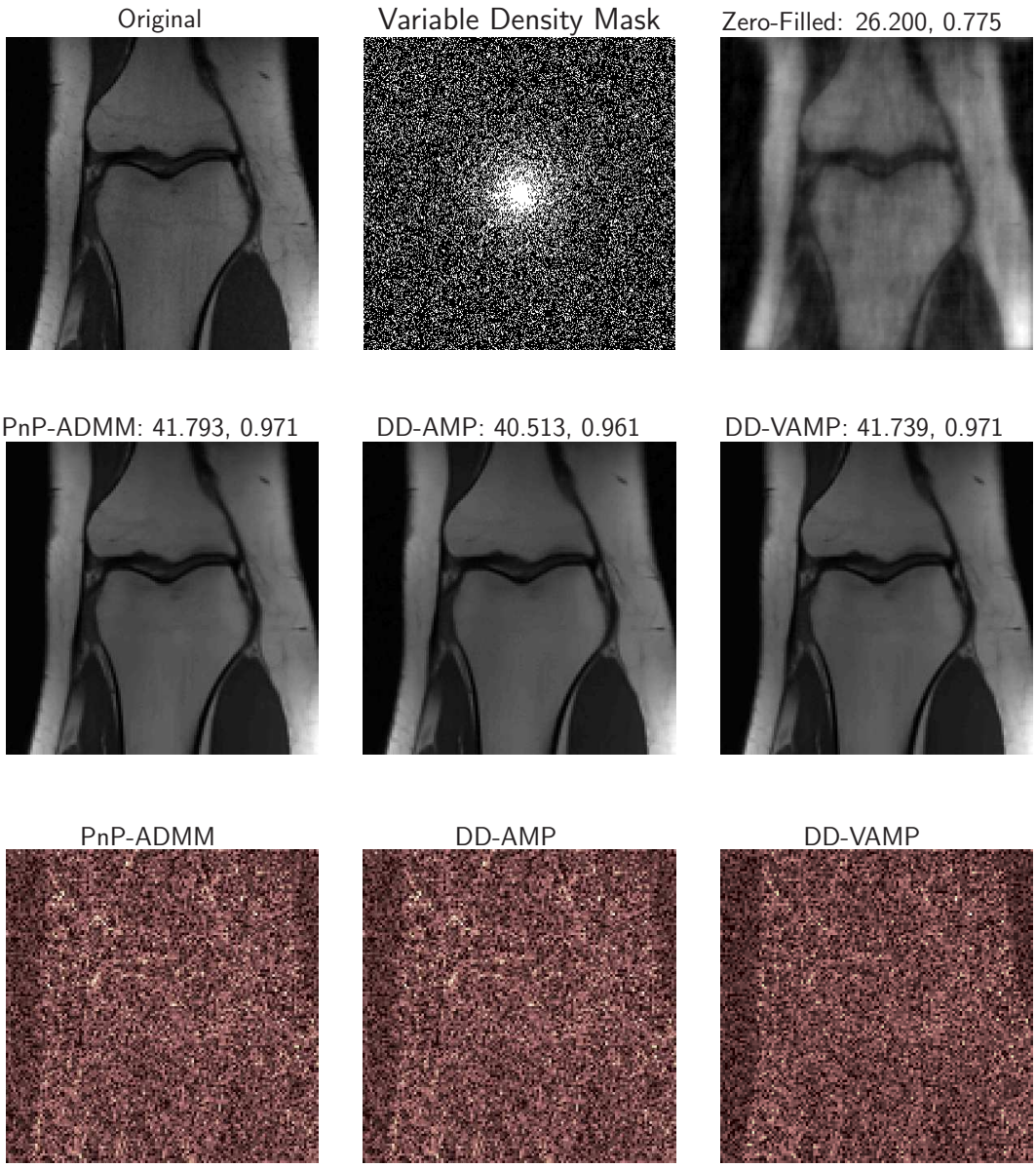

Figure 4.11: Reconstruction PSNR (dB) and SSIM of multicoil knee MRI for 4 receiver coils and variable density sampling of acceleration $R = 4$. Using DnCNN denoiser in DD-AMP, DD-VAMP and PnP-ADMM algorithms. Figures in last row are the error images.

4.6.3 Randomized Single-Coil MRI

In this experiment, we used a different measurement model in single-coil MRI,

$$\boldsymbol{y} = \boldsymbol{SFJDx} + \boldsymbol{w}, \tag{4.31}$$

where $\boldsymbol{J}$ is a random permutation matrix, i.e., a random shuffling of the columns of $\boldsymbol{I}$, and $\boldsymbol{D} = \text{Diag}\{d_1, \dots, d_N\}$ for d_i uniformly sampled from $\{-1, 1\}$. The matrix $\boldsymbol{JD}$ randomizes the measurements $\boldsymbol{y}$. This measurement model is not commonly seen in practice but we consider it to see whether the algorithms benefit from a simple randomization of the measurement operator.

We compared DD-AMP and DD-VAMP with PnP-ADMM using model (4.31). For single-coil MRI, Cartesian sampling mask of $R = 4$ and DnCNN denoiser, Fig. 4.12 shows the PSNR (dB) and SSIM performance of the algorithms. We see that the randomization in MRI helps AMP-type algorithms such as DD-AMP and DD-VAMP outperform PnP-ADMM. If one could implement the randomization $\boldsymbol{JD}$ in a real-life MRI scanner, then it would be very beneficial to AMP-type algorithms.

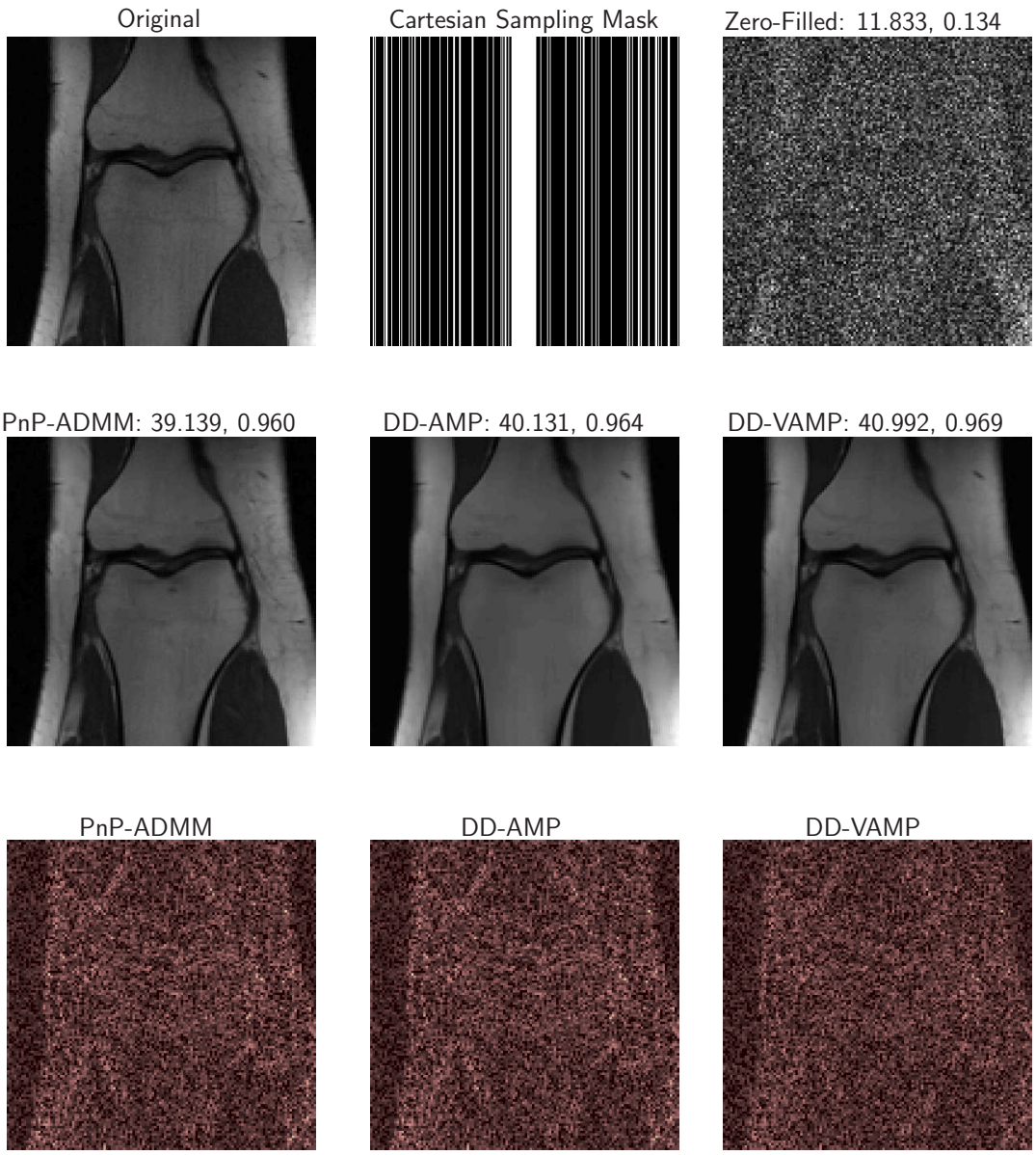

Figure 4.12: Reconstruction PSNR (dB) and SSIM of single-coil knee MRI with random coding, and Cartesian sampling of acceleration $R = 4$. Using DnCNN denoiser in DD-AMP, DD-VAMP and PnP-ADMM algorithms. Figures in last row are the error images.

Chapter 5: Conclusions

In this dissertation, we considered several linear and bilinear inverse problems and proposed algorithms to solve them using the Approximate Message Passing framework. We first proposed some theoretical results on the VAMP [73] algorithm, where we extended its state evolution (SE) analysis from separable to non-separable Lipschitz denoisers. Using the SE formalism, when the forward operator is right rotationally invariant (RRI), we can accurately predict the MSE performance of VAMP in high dimensions.

Next, we proposed the BAdVAMP algorithm for solving bilinear inverse problems. BAdVAMP jointly recovers the vector $\boldsymbol{b}$ and the matrix $\boldsymbol{C}$ from noisy measurements $\boldsymbol{Y} = \boldsymbol{A}(\boldsymbol{b})\boldsymbol{C} + \boldsymbol{W}$, where $\boldsymbol{A}(\cdot)$ is a known affine linear function of $\boldsymbol{b}$ (i.e., $\boldsymbol{A}(\boldsymbol{b}) = \boldsymbol{A}_0 + \sum_{i=1}^{Q} b_i \boldsymbol{A}_i$ with known matrices $\boldsymbol{A}_i$). To solve this problem, we combined the VAMP algorithm [75], the EM algorithm [61], and variance auto-tuning [45] in a manner appropriate for bilinear recovery. We demonstrated numerically that the proposed approach has robustness advantages over other state-of-the-art bilinear recovery algorithms, including lifted VAMP [35] and EM-PBiGAMP [66]. As future work, we hope to rigorously analyze BAd-VAMP through the state-evolution formalism.

Lastly, we proposed image reconstruction algorithms in MRI by modifying the existing AMP [32] and VAMP [73] algorithms. In AMP, we scaled the measurements in k-space by the inverse probability of the random sampling mask to debias the denoiser input error, which led to a significant increase in the reconstruction accuracy. In VAMP, we proposed an improved damping scheme that empirically stabilizes it for MRI. We numerically demonstrated that our proposed algorithms are more accurate than existing AMP-based algorithms in single-coil MRI. We showed that our algorithms are also applicable to multicoil MRI with known coil sensitivities. As a future work, we hope to do the following: rigorous analysis of the proposed damping scheme in VAMP; using ideas from BAdVAMP, learn the coil sensitivities and recover the image simultaneously in VAMP; design a good denoiser for complex-valued MR images and apply our algorithms.

Appendix A: Signal Denoisers

A.1 Singular Value Thresholding

Consider the estimation of a low-rank matrix $\boldsymbol{X}_0$ from linear measurements $\boldsymbol{y} = \mathcal{A}(\boldsymbol{X}_0)$, where $\mathcal{A}$ is some linear operator [16]. Writing the SVD of $\boldsymbol{R}$ as $\boldsymbol{R} = \sum_i \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^{\mathsf{T}}$, the SVT denoiser is defined as

$$\boldsymbol{g}_1(\boldsymbol{R}, \gamma) \triangleq \sum_i (\sigma_i - \gamma)_+ \boldsymbol{u}_i \boldsymbol{v}_i^{\mathsf{T}}, \tag{A.1}$$

where $(x)_+ \triangleq \max\{0, x\}$. To show that $\boldsymbol{g}_1(\cdot)$ in (A.1) is uniformly pseudo-Lipschitz, we first note that $\boldsymbol{g}_1(\cdot)$ is the proximal operator of the nuclear norm $\|\cdot\|_*$, i.e.,

$$\boldsymbol{g}_1(\boldsymbol{R}, \gamma) = \underset{\boldsymbol{X} \in \mathbb{R}^{N_1 \times N_2}}{\arg\min}\ \gamma \|\boldsymbol{X}\|_* + \frac{1}{2}\|\boldsymbol{X} - \boldsymbol{R}\|_F^2.$$

Define $\langle \cdot, \cdot \rangle$ to be the standard inner product, i.e., $\langle \boldsymbol{R}_1, \boldsymbol{R}_2 \rangle = \mathrm{tr}\{\boldsymbol{R}_2^{\mathsf{T}} \boldsymbol{R}_1\}$. From [24], we have that $\boldsymbol{g}_1(\cdot)$ is non-expansive because the nuclear norm is convex and proper, i.e., we can write

$$\begin{aligned} &\|\boldsymbol{g}_1(\boldsymbol{R}_1, \gamma) - \boldsymbol{g}_1(\boldsymbol{R}_2, \gamma)\|_F^2 \leq \langle \boldsymbol{R}_1 - \boldsymbol{R}_2, \boldsymbol{g}_1(\boldsymbol{R}_1, \gamma) - \boldsymbol{g}_1(\boldsymbol{R}_2, \gamma) \rangle \\ \Rightarrow\ &\|\boldsymbol{g}_1(\boldsymbol{R}_1, \gamma) - \boldsymbol{g}_1(\boldsymbol{R}_2, \gamma)\|_F \leq \|\boldsymbol{R}_1 - \boldsymbol{R}_2\|_F. \end{aligned} \tag{A.2}$$

Let the SVD of $\boldsymbol{R}_2 \in \mathbb{R}^{N_1 \times N_2}$ be $\sum_{i=1}^{\min\{N_1,N_2\}} \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^{\mathsf{T}}$. We can generalize the Lipschitz condition in (A.2) into

$$\begin{aligned}
\|\boldsymbol{g}_1(\boldsymbol{R}_1,\gamma_1) - \boldsymbol{g}_1(\boldsymbol{R}_2,\gamma_2)\|_F &= \|\boldsymbol{g}_1(\boldsymbol{R}_1,\gamma_1) - \boldsymbol{g}_1(\boldsymbol{R}_2,\gamma_1) + \boldsymbol{g}_1(\boldsymbol{R}_2,\gamma_1) - \boldsymbol{g}_1(\boldsymbol{R}_2,\gamma_2)\|_F \\
&\leq \|\boldsymbol{g}_1(\boldsymbol{R}_1,\gamma_1) - \boldsymbol{g}_1(\boldsymbol{R}_2,\gamma_1)\|_F + \|\boldsymbol{g}_1(\boldsymbol{R}_2,\gamma_1) - \boldsymbol{g}_1(\boldsymbol{R}_2,\gamma_2)\|_F \\
&\leq \|\boldsymbol{R}_1 - \boldsymbol{R}_2\|_F + \|\boldsymbol{g}_1(\boldsymbol{R}_2,\gamma_1) - \boldsymbol{g}_1(\boldsymbol{R}_2,\gamma_2)\|_F \\
&\overset{\text{(a)}}{=} \|\boldsymbol{R}_1 - \boldsymbol{R}_2\|_F \\
&\quad + \left\| \sum_{i=1}^{\min\{N_1,N_2\}} ((\sigma_i - \gamma_1)_+ - (\sigma_i - \gamma_2)_+) \boldsymbol{u}_i \boldsymbol{v}_i^{\mathsf{T}} \right\|_F \\
&\leq \|\boldsymbol{R}_1 - \boldsymbol{R}_2\|_F + \sum_{i=1}^{\min\{N_1,N_2\}} |(\sigma_i - \gamma_1)_+ - (\sigma_i - \gamma_2)_+| \\
&\leq \|\boldsymbol{R}_1 - \boldsymbol{R}_2\|_F + \min\{N_1,N_2\}|\gamma_1 - \gamma_2| \\
&\overset{\text{(b)}}{\leq} \|\boldsymbol{R}_1 - \boldsymbol{R}_2\|_F + \sqrt{N}|\gamma_1 - \gamma_2|,
\end{aligned}$$

where in (a) we have used the the definition of $\boldsymbol{g}_1(\cdot)$ from (A.1) and the SVD of $\boldsymbol{R}_2$, and in (b) we used $\min\{N_1,N_2\} \leq \sqrt{N_1 N_2} = \sqrt{N}$. Next, we show that $\boldsymbol{g}_1(\cdot)$ also satisfies the convergence conditions in Definition 2. Let $\boldsymbol{Z}_1$ and $\boldsymbol{Z}_2$ be two sequences constructed according to Definition 1 and let $\boldsymbol{x}_0$ be the true signal. Assume that

$$\lim_{N\to\infty} \frac{1}{N}\|\boldsymbol{X}_0\|_F^2 \quad \text{and} \quad \lim_{N\to\infty} \frac{1}{N}\langle \boldsymbol{X}_0, \boldsymbol{Z}_1 \rangle \quad \text{exist almost surely.} \tag{A.3}$$

If we write $\widetilde{\boldsymbol{g}}(\boldsymbol{R},\gamma) = \mathsf{vec}(\boldsymbol{g}_1(\boldsymbol{R},\gamma)) \triangleq [g_{1,1}(\boldsymbol{R},\gamma),\ldots,g_{1,N}(\boldsymbol{R},\gamma)]^{\mathsf{T}}$, then the following series converges because it is bounded:

$$\begin{aligned}
&\lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} |\widetilde{g}_i(\boldsymbol{X}_0 + \boldsymbol{Z}_1,\gamma_1)\widetilde{g}_i(\boldsymbol{X}_0 + \boldsymbol{Z}_2,\gamma_2)| \\
&\leq \lim_{N\to\infty} \frac{1}{N} \|\widetilde{\boldsymbol{g}}(\boldsymbol{X}_0 + \boldsymbol{Z}_1,\gamma_1)\|_2 \|\widetilde{\boldsymbol{g}}(\boldsymbol{X}_0 + \boldsymbol{Z}_2,\gamma_2)\|_2 \\
&\leq \lim_{N\to\infty} \sqrt{\frac{1}{N}\|\boldsymbol{X}_0 + \boldsymbol{Z}_1\|_F^2} \sqrt{\frac{1}{N}\|\boldsymbol{X}_0 + \boldsymbol{Z}_2\|_F^2} \\
&\overset{\text{(a)}}{<} \infty,
\end{aligned}$$

where (a) follows from the assumption (A.3). Since absolute convergence implies convergence, the following series also converges:

$$\begin{aligned}&\lim_{N\to\infty}\frac{1}{N}\langle \boldsymbol{g}_1(\boldsymbol{X}_0+\boldsymbol{Z}_1,\gamma_1),\boldsymbol{g}_1(\boldsymbol{X}_0+\boldsymbol{Z}_2,\gamma_2)\rangle\\&=\lim_{N\to\infty}\frac{1}{N}\sum_{i=1}^{N}\widetilde{g}_i(\boldsymbol{X}_0+\boldsymbol{Z}_1,\gamma_1)\widetilde{g}_i(\boldsymbol{X}_0+\boldsymbol{Z}_2,\gamma_2).\end{aligned}\tag{A.4}$$

If we choose the covariance matrix in Definition 1 to be $\boldsymbol{S}=\left[\begin{smallmatrix}1&0\\0&0\end{smallmatrix}\right]$, then we get

$$\lim_{N\to\infty}\frac{1}{N}\langle \boldsymbol{g}_1(\boldsymbol{X}_0+\boldsymbol{Z}_1,\gamma_1),\boldsymbol{X}_0\rangle=\lim_{N\to\infty}\frac{1}{N}\langle \boldsymbol{g}_1(\boldsymbol{X}_0+\boldsymbol{Z}_1,\gamma_1),\boldsymbol{g}_1(\boldsymbol{X}_0+\boldsymbol{Z}_2,0)\rangle.\tag{A.5}$$

Thus, (A.5) also converges since it is a special case of (A.4). Similarly, it can be shown that $\frac{1}{N}\langle \boldsymbol{Z}_2,\boldsymbol{g}_1(\boldsymbol{X}_0+\boldsymbol{Z}_1,\gamma_1)\rangle$ is uniformly Lipschitz.

Bibliography

[1] A. Agarwal, S. Negahban, and M. J. Wainwright. Matrix decomposition via convex relaxation: Optimal rates in high dimensions. *Ann. Statist.*, 40(2):1171–1197, 2012.

[2] Michal Aharon, Michael Elad, and Alfred Bruckstein. k-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Trans. Signal Process.*, 54(11):4311–4322, 2006.

[3] A. Ahmed, B. Recht, and J. Romberg. Blind deconvolution using convex programming. *IEEE Trans. Inform. Theory*, 60(3):1711–1732, 2014.

[4] S. D. Babacan, M. Luessi, R. Molina, and A. K. Katsaggelos. Sparse Bayesian methods for low-rank matrix estimation. *IEEE Trans. Signal Process.*, 60(8):3964–3977, August 2012.

[5] Laura Balzano, Robert Nowak, and Benjamin Recht. Online identification and tracking of subspaces from highly incomplete information. In *Proc. Allerton Conf. Commun. Control Comput.*, pages 704–711, September 2010.

[6] Jean Barbier, Mohamad Dia, Nicolas Macris, and Florent Krzakala. The mutual information in random linear estimation. In *Proc. Allerton Conf. Commun. Control Comput.*, pages 625–632, 2016.

[7] Richard Barrett, Michael Berry, Tony F Chan, James Demmel, June Donato, Jack Dongarra, Victor Eijkhout, Roldan Pozo, Charles Romine, and Henk Van der Vorst. *Templates for the solution of linear systems: building blocks for iterative methods*. SIAM, 1994.

[8] M. Bayati, M. Lelarge, and A. Montanari. Universality in polytope phase transitions and message passing algorithms. *Ann. App. Prob.*, 25(2):753–822, 2015.

[9] M. Bayati and A. Montanari. The dynamics of message passing on dense graphs, with applications to compressed sensing. *IEEE Trans. Inform. Theory*, 57(2):764–785, February 2011.

[10] A. Beck and M. Teboulle. Fast gradient-based algorithms for constrained total variation image denoising and deblurring problem. *IEEE Trans. Image Process.*, 18(11):2419–2434, 2009.

[11] S. Bellini. Bussgang techniques for blind deconvolution and equalization. In S. Haykin, editor, *Blind Deconvolution*, chapter 2, pages 8–59. Prentice Hall, 1994.

[12] Raphael Berthier, Andrea Montanari, and Phan-Minh Nguyen. State evolution for approximate message passing with non-separable functions. *Inform. Inference*, 2019.

[13] C. Bilen, G. Puy, and R. Gribonval. Convex optimization approaches for blind sensor calibration using sparsity. *IEEE Trans. Signal Process.*, 62(18):4847–4856, 2014.

[14] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.*, 3(1):1–122, 2011.

[15] A. Buades, B. Coll, and J.-M. Morel. A non-local algorithm for image denoising. In *Proc. IEEE Conf. Comp. Vision Pattern Recog.*, volume 2, pages 60–65, 2005.

[16] J.-F. Cai, E. J. Candès, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM J. Optim.*, 20(4):1956–1982, January 2010.

[17] Burak Çakmak and Ole Winther Bernard H. Fleury. S-AMP for non-linear observation models. In *Proc. IEEE Int. Symp. Inform. Thy.*, pages 2807–2811, 2015.

[18] F. Caltagirone, F. Krzakala, and L. Zdeborová. On convergence of approximate message passing. In *Proc. IEEE Int. Symp. Inform. Thy.*, pages 1812–1816, July 2014.

[19] E. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Trans. Inform. Theory*, 52(2):489–509, February 2006.

[20] E. J. Candès, X. Li, Yi Ma, and J. Wright. Robust principal component analysis? *J. ACM*, 58(3):11, May 2011.

[21] E. J. Candès and Y. Plan. Matrix completion with noise. *Proc. IEEE*, 98(6):925–936, June 2010.

[22] E. J. Candès and Y. Plan. Tight oracle inequalities for low-rank matrix recovery from a minimal number of noisy random measurements. *IEEE Trans. Inform. Theory*, 57(4):2342–2359, 2011.

[23] E. J. Candès, T. Strohmer, and V. Voroninski. PhaseLift: Exact and stable signal recovery from magnitude measurements via convex programming. *Commun. Pure & Appl. Math.*, 66(8):1241–1274, 2013.

[24] Emmanuel J Candes, Carlos A Sing-Long, and Joshua D Trzasko. Unbiased risk estimates for singular value thresholding and spectral estimators. *IEEE Trans. Signal Process.*, 61(19):4643–4657, 2013.

[25] Stanley H Chan, Xiran Wang, and Omar A Elgendy. Plug-and-play admm for image restoration: Fixed-point convergence and applications. *IEEE Trans. Comp. Imag.*, 3(1):84–98, 2016.

[26] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian. Image denoising by sparse 3-D transform-domain collaborative filtering. *IEEE Trans. Image Process.*, 16(8):2080–2095, 2007.

[27] I. Daubechies, M. Defrise, and C. D. Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint,. *Commun. Pure & Appl. Math.*, 57(11):1413–1457, November 2004.

[28] M. A. Davenport and J. Romberg. An overview of low-rank matrix recovery from incomplete observations. *IEEE J. Sel. Topics Signal Process.*, 10(4):608–622, 2016.

[29] A. Dempster, N. M. Laird, and D. B. Rubin. Maximum-likelihood from incomplete data via the EM algorithm. *J. Roy. Statist. Soc.*, 39:1–17, 1977.

[30] Yash Deshpande and Andrea Montanari. Information-theoretically optimal sparse PCA. *Proc. IEEE Int. Symp. Inform. Thy.*, pages 2197–2201, 2014.

[31] D. L. Donoho. Compressed sensing. *IEEE Trans. Inform. Theory*, 52(9):1289–1306, September 2006.

[32] D. L. Donoho, A. Maleki, and A. Montanari. Message passing algorithms for compressed sensing. *Proc. Nat. Acad. Sci.*, 106(45):18914–18919, November 2009.

[33] Ender M Eksioglu and A Korhan Tanc. Denoising amp for mri reconstruction: Bm3d-amp-mri. *SIAM J. Imag. Sci.*, 11(3):2090–2109, 2018.

[34] Y. C. Eldar and G. Kutyniok. *Compressed Sensing: Theory and Applications*. Cambridge Univ. Press, New York, 2012.

[35] A. K. Fletcher, P. Pandit, S. Rangan, S. Sarkar, and P. Schniter. Plug-in estimation in high-dimensional linear inverse problems: A rigorous analysis. In *Proc. Neural Inform. Process. Syst. Conf.*, pages 7440–7449, 2018.

[36] Alyson K. Fletcher, Mojtaba Sahraee-Ardakan, Sundeep Rangan, and Philip Schniter. Expectation consistent approximate inference: Generalizations and convergence. In *Proc. IEEE Int. Symp. Inform. Thy.*, pages 190–194, 2016.

[37] Alyson K. Fletcher, Mojtaba Sahraee-Ardakan, Sundeep Rangan, and Philip Schniter. Rigorous dynamics and consistent estimation in arbitrarily conditioned linear systems. In *Proc. Neural Inform. Process. Syst. Conf.*, pages 2542–2551, 2017.

[38] T. Goldstein, C. Studer, and R. Baraniuk. Forward-backward splitting with a FASTA implementation. *arXiv:1411.3406*, 2014.

[39] Jun He, Laura Balzano, and Arthur Szlam. Incremental gradient on the Grassmannian for online foreground and background separation in subsampled video. In *Proc. IEEE Conf. Comp. Vision Pattern Recog.*, pages 1568–1575, 2012.

[40] C. Hegde and R. G. Baraniuk. Sampling and recovery of pulse streams. *IEEE Trans. Signal Process.*, 59(14):1505–1517, 2011.

[41] T. Heskes and O. Zoeter. Expectation propagation for approximate inference in dynamic Bayesian networks. In *Proc. Uncertainty Artif. Intell.*, pages 313–320, 2002.

[42] Yoshiyuki Kabashima, Florent Krzakala, Marc Mézard, Ayaka Sakata, and Lenka Zdeborová. Phase transitions and sample complexity in Bayes-optimal matrix factorization. *IEEE Trans. Inform. Theory*, 62(7):4228–4265, 2016.

[43] G. K. Kaleh and R. Vallet. Joint parameter estimation and symbol detection for linear or nonlinear unknown channels. *IEEE Trans. Commun.*, 42:2406–2413, Jul. 1994.

[44] U. Kamilov, H. Mansour, and B. Wohlberg. A plug-and-play priors approach for solving nonlinear imaging inverse problems. *IEEE Signal Process. Lett.*, 24(12):1872–1876, May 2017.

[45] U. S. Kamilov, S. Rangan, A. K. Fletcher, and M. Unser. Approximate message passing with consistent parameter estimation and applications to sparse learning. *IEEE Trans. Inform. Theory*, 60(5):2969–2985, May 2014.

[46] A. Kyrillidis and V. Cevher. Matrix recipes for hard thresholding methods. *J. Math. Imaging Vis.*, 48:235–265, 2014.

[47] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Proc. Neural Inform. Process. Syst. Conf.*, pages 556–562, 2001.

[48] Thibault Lesieur, Florent Krzakala, and Lenka Zdeborova. Phase transitions in sparse PCA. In *Proc. IEEE Int. Symp. Inform. Thy.*, pages 1635–1639, 2015.

[49] Thibault Lesieur, Florent Krzakala, and Lenka Zdeborov. Constrained low-rank matrix estimation: Phase transitions, approximate message passing and applications. *J. Stat. Mech.*, 2017(7):073403, 2017.

[50] Z. Lin, M. Chen, L. Wu, and Y. Ma. The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices. *arXiv:1009.5055*, 2010.

[51] S. Ling and T. Strohmer. Self-calibration and biconvex compressive sensing. *Inverse Problems*, 31(11):115002, 2015.

[52] Pengju Liu, Hongzhi Zhang, Kai Zhang, Liang Lin, and Wangmeng Zuo. Multi-level wavelet-cnn for image restoration. In *Proc. IEEE Conf. Comp. Vision Pattern Recog.*, pages 773–782, 2018.

[53] M. Lustig, D. L. Donoho, J. M. Santos, and J. M. Pauly. Compressed sensing MRI. *IEEE Signal Process. Mag.*, 25(2):72–82, March 2008.

[54] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *J. Mach. Learn. Res.*, 11:19–60, January 2010.

[55] Ryosuke Matsushita and Toshiyuki Tanaka. Low-rank matrix reconstruction and clustering via approximate message passing. In *Proc. Neural Inform. Process. Syst. Conf.*, pages 917–925, 2013.

[56] Christopher A. Metzler, Arian Maleki, and Richard G. Baraniuk. From denoising to compressed sensing. *IEEE Trans. Inform. Theory*, 62(9):5117–5144, 2016.

[57] Charles Millard, Aaron T Hess, Boris Mailhé, and Jared Tanner. Approximate message passing with a colored aliasing model for variable density fourier sampled images. *arXiv:2003.02701*, 2020.

[58] T. Minka. *A Family of Approximate Algorithms for Bayesian Inference*. PhD thesis, Dept. Comp. Sci. Eng., MIT, Cambridge, MA, Jan. 2001.

[59] Léo Miolane. Fundamental limits of low-rank matrix estimation: The non-symmetric case. *arXiv:1702.00473*, 2017.

[60] B. K. Natarajan. Sparse approximate solutions to linear systems. *SIAM J. Scientific Comput.*, 24(2):227–234, 1995.

[61] Radford Neal and Geoffrey Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In Michael I. Jordan, editor, *Learning in Graphical Models*, pages 355–368. MIT Press, 1998.

[62] Shunsuke Ono. Primal-dual plug-and-play image restoration. *IEEE Signal Process. Lett.*, 24(8):1108–1112, 2017.

[63] M. Opper and O. Winther. Expectation consistent free energies for approximate inference. In *Proc. Neural Inform. Process. Syst. Conf.*, pages 1001–1008, 2005.

[64] Manfred Opper and Ole Winther. Adaptive and self-averaging thouless-anderson-palmer mean-field theory for probabilistic modeling. *Physical Rev. E*, 64(5):056131, 2001.

[65] Christopher C Paige and Michael A Saunders. Lsqr: An algorithm for sparse linear equations and sparse least squares. *ACM Transactions on Mathematical Software (TOMS)*, 8(1):43–71, 1982.

[66] J. T. Parker and P. Schniter. Parametric bilinear generalized approximate message passing. *IEEE J. Sel. Topics Signal Process.*, 10(4):795–808, 2016.

[67] J. T. Parker, P. Schniter, and V. Cevher. Bilinear generalized approximate message passing—Part I: Derivation. *IEEE Trans. Signal Process.*, 62(22):5839–5853, November 2014.

[68] J. T. Parker, P. Schniter, and V. Cevher. Bilinear generalized approximate message passing—Part II: Applications. *IEEE Trans. Signal Process.*, 62(22):5854–5867, November 2014.

[69] Lawrence R Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[70] Sathish Ramani, Thierry Blu, and Michael Unser. Monte-Carlo SURE: A black-box optimization of regularization parameters for general denoising algorithms. *IEEE Trans. Image Process.*, 17(9):1540–1554, 2008.

[71] S. Rangan. Generalized approximate message passing for estimation with random linear mixing. In *Proc. IEEE Int. Symp. Inform. Thy.*, pages 2168–2172, August 2011. (full version at *arXiv:1010.5141*).

[72] S. Rangan and A. K. Fletcher. Iterative estimation of constrained rank-one matrices in noise. In *Proc. IEEE Int. Symp. Inform. Thy.*, pages 1246–1250, 2012.

[73] S. Rangan, P. Schniter, and A. K. Fletcher. Vector approximate message passing. *IEEE Trans. Inform. Theory*, to appear (see also arXiv:1610.03082).

[74] S. Rangan, P. Schniter, E. Riegler, A. Fletcher, and V. Cevher. Fixed points of generalized approximate message passing with arbitrary matrices. *IEEE Trans. Inform. Theory*, 62(12):7464–7474, December 2016.

[75] Sundeep Rangan, Philip Schniter, and Alyson K Fletcher. Vector approximate message passing. In *Proc. IEEE Int. Symp. Inform. Thy.*, pages 1588–1592, 2017.

[76] E. T. Reehorst and P. Schniter. Regularization by denoising: Clarifications and new interpretations. *IEEE Trans. Comp. Imag.*, 5(1):52–67, March 2019.

[77] G. Reeves. Additivity of information in multilayer networks via additive Gaussian noise transforms. In *Proc. Allerton Conf. Commun. Control Comput.*, pages 1064–1070, 2017.

[78] Galen Reeves and Henry D. Pfister. The replica-symmetric prediction for compressed sensing with Gaussian matrices is exact. In *Proc. IEEE Int. Symp. Inform. Thy.*, 2016.

[79] Yaniv Romano, Michael Elad, and Peyman Milanfar. The little engine that could: Regularization by denoising (RED). *SIAM J. Imag. Sci.*, 10(4):1804–1844, 2017.

[80] Elad Romanov and Matan Gavish. Near-optimal matrix recovery from random linear measurements. *Proc. Nat. Acad. Sci.*, 2018.

[81] Ron Rubinstein, Alfred M Bruckstein, and Michael Elad. Dictionaries for sparse representation modeling. *Proc. IEEE*, 98(6):1045–1057, 2010.

[82] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D*, 60:259–268, 1992.

[83] Cynthia Rush and Ramji Venkataramanan. Finite-sample analysis of approximate message passing. In *Proc. IEEE Int. Symp. Inform. Thy.*, pages 755–759, 2016.

[84] Philip Schniter, Sundeep Rangan, and Alyson Fletcher. Plug-and-play image recovery using vector amp. *Proc. Intl. Biomed. Astronom. Signal Process. (BASP) Frontiers Workshop*, 2017.

[85] C. Schülke, P. Schniter, and L. Zdeborová. Phase diagram of matrix compressed sensing. *Physical Rev. E*, 94(6):062136(1–16), December 2016.

[86] M. Seeger. Expectation propagation for exponential families. Technical Report 161464, EPFL, 2005.

[87] Daniel A Spielman, Huan Wang, and John Wright. Exact recovery of sparsely-used dictionaries. In *Proc. Conf. Learning Thy.*, pages 37.1–37.18, 2012.

[88] P. Sun, Z. Wang, and P. Schniter. Joint channel-estimation and equalization of single-carrier systems via bilinear AMP. *IEEE Trans. Signal Process.*, 66(10):2772–2785, 2018.

[89] K. Takeuchi. Rigorous dynamics of expectation-propagation-based signal recovery from unitarily invariant measurements. In *Proc. IEEE Int. Symp. Inform. Thy.*, pages 501–505, 2017.

[90] D. J. Thouless, P. W. Anderson, and R. G. Palmer. Solution of solvable model of a spin glass. *Phil. Mag.*, 35:593–601, 1977.

[91] A. N. Tikhonov and V. Arsenin. *Solutions of Ill-Posed Problems*. V.H. Winston and Sons, Washington, 1977.

[92] Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *Proc. IEEE Int. Conf. Comput. Vis.*, pages 839–846. IEEE, 1998.

[93] Antonia M. Tulino, Giuseppe Caire, Sergio Verdú, and Shlomo Shamai (Shitz). Support recovery with sparsely sampled free random matrices. *IEEE Trans. Inform. Theory*, 59(7):4243–4271, July 2013.

[94] Martin Uecker, Peng Lai, Mark J Murphy, Patrick Virtue, Michael Elad, John M Pauly, Shreyas S Vasanawala, and Michael Lustig. Espiritan eigenvalue approach to autocalibrating parallel mri: where sense meets grappa. *Magnetic Resonance Med.*, 71(3):990–1001, 2014.

[95] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg. Plug-and-play priors for model based reconstruction. In *Proc. IEEE Global Conf. Signal Info. Process.*, pages 945–948, 2013.

[96] J. Vila, P. Schniter, S. Rangan, F. Krzakala, and L. Zdeborová. Adaptive damping and mean removal for the generalized approximate message passing algorithm. In *Proc. IEEE Int. Conf. Acoust. Speech & Signal Process.*, pages 2021–2025, 2015.

[97] J. P. Vila and P. Schniter. Expectation-maximization Gaussian-mixture approximate message passing. *IEEE Trans. Signal Process.*, 61(19):4658–4672, Oct. 2013.

[98] M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Found. Trends Mach. Learn.*, 1, May 2008.

[99] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[100] A. E. Waters, A. C. Sankaranarayanan, and R. G. Baraniuk. SpaRCS: Recovering low-rank and sparse matrices from compressive measurements. In *Proc. Neural Inform. Process. Syst. Conf.*, pages 1089–1097, 2011.

[101] Zaiwen Wen, Wotao Yin, and Yin Zhang. Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm. *Mathematical Programming Computation*, 4:333–361, 2012.

[102] J. Wright, A. Ganesh, K. Min, and Y. Ma. Compressive principal component pursuit. *Inform. Inference*, 2(1):32–68, 2013.

[103] Jure Zbontar, Florian Knoll, Anuroop Sriram, Matthew J. Muckley, Mary Bruno, Aaron Defazio, Marc Parente, Krzysztof J. Geras, Joe Katsnelson, Hersh Chandarana, Zizhao Zhang, Michal Drozdzal, Adriana Romero, Michael Rabbat, Pascal Vincent, James Pinkerton, Duo Wang, Nafissa Yakubova, Erich Owens, C. Lawrence Zitnick, Michael P. Recht, Daniel K. Sodickson, and Yvonne W. Lui. fastMRI: An open dataset and benchmarks for accelerated MRI. *arXiv:1811.08839*, 2018.

[104] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising. *IEEE Trans. Image Process.*, 26(7):3142–3155, 2017.

[105] H. Zhu, G. Leus, and G. B. Giannakis. Sparsity-cognizant total least-squares for perturbed compressive sampling. *IEEE Trans. Signal Process.*, 59(5):2002–2016, May 2011.