Empirical-Bayes Approaches to Recovery of Structured Sparse Signals via Approximate Message Passing

Dissertation

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

By

Jeremy Paul Vila, B.S., M.S.

Graduate Program in Electrical and Computer Engineering

The Ohio State University

2015

Dissertation Committee:

Dr. Philip Schniter, Advisor

Dr. Lee Potter

Dr. Yuejie Chi

Abstract

In recent years, there have been massive increases in both the dimensionality and sample sizes of data due to ever-increasing consumer demand coupled with relatively inexpensive sensing technologies. These high-dimensional datasets bring challenges such as complexity, along with numerous opportunities. Though many signals of interest live in a high-dimensional ambient space, they often have a much smaller inherent dimensionality which, if leveraged, lead to improved recoveries. For example, the notion of sparsity is a requisite in the compressive sensing (CS) field, which allows for accurate signal reconstruction from sub-Nyquist sampled measurements given certain conditions.

When recovering a sparse signal from noisy compressive linear measurements, the distribution of the signal's non-zero coefficients can have a profound effect on recovery mean-squared error (MSE). If this distribution is apriori known, then one could use computationally efficient approximate message passing (AMP) techniques that yield approximate minimum MSE (MMSE) estimates or critical points to the maximum a posteriori (MAP) estimation problem. In practice, though, the distribution is unknown, motivating the use of robust, convex algorithms such as LASSO–which

is nearly minimax optimal–at the cost of significantly larger MSE for non-least-favorable distributions. As an alternative, this dissertation focuses on empirical-Bayesian techniques that simultaneously learn the underlying signal distribution using the expectation-maximization (EM) algorithm while recovering the signal. These techniques are well-justified in the high-dimensional setting since, in the large system limit under specific problem conditions, the MMSE version of AMP's posteriors converge to the true posteriors and a generalization of the resulting EM procedure yields consistent parameter estimates.

Furthermore, in many practical applications, we can exploit additional signal structure beyond simple sparsity for improved MSE. In this dissertation, we investigate signals that are non-negative, obey linear equality constraints, and exhibit amplitude correlation/structured sparsity across its elements. To perform statistical inference on these structured signals, we first demonstrate how to incorporate these structures into our Bayesian model, then employ a technique called "turbo" approximate message passing on the underlying factor graph. Specifically, we partition the factor graph into the Markov and generalized linear model subgraphs, the latter of which can be efficiently implemented using approximate message passing methods, and combine the subgraphs using a "turbo" message passing approach. Numerical experiments on the compressive sensing and hyperspectral unmixing applications confirm the state-of-the-art performance of our approach, in both reconstruction error and runtime, on both synthetic and real-world datasets.

In memory of Pamela LaFrance Vila. “Vive LaFrance.”

Acknowledgments

This dissertation is indebted to the contributions of numerous people whom have helped guide me on this long journey.

First, I would like to thank my advisor, Philip Schniter, for providing the perfect atmosphere to conduct my research. Overall, I felt encouraged by his positivity, driven by his high standards, inspired by his work ethic, and illuminated by his insights. Most importantly, he recognized potential in me since the very beginning, and that confidence fueled me to where I am today. I would also like to thank my mentors at the Air Force Research Lab, Jason Parker and Joseph Meola, for all of their guidance. Certainly, my approaches would be non-existent without the groundbreaking algorithms that Jason painstakingly developed.

I am also grateful for my interactions and collaborations with my fellow Information Processing Systems lab mates. I'd like to personally thank Rohit Aggarwal, Matt Borgerding, Evan Byrne, Brian Day, Adam Rich, Michael Riedl, Yan Shou, and Justin Ziniel. When I found myself stuck on a problem, they regularly provided an outside perspective, and, more often, a much-needed distraction. I am also beholden to my sponsors and colleagues at the Center for Surveillance Research (CSR), as they provided a challenging and collaborative environment, in addition to quite literally funding this dissertation.

During my 9 years of study at The Ohio State University, I made a variety of friends whom made Columbus feel like home. In particular, I will always cherish the laughter and conversations shared with Matt Casto, Kevin Drewyor, Andrea Lotzar, Lauren Masquelier, and Dan McKeever. Most importantly, I am eternally grateful for the love and support that Diana Wieser has given me during this difficult year. She is, in absolutely every sense of the term, my better half, as she continuously motivates me to be a better person.

Lastly, I thank my family for providing me continuing encouragement and reassurance, despite the 930 mile separation: my brother Curtis, who always looked after me and I always looked up to; my sister Karrah, who is always willing to listen and say what is desperately needed; my father, who trusts my ambitions and gave me the work ethic to achieve them; my departed mother, whose undying love and positivity I attempt to mirror. I could not have done this without all of you.

Vita

December 2010	B.S. Electrical and Computer Engineering, The Ohio State University
August 2014	M.S. Electrical and Computer Engineering, The Ohio State University
2011 - 2012	University Fellow, The Ohio State University
2012 - present	Graduate Research Assistant, The Ohio State University

Publications

J. Vila and P. Schniter, "An Empirical-Bayes Approach to Recovering Linearly Constrained Non-Negative Sparse Signals," *IEEE Transactions on Signal Processing*, vol. 62, no. 18, pp. 4689-4703, Sept. 2014.

J. Vila and P. Schniter, "An Empirical-Bayes Approach to Recovering Linearly Constrained Non- Negative Sparse Signals," *Proc. IEEE Workshop Comp. Adv. Multi-Sensor Adaptive Process.*, (CAMSAP) (Saint Martin), Dec. 2013.

J. Vila and P. Schniter, "Expectation-Maximization Gaussian-Mixture Approximate Message Passing," *IEEE Transactions on Signal Processing*, vol. 61, no. 19, pp. 4658-4672, Oct. 2013.

J. Vila, P. Schniter, and J. Meola, "Hyperspectral Image Unmixing via Bilinear Generalized Approximate Message Passing," *Proc. SPIE*, (Baltimore, MD), Apr. 2013, (longer version in arXiv:1502.06435).

J. Vila and P. Schniter, "Expectation-Maximization Gaussian-Mixture Approximate Message Passing," *Proc. Conf. on Information Sciences and Systems* (Princeton, NJ), Mar. 2012.

J. Vila and P. Schniter, "Expectation-Maximization Bernoulli-Gaussian Approximate Message Passing," *Proc. Asilomar Conf. on Sig., Syst., and Comp.*, (Pacific Grove, CA), Nov. 2011.

Fields of Study

Major Field: Electrical and Computer Engineering

Specializations: Digital Signal Processing, Machine Learning

Table of Contents

Page

List of Tables

List of Figures

Figure **Page**

Chapter 1: Introduction

"Big Data" has become a popular term associated with the recent explosion of dimensionality and sample sizes of modern day datasets. As of March 2014, it was estimated that 90% of all data was generated the previous two years with a grand total of 2.5 billion gigabytes generated in 2012 alone [1]. Due to this trend, it comes as no surprise that many researchers are focused on efficient techniques for data acquisition and signal inference.

With "Big Data" comes many big challenges and insights. In his Aide-Memoir [2], renowned statistician David Donoho popularized the term "curses and blessings of dimensionality" describing this dichotomy. The main curse of dimensionality is the apparent intractability and complexity of searching through the high-dimensional space or performing integrals of high-dimensional functions, typically resulting in increased computational runtime. However, in many practical applications, the blessings of dimensionality manifest in the success of asymptotic methods, usually in a statistical framework, where rigorous analyses can be made in the large system limit.

In this dissertation, we explore the the application of a family of these methods for high-dimensional Bayesian signal inference and the task of "learning" the underlying probability distributions that describe the signal and measurement models. We then demonstrate how to perform these tasks in conjunction, resulting in a so-called

empirical-Bayes [3] technique. In particular, we employ flexible Bayesian models that leverage known low-dimensional signal structure, which is commonly exhibited in the high-dimensional inference setting. The resulting algorithms contain zero tuning parameters as they automatically tune all free parameters from realization specific data. Furthermore, they yield per-iteration computational complexities that scale linearly in each problem dimension.

1.1 Problem Statement: Structured Sparse Recovery

In this dissertation, we are interested in inferring a signal $\boldsymbol{x} \in \mathbb{R}^N$ from noisy measurements $\boldsymbol{y} \in \mathbb{R}^M$. In addition, we focus on signals that share a common low-dimensional structure, called *sparsity*, where only a fraction of the signal elements are non-zero.[1] We call a signal K-sparse if it has K non-zero elements.

In the subsequent sections, we will formalize the observation models investigated in this dissertation. As we progress, we will introduce additional examples of signal structure and methods to incorporate them into the inference problem.

1.1.1 Linear Observation Model

Formally stated, we frame the sparse signal recovery problem as the estimation of a K-sparse (or compressible) signal $\boldsymbol{x} \in \mathbb{R}^N$ from the noisy linear measurements

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{w} \in \mathbb{R}^M, \tag{1.1}$$

where $\boldsymbol{A}$ is known and $\boldsymbol{w}$ is additive white Gaussian noise (AWGN).

In traditional inverse problems, the signal is first recovered from measurements obtained from Shannon-Nyquist sampling [5](i.e., $M \geq N$)) via standard tools, e.g.,

[1] Similarly, a signal is called *compressible* if a majority of its elements are negligible in amplitude. For a formal definition of compressibility, we refer interested readers to [4].

least squares recovery. It can then be subsequently compressed, exploiting its underlying sparsity. *Compressive sensing* CS [6,7], on the other hand, attempts to recover the sparse signal $\boldsymbol{x}$ using fewer measurements than unknowns, i.e., $M < N$. Whereas signal recovery in the Shannon-Nyquist paradigm compresses *after* sampling, CS attempts to compress *while* sampling. This is particularly useful when measurement acquisition incurs a large cost (e.g., acquisition time, monetary value, availability), as exhibiting in many applications such as magnetic resonance imaging (MRI) [8] and radar imaging [9]. In CS, it is well established that accurate signal recovery is known to be possible, relative the the variance of the noise $\boldsymbol{w}$, with polynomial-complexity algorithms when $\boldsymbol{x}$ is sufficiently sparse[2] and when $\boldsymbol{A}$ satisfies certain restricted isometry properties [10] or when $\boldsymbol{A}$ has i.i.d zero-mean sub-Gaussian entries [11].

To recover a sparse $\hat{\boldsymbol{x}}$ from the CS problem (1.1), one approach is to solve

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_0 \text{ s.t. } \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \leq \epsilon, \tag{1.2}$$

where $\|\cdot\|_0$ and ϵ controls sparsity and the fit to the measurements, respectively. Unfortunately, this optimization problem is NP-hard [12], so alternative approaches need to be considered. The well-known LASSO algorithm [13] (or, equivalently, Basis Pursuit Denoising [14]) relaxes the problem (1.2) by solving the convex problem

$$\hat{\boldsymbol{x}}_{\mathsf{lasso}} = \arg\min_{\boldsymbol{x}} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda_{\mathsf{lasso}}\|\boldsymbol{x}\|_1, \tag{1.3}$$

where λ_{lasso} is a tuning parameter that serves as the tradeoff between sparsity and the "fit" to the measurements, and is, in general, difficult to optimize. A remarkable property of LASSO is that its noiseless phase transition, or, the number of measurements M needed to recover a K-sparse signal (in the large system limit for fixed

[2]Without loss of generality, we assume that $\boldsymbol{x}$ is sparse in the canonical basis, or known to have a sparse representation in a known basis $\boldsymbol{\Psi}$.

ratios M/N and K/M), is nearly minimax optimal with respect to the distribution of the non-zero signal coefficients [15]. In other words, if the vector $\boldsymbol{x}$ is drawn i.i.d from the pdf

$$p_{\mathsf{x}}(x) = \lambda p_{\mathsf{act}}(x) + (1-\lambda)\delta(x), \tag{1.4}$$

where $\delta(\cdot)$ is the Dirac delta, $p_{\mathsf{act}}(\cdot)$ is the active-coefficient pdf (with zero probability mass at $x = 0$), and $\lambda \triangleq K/N$, then the LASSO phase transition curve is invariant to $p_{\mathsf{act}}(\cdot)$. However, this implies that LASSO cannot benefit from the case that $p_{\mathsf{act}}(\cdot)$ is an "easy" distribution. For example, if we restrict $p_{\mathsf{act}}(\cdot)$ to the class of non-negative distributions, polynomial-complexity algorithms exist with PTC that are better than LASSO's [16]. On the other hand, in the rare case that the "active" signal distribution $p_{\mathsf{act}}(\cdot)$ in (1.4) is known, then MMSE-optimal recovery, though computationally intractable for large problems, yields the large-system limit PTC $\frac{K}{M} = 1$ [17].

1.1.2 Bilinear Observation Model

In many instances, we often observe a matrix of noisy measurements $\boldsymbol{Y} \in \mathbb{R}^{M\times T}$ and are asked to find a matrix-factorization, i.e., $\boldsymbol{Y} \approx \boldsymbol{A}\boldsymbol{X}$, where $\boldsymbol{A}$ and $\boldsymbol{X}$ are unknown. Applications exhibiting such factorizations are hyperspectral unmixing [18] (investigated in detail in Chapter 6), dictionary learning [19], matrix completion (MC) [20], and non-negative matrix factorizations (NNMF) [21]. These applications also share a commonality; they rely on the assumption that the factorization is *low-rank*. In many cases, they must also assume that the signal $\boldsymbol{X}$ is sufficiently sparse.

Assuming an additive noise model,[3] we state the bilinear observation model to be

$$\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{X} + \boldsymbol{W} \in \mathbb{R}^{M\times T}, \tag{1.5}$$

where the dictionary $\boldsymbol{A} \in \mathbb{R}^{M\times N}$ and the signal $\boldsymbol{X} \in \mathbb{R}^{N\times T}$ are unknown, N is the (known or hypothesized) rank, and $\boldsymbol{W}$ is noise. A straightforward approach to solving (1.5) would be to employ an alternating least-squares approach, where, given an initialization $\hat{\boldsymbol{X}}$, we iterate

$$\hat{\boldsymbol{A}} = \arg\min_{\boldsymbol{A}} \|\boldsymbol{Y} - \boldsymbol{A}\hat{\boldsymbol{X}}\|_F^2 \tag{1.6}$$

$$\hat{\boldsymbol{X}} = \arg\min_{\boldsymbol{X}} \|\boldsymbol{Y} - \hat{\boldsymbol{A}}\boldsymbol{X}\|_F^2. \tag{1.7}$$

Unfortunately, this procedure typically takes many iterations to converge, and when it does, usually at a local-minimum. Hence, the algorithms to perform the low-rank matrix-factorization task often vary with the application, often exploiting additional known signal structures or properties.

1.1.3 Additional Signal Structure

In many practical applications, the signals of interest exhibit additional structure beyond simple sparsity, where, if leveraged, leads to improved recoveries. Many signals exhibit known geometric constraints such as *non-negativity* and *linear-equality* constraints, e.g., $\boldsymbol{B}\boldsymbol{x} = \boldsymbol{c} \in \mathbb{R}^P$. Incorporating such constraints into optimization problems such as LASSO is straightforward, i.e., (1.3) transforms to

$$\hat{\boldsymbol{x}}_{\text{c-lasso}} = \arg\min_{\boldsymbol{x}\geq 0} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda_{\text{c-lasso}}\|\boldsymbol{x}\|_1 \text{ s.t. } \boldsymbol{B}\boldsymbol{x} = \boldsymbol{c}. \tag{1.8}$$

[3] The MC problem's noise model is more involved since there are many unobserved or "incomplete" entries in $\boldsymbol{Y}$.

Alternatively, Bayesian estimation approaches can employed (e.g., Gibbs sampling [22]), but they often necessitate the use of more sophisticated priors, typically resulting in additional computational burdens.

In addition to these geometrical constraints, many real-world signals feature other, less-strict types of structure. For instance, the amplitudes of the signal $\boldsymbol{X}$ in (1.5) can be correlated across the row or column elements. Incorporating such a "smoothness" structure into optimization techniques such as (1.8) often involve adding a regularization penalty, e.g., a total variation norm (akin to image denoising [23, 24]). We note that these techniques introduce yet another tuning parameter that depends on the amount of correlation, which is typically unknown and can vary from problem to problem.

The sparsity itself can also be structured. For example, the large coefficients of an image in the wavelet domain exhibit a tree structure, often referred to as a persistence across scales [25]. Incorporating *structured-sparsity* in the CS problem often leads to vastly improved performance gains in regards to the sampling rate and robustness to signal-to-noise ratio (SNR).

1.2 Proposed Approach

In the this dissertation, we develop an empirical-Bayes [3] approach to the recovery of structured-sparse signals. It is *Bayesian* in the sense that we assume families of sparsity-promoting priors on the signal, likelihoods on the observations, and other probabilistic models describing the underlying structure, which we then employ to generate maximum a posteriori (MAP) or minimum MSE (MMSE) estimates. However, since perfect prior knowledge is rarely available, we propose to "learn" the

parameters that describe these statistical models from the realization-specific data, hence the phrase *empirical*. Thus, the resulting procedure can be interpreted as a feedback mechanism where the signal estimates improve the distributions, and vice versa.

Unfortunately, for many of the signal and noise distributions we assume, exact MAP or MMSE inference is intractable. For instance, if we performed MMSE inference via loopy belief propagation (LBP) [26] on the underlying factor graph, the number of messages grow quickly with problem dimensions and the computation of the messages is intractable. Because of this, we turn to the recent approximate message passing (AMP) family of algorithms, which simultaneously simplify the form and reduce the number of messages of LBP. In particular, this dissertation focuses on three versions of AMP:

- **Bayesian AMP** [27] (AMP): performs MAP or MMSE inference of $\boldsymbol{x}$ with known, separable priors and AWGN.
- **Generalized AMP** [28] (GAMP): extends Bayesian AMP to known, separable likelihood models on the observations.
- **Bilinear GAMP** [29] (BiG-AMP): extends GAMP to MMSE inference of unknown dictionary $\boldsymbol{A}$ and signal $\boldsymbol{X}$, assuming known, separable priors.

These algorithms approximate LBP on the underlying factor graph (GAMP's factor graph is illustrated in Fig. 2.1) using central-limit-theorem and Taylor-series approximations that become exact in the large-system limit under i.i.d zero-mean sub-Gaussian $\boldsymbol{A}$.[4] Moreover, AMP and GAMP obeys a state-evolution whose fixed

[4]Certain BiG-AMP variables must also assume additional scalings in the large-system limit.

points, when unique, are optimal [30]. Since a major portion of this dissertation relies on the AMP family of algorithms, we give a brief overview of GAMP and BiG-AMP in Chapter 2.

After using AMP, GAMP, or BiG-AMP, we then employ the expectation maximization (EM) algorithm [31] to automatically tune their parameters from the data to "fit" the underlying signal distribution. Unlike many other inference procedures (e.g., LASSO), our approach contains no tuning parameters in any of the proposed approaches due to the aforementioned EM algorithm and supplemental model order selection strategies. If additional structure exists, we introduce auxiliary random variables describing this structure, then model the signal as independent conditioned on these random variables. Then, rather than performing LBP on the entire factor graph, we can leverage the "turbo" AMP framework [32] to separate the main inference sub-task (using AMP family of algorithms) from the structure-promoting sub-tasks.

The remainder of this dissertation is structured as follows. Chapter 2 gives an overview of the GAMP and BiG-AMP algorithms that are extensively used throughout the dissertation. In Chapter 3, we develop a mean-removal and adaptive-damping procedure to mitigate the occurences of GAMP divergence for non-i.i.d zero-mean sub-Gaussian $\boldsymbol{A}$ [33]. In Chapter 4, we employ our empirical-Bayes approach, EM-Gaussian-Mixture-GAMP (EM-GM-GAMP) [34], to the CS problem. In Chapter 5, we demonstrate how our approach, EM-non-negative-GAMP (EM-NN-GAMP) [35] can incorporate known non-negativity and linear equality constraints into the Bayesian context. Additionally, we demonstrate how EM-NN-GAMP can solve (1.8) exactly,

as well as update the nuisance parameter $\lambda_{\text{c-lasso}}$. In Chapter 6, we investigate the hyperspectral unmixing (HU) problem, and propose the HU-turbo-AMP (HUT-AMP) [36, 37] algorithm that exploits additional structure in the scene. To enhance the readability of this dissertation, detailed derivations and results are supplemented in the appendices. In all sections, we perform thorough numerical experiments on synthetic and real-world datasets demonstrating the state-of-the-art performance of our approaches in terms of recovery accuracy and runtime.

1.3 Notation

For matrices, we use boldface capital letters like $\boldsymbol{A}$, and we use $\boldsymbol{A}^{\mathsf{T}}$, $\boldsymbol{A}^{\mathsf{H}}$, $\mathrm{tr}(\boldsymbol{A})$, and $\|\boldsymbol{A}\|_F$ to denote the transpose, Hermitian transpose, trace, and Frobenius norm, respectively. For vectors, we use boldface small letters like $\boldsymbol{x}$, and we use $\|\boldsymbol{x}\|_p = (\sum_n |x_n|^p)^{1/p}$ to denote the ℓ_p norm, with $x_n = [\boldsymbol{x}]_n$ representing the n^{th} element of $\boldsymbol{x}$. We use $\mathbf{1}_N$ to denote the $N \times 1$ vector of ones. Deterministic quantities are denoted using serif typeface (e.g., $x, \boldsymbol{x}, \boldsymbol{X}$), while random quantities are denoted using sans-serif typeface (e.g., $\mathsf{x}, \boldsymbol{\mathsf{x}}, \boldsymbol{\mathsf{X}}$). For random variable x, we write the pdf as $p_{\mathsf{x}}(x)$, the expectation as $\mathrm{E}\{\mathsf{x}\}$, and the variance as $\mathrm{var}\{\mathsf{x}\}$. For a Gaussian random variable x with mean m and variance v, we write the pdf as $\mathcal{N}(x; m, v)$ and, for the special case of $\mathcal{N}(x; 0, 1)$, we abbreviate the pdf as $\varphi(x)$ and write the cdf and complimentary cdf as $\Phi(x)$ and $\Phi_c(x)$, respectively. Meanwhile, for a Laplacian random variable x with location m and scale v, we write the pdf as $\mathcal{L}(x; m, v)$. Finally, we use $\delta(x)$ (where $x \in \mathbb{R}$) to denote the Dirac delta distribution and δ_n (where $n \in \mathbb{Z}$) to denote the Kronecker delta sequence, and $\int_+ g(x)dx$ for the integral of $g(x)$ over $x \in [0, \infty)$.

Chapter 2: Generalized Approximate Message Passing Overview

As described in Chapter 1.2, the generalized approximate message passing algorithm [28] is an inference algorithm capable of computing either MAP or approximate-MMSE estimates of $\boldsymbol{x} \in \mathbb{R}^N$, where $\boldsymbol{x}$ is a realization of random vector x with a prior of the form (2.1), from generalized-linear observations $\boldsymbol{y} \in \mathbb{R}^M$ that yield a likelihood of the form (2.2),

$$p_{\mathsf{x}}(\boldsymbol{x}) \propto \textstyle\prod_{n=1}^{N} p_{\mathsf{x}_n}(x_n) \tag{2.1}$$

$$p_{\mathsf{y}|\mathsf{z}}(\boldsymbol{y} \,|\, \boldsymbol{A}\boldsymbol{x}) \propto \textstyle\prod_{m=1}^{M} p_{\mathsf{y}_m|\mathsf{z}_m}(y_m \,|\, [\boldsymbol{A}\boldsymbol{x}]_m), \tag{2.2}$$

where $\mathsf{z} \triangleq \boldsymbol{A}\mathsf{x}$ represents "noiseless" transform outputs.

GAMP generalizes Donoho, Maleki, and Montanari's Approximate Message Passing (AMP) algorithms [15, 16] from the case of AWGN-corrupted linear observations to the generalized-linear model (2.2). As we shall see, this generalization is useful when enforcing linear equality constraints such as $\boldsymbol{B}\boldsymbol{x} = \boldsymbol{c}$ and when formulating non-quadratic variations of (1.8).

GAMP is derived from particular approximations of loopy belief propagation (based on Taylor-series and central-limit-theorem arguments) that yield computationally simple "first-order" algorithms bearing strong similarity to primal-dual algorithms [38, 39]. Importantly, GAMP admits rigorous analysis in the large-system limit (i.e., $M, N \to \infty$ for fixed ratio M/N) under i.i.d sub-Gaussian $\boldsymbol{A}$ [28, 40], where its iterations obey a state evolution whose fixed points are optimal whenever they are unique. In this asymptotic setting, it is readily shown that sum-product GAMP's marginal posteriors converge to the true posteriors. Meanwhile, for finite-sized problems and generic $\boldsymbol{A}$, max-sum GAMP yields critical points of the MAP cost function whenever it converges,[5] whereas sum-product GAMP minimizes a certain mean-field variational objective [38] (detailed in Chapter 3). Although performance guarantees for generic finite-dimensional $\boldsymbol{A}$ are lacking except in special cases (e.g., [39]), in-depth empirical studies have demonstrated that (G)AMP performs relatively well for the $\boldsymbol{A}$ typically used in compressive sensing applications (see, e.g., [34]). An example of GAMP's factor graph is illustrated in Fig. 2.1.

Table 2.1 summarizes the GAMP algorithm. Effectively, GAMP converts the computationally intractable MAP and MMSE high-dimensional vector inference problems to a sequence of scalar inference problems. In the end, its complexity is dominated by four[6] matrix-vector multiplies per iteration: steps (R1), (R2), (R9), (R10). Furthermore, GAMP can take advantage of fast implementations of the matrix-vector multiplies (e.g., FFT) when they exist. For max-sum GAMP, scalar inference is

[5] If the MAP cost function is convex, then max-sum GAMP yields the MAP solution when it converges.

[6] Two matrix multiplies per iteration, those in (R1) and (R9), can be eliminated using the "scalar variance" modification of GAMP, with vanishing degradation in the large-system limit [28].

accomplished by lines (R3) and (R11), which involve the proximal operator

$$\text{prox}_g(\hat{v};\mu^v) \triangleq \arg\min_{x\in\mathbb{R}} g(x) + \frac{1}{2\mu^v}|x-\hat{v}|^2 \tag{2.3}$$

for generic scalar function $g(\cdot)$, as well as lines (R4) and (R12), which involve the derivative of the prox operator (2.3) with respect to its first argument. Meanwhile, for sum-product GAMP, scalar inference is accomplished by lines (R5) and (R6), which compute the mean and variance of GAMP's iteration-t approximation to the marginal posterior on z_m,

$$p_{\mathsf{z}_m|\mathsf{p}_m}(z\,|\,\hat{p}_m(t);\mu^p_m(t)) \propto p_{\mathsf{y}_m|\mathsf{z}_m}(y_m|z)\mathcal{N}(z;\hat{p}_m(t),\mu^p_m(t)), \tag{2.4}$$

and by lines (R13) and (R14), which compute the mean and variance of the GAMP-approximate marginal posterior on x_n,

$$p_{\mathsf{x}_n|\mathsf{r}_n}(x\,|\,\hat{r}_n(t);\mu^r_n(t)) \propto p_{\mathsf{x}_n}(x)\mathcal{N}(x;\hat{r}_n(t),\mu^r_n(t)). \tag{2.5}$$

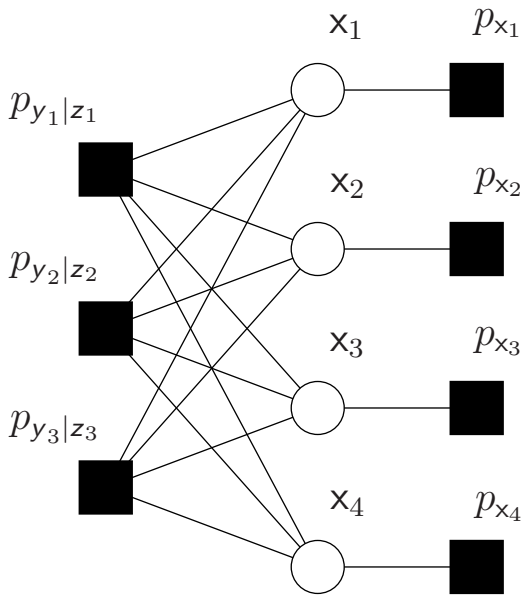


Figure 2.1: The GAMP factor graph when $N = 4$ and $M = 3$, assuming a known, separable likelihood $p_{\boldsymbol{y}|\boldsymbol{z}}(\boldsymbol{y}|\boldsymbol{z})$ and prior $p_{\boldsymbol{x}}(\boldsymbol{x})$.

We now provide background on GAMP that helps to explain (2.4)-(2.5) and Table 2.1. First and foremost, GAMP can be interpreted as an iterative thresholding algorithm, in the spirit of, e.g., [41, 42]. In particular, when the GAMP-assumed distributions are matched to the true ones, the variable $\hat{r}_n(t)$ produced in (R10) is *an approximately AWGN-corrupted version of the true coefficient* x_n (i.e., a realization of $\hat{\mathsf{r}}_n(t) = \mathsf{x}_n + \tilde{\mathsf{r}}_n(t)$ with $\tilde{\mathsf{r}}_n(t) \sim \mathcal{N}(0, \mu^r_n(t))$ independent of x_n) where $\mu^r_n(t)$ is computed in (R9) and the approximation becomes exact in the large-system limit with i.i.d sub-Gaussian $\boldsymbol{A}$ [28, 40]. Note that, under this AWGN corruption model, the pdf of x_n given $\hat{\mathsf{r}}_n(t)$ takes the form in (2.5). Thus, in sum-product mode, GAMP sets $\hat{x}_n(t+1)$ at the scalar MMSE estimate of x_n given $\hat{r}_n(t)$, as computed via the conditional mean in (R13), and it sets $\mu^x_n(t+1)$ as the corresponding MMSE, as computed via the conditional variance in (R14). Meanwhile, in max-sum mode, GAMP sets $\hat{x}_n(t+1)$ at

the scalar MAP estimate of x_n given $\hat{r}_n(t)$, as computed by the prox step in (R11), and it sets $\mu_n^x(t+1)$ in accordance with the sensitivity of this proximal thresholding, as computed in (R12). This explains (2.5) and lines (R9)-(R14) in Table 2.1.

We now provide a similar explanation for (2.4) and lines (R1)-(R6) in Table 2.1. When the GAMP distributions are matched to the true ones, $\hat{p}_m(t)$ produced in (R2) is *an approximately AWGN-corrupted version of the true transform output* z_m (i.e., a realization of $\hat{\mathsf{p}}_m(t) = \mathsf{z}_m + \tilde{\mathsf{p}}_m(t)$ with $\tilde{\mathsf{p}}_m(t) \sim \mathcal{N}(0, \mu_m^p(t))$ independent of $\hat{\mathsf{p}}_m(t)$) where $\mu_m^p(t)$ is computed in (R1) and the approximation becomes exact in the large-system limit with i.i.d sub-Gaussian $\boldsymbol{A}$ [28, 40]. Under this model, the pdf of z_m given $\hat{\mathsf{p}}_m(t)$ and y_m takes the form in (2.4). Thus, in sum-product mode, GAMP sets $\hat{z}_m(t)$ at the scalar MMSE estimate of z_m given $\hat{p}_m(t)$ and y_m, as computed via the conditional mean in (R5), and it sets $\mu_m^z(t)$ as the corresponding MMSE, as computed via the conditional variance in (R6). Meanwhile, in max-sum mode, GAMP sets $\hat{z}_m(t)$ at the scalar MAP estimate of z_m given $\hat{p}_m(t)$ and y_m, as computed by the prox operation in (R3), and it sets $\mu_m^z(t)$ in accordance with the sensitivity of this prox operation, as computed in (R4).

Indeed, what sets GAMP (and its simpler incarnation AMP) apart from other iterative thresholding algorithms is that the thresholder inputs $\hat{r}_n(t)$ and $\hat{p}_m(t)$ are (approximately) AWGN corrupted observations of x_n and z_m, respectively, ensuring that the scalar thresholding steps (R3)-(R6) and (R11)-(R14) are well justified from the MAP or MMSE perspectives. Moreover, it is the "Onsager" correction "$-\mu_m^p(t)\hat{s}_m(t-1)$" in (R2) that ensures the AWGN nature of the corruptions; without it, AMP reduces to classical iterative thresholding [16], which performs much worse [43]. Computing the Onsager correction involves (R7)-(R8). To our knowledge,

the simplest interpretation of the variables $\hat{s}_m(t)$ and $\mu_m^s(t)$ computed in (R7)-(R8) comes from primal-dual optimization theory, as established in [39]: whereas $\hat{x}_n(t)$ are estimates of the primal variables, $\hat{s}_m(t)$ are estimates of the dual variables; and whereas $\mu_n^x(t)$ relates to the primal sensitivity at the point $\hat{x}_n(t)$, $\mu_m^s(t)$ relates to the dual sensitivity at $\hat{s}_m(t)$.

inputs: $\forall m,n : p_{\mathsf{x}_n}, p_{\mathsf{y}_m|\mathsf{z}_m}, A_{mn}, T_{\max}, \epsilon_{\mathsf{gamp}} > 0, \mathsf{MaxSum} \in \{0,1\}$

definitions:

$$p_{\mathsf{z}_m|\mathsf{p}_m}(z \,|\, \hat{p}; \mu^p) \;\triangleq\; \frac{p_{\mathsf{y}_m|\mathsf{z}_m}(y_m|z)\,\mathcal{N}(z;\hat{p},\mu^p)}{\int_z p_{\mathsf{y}_m|\mathsf{z}_m}(y_m|z)\,\mathcal{N}(z;\hat{p},\mu^p)} \tag{D1}$$

$$p_{\mathsf{x}_n|\mathsf{r}_n}(x \,|\, \hat{r}; \mu^r) \;\triangleq\; \frac{p_{\mathsf{x}_n}(x)\,\mathcal{N}(x;\hat{r},\mu^r)}{\int_x p_{\mathsf{x}_n}(x)\,\mathcal{N}(x;\hat{r},\mu^r)} \tag{D2}$$

initialize:

$$\forall n : \hat{x}_n(1) = \textstyle\int_x x\, f_{\mathsf{x}_n}(x) \tag{I1}$$

$$\forall n : \mu^x_n(1) = \textstyle\int_x |x - \hat{x}_n(1)|^2 f_{\mathsf{x}_n}(x) \tag{I2}$$

$$\forall m : \hat{s}_m(0) = 0 \tag{I3}$$

for $t = 1 : T_{\max}$,

$$\forall m : \mu^p_m(t) = \textstyle\sum_{n=1}^N |A_{mn}|^2 \mu^x_n(t) \tag{R1}$$

$$\forall m : \hat{p}_m(t) = \textstyle\sum_{n=1}^N A_{mn}\hat{x}_n(t) - \mu^p_m(t)\,\hat{s}_m(t-1) \tag{R2}$$

if MaxSum then

$$\forall m : \hat{z}_m(t) = \mathrm{prox}_{-\ln p_{\mathsf{y}_m|\mathsf{z}_m}}(\hat{p}_m(t); \mu^p_m(t)) \tag{R3}$$

$$\forall m : \mu^z_m(t) = \mu^p_m(t)\,\mathrm{prox}'_{-\ln p_{\mathsf{y}_m|\mathsf{z}_m}}(\hat{p}_m(t); \mu^p_m(t)) \tag{R4}$$

else

$$\forall m : \hat{z}_m(t) = \mathrm{E}\{\mathsf{z}_m | \mathsf{p}_m = \hat{p}_m(t); \mu^p_m(t)\} \tag{R5}$$

$$\forall m : \mu^z_m(t) = \mathrm{var}\{\mathsf{z}_m | \mathsf{p}_m = \hat{p}_m(t); \mu^p_m(t)\} \tag{R6}$$

end if

$$\forall m : \mu^s_m(t) = \big(1 - \mu^z_m(t)/\mu^p_m(t)\big)/\mu^p_m(t) \tag{R7}$$

$$\forall m : \hat{s}_m(t) = \big(\hat{z}_m(t) - \hat{p}_m(t)\big)/\mu^p_m(t) \tag{R8}$$

$$\forall n : \mu^r_n(t) = \big(\textstyle\sum_{m=1}^M |A_{mn}|^2 \mu^s_m(t)\big)^{-1} \tag{R9}$$

$$\forall n : \hat{r}_n(t) = \hat{x}_n(t) + \mu^r_n(t) \textstyle\sum_{m=1}^M A^*_{mn}\hat{s}_m(t) \tag{R10}$$

if MaxSum then

$$\forall n : \hat{x}_n(t+1) = \mathrm{prox}_{-\ln p_{\mathsf{x}_n}}(\hat{r}_n(t); \mu^r_n(t)) \tag{R11}$$

$$\forall n : \mu^x_n(t+1) = \mu^r_n(t)\,\mathrm{prox}'_{-\ln p_{\mathsf{x}_n}}(\hat{r}_n(t); \mu^r_n(t)) \tag{R12}$$

else

$$\forall n : \hat{x}_n(t+1) = \mathrm{E}\{\mathsf{x}_n | \mathsf{r}_n = \hat{r}_n(t); \mu^r_n(t)\} \tag{R13}$$

$$\forall n : \mu^x_n(t+1) = \mathrm{var}\{\mathsf{x}_n | \mathsf{r}_n = \hat{r}_n(t); \mu^r_n(t)\} \tag{R14}$$

end if

$$\text{if } \textstyle\sum_{n=1}^N |\hat{x}_n(t+1) - \hat{x}_n(t)|^2 < \epsilon_{\mathsf{gamp}} \sum_{n=1}^N |\hat{x}_n(t)|^2, \text{ break} \tag{R15}$$

end

outputs: $\forall m,n : \hat{z}_m(t), \mu^z_m(t), \hat{r}_n(t), \mu^r_n(t), \hat{x}_n(t+1), \mu^x_n(t+1)$

Table 2.1: The GAMP Algorithm from [28] with max iterations $T_{\max}$ and stopping tolerance ϵ_{gamp}.

2.1 Bilinear GAMP Overview

Consider the problem of estimating the matrices $\boldsymbol{A} \in \mathbb{R}^{M\times N}$ and $\boldsymbol{X} \in \mathbb{R}^{N\times T}$ from a noisy observation $\boldsymbol{Y} \in \mathbb{R}^{M\times T}$ of the hidden bilinear form $\boldsymbol{Z} \triangleq \boldsymbol{A}\boldsymbol{X} \in \mathbb{R}^{M\times T}$. BiG-AMP treats the elements in both $\boldsymbol{A}$ and $\boldsymbol{X}$ as *independent* random variables a_{mn} and x_{nt} with known prior pdfs $p_{\mathsf{a}_{mn}}(\cdot)$ and $p_{\mathsf{x}_{nt}}(\cdot)$, respectively, with a_{mn} being zero-mean. Furthermore, it assumes that the likelihood function of $\boldsymbol{Z}$ is known and *separable*, i.e., of the form

$$p_{\boldsymbol{\mathsf{Y}}|\boldsymbol{\mathsf{Z}}}(\boldsymbol{Y}|\boldsymbol{Z}) = \prod_{m=1}^{M}\prod_{t=1}^{T} p_{\mathsf{y}_{mt}|\mathsf{z}_{mt}}(y_{mt}|z_{mt}). \tag{2.6}$$

BiG-AMP then converts the computationally intractable matrix-inference problem to a sequence of simple scalar-inference problems. In particular, it iteratively computes approximations to the marginal posterior pdfs $p_{\mathsf{a}_{mn}|\boldsymbol{\mathsf{Y}}}(\cdot|\boldsymbol{Y})$ and $p_{\mathsf{x}_{nt}|\boldsymbol{\mathsf{Y}}}(\cdot|\boldsymbol{Y})$ using an approximation of the sum-product algorithm (SPA) [44].

BiG-AMP's SPA approximation is primarily based on central-limit-theorem (CLT) and Taylor-series arguments that become exact in the large-system limit, i.e., as $M, N, T \to \infty$ with M/N and T/N converging to fixed positive constants. One important property of this approximation is that the messages from the a_{mn} nodes to the $p_{\mathsf{a}_{mn}}(a_{mn})$ nodes are Gaussian, as are those from the x_{nt} nodes to the $p_{\mathsf{x}_{nt}}(x_{nt})$ nodes. We refer the interested reader to [29] for derivation details and a full statement of the algorithm.

BiG-AMP's complexity is dominated by ten matrix multiplies (of the form $\boldsymbol{A}\boldsymbol{X}$) per iteration, although simplifications can be made in the case of additive white Gaussian $p_{\mathsf{y}_{mt}|\mathsf{z}_{mt}}(y_{mt}|z_{mt})$ that reduce the complexity to three matrix multiplies per iteration [29]. Furthermore, when BiG-AMP's likelihood function and priors include

unknown parameters $\mathbf{\Omega}$, expectation-maximization (EM) methods can be used to learn them, as described in [29].

Chapter 3: Adaptive Damping and Mean Removal for the Generalized Approximate Message Passing Algorithm

Assuming knowledge of the prior $p_{\boldsymbol{x}}(\boldsymbol{x}) = \prod_{n=1}^{N} p_{\mathsf{x}_n}(x_n)$ and likelihood $p_{\boldsymbol{y}|\boldsymbol{z}}(\boldsymbol{y}|\boldsymbol{z})$, typical estimation goals are to compute the minimum mean-squared error (MMSE) estimate $\hat{\boldsymbol{x}}_{\mathsf{MMSE}} \triangleq \int_{\mathbb{R}^N} \boldsymbol{x} p_{\boldsymbol{x}|\boldsymbol{y}}(\boldsymbol{x}|\boldsymbol{y}) d\boldsymbol{x}$ or the maximum a posteriori (MAP) estimate $\hat{\boldsymbol{x}}_{\mathsf{MAP}} \triangleq \arg\max_{\boldsymbol{x}} p_{\boldsymbol{x}|\boldsymbol{y}}(\boldsymbol{x}|\boldsymbol{y}) = \arg\min_{\boldsymbol{x}} J_{\mathsf{MAP}}(\boldsymbol{x})$ for the MAP cost

$$J_{\mathsf{MAP}}(\hat{\boldsymbol{x}}) \triangleq -\ln p_{\boldsymbol{y}|\boldsymbol{z}}(\boldsymbol{y}|\boldsymbol{A}\hat{\boldsymbol{x}}) - \ln p_{\boldsymbol{x}}(\hat{\boldsymbol{x}}). \tag{3.1}$$

As described in Chapter 2, the *generalized approximate message passing* (GAMP) algorithm [28] is a means of tackling these two problems in the case that M and N are large.[7]

GAMP is well motivated in the case that $\boldsymbol{A}$ is a realization of a large random matrix with i.i.d zero-mean sub-Gaussian entries. For such $\boldsymbol{A}$, in the large-system limit (i.e., $M, N \to \infty$ for fixed $M/N \in \mathbb{R}_+$), GAMP is characterized by a state evolution whose fixed points, when unique, are MMSE or MAP optimal [11, 28, 40]. Furthermore, for *generic* $\boldsymbol{A}$, it has been shown [38] that MAP-GAMP's fixed points coincide with the critical points of the cost function (3.1) and that MMSE-GAMP's fixed points coincide with those of a system-limit approximation of the Bethe free entropy [45], as discussed in detail in Chapter 3.1.2.

[7] This chapter is excerpted from our work developed in [33].

For generic $\boldsymbol{A}$, however, GAMP may not reach its fixed points, i.e., it may diverge (e.g., [46]). The convergence of GAMP has been fully characterized in [39] for the simple case that p_{x_n} and $p_{\mathsf{y}_m|\mathsf{z}_m}$ are Gaussian. There, it was shown that Gaussian-GAMP converges if and only if the peak-to-average ratio of the squared singular values of $\boldsymbol{A}$ is sufficiently small. A *damping* modification was then proposed in [39] that guarantees the convergence of Gaussian-GAMP with arbitrary $\boldsymbol{A}$, at the expense of a slower convergence rate. For strictly log-concave p_{x_n} and $p_{\mathsf{y}_m|\mathsf{z}_m}$, the *local* convergence of GAMP was also characterized in [39]. However, the global convergence of GAMP under generic $\boldsymbol{A}$, p_{x_n}, and $p_{\mathsf{y}_m|\mathsf{z}_m}$ is not yet understood.

Because of its practical importance, prior work has attempted to robustify the convergence of GAMP in the face of "difficult" $\boldsymbol{A}$ (e.g., high peak-to-average singular values) for generic p_{x_n} and $p_{\mathsf{y}_m|\mathsf{z}_m}$. For example, "swept" GAMP (SwAMP) [47] updates the estimates of $\{\mathsf{x}_n\}_{n=1}^N$ and $\{\mathsf{z}_m\}_{m=1}^M$ sequentially, in contrast to GAMP, which updates them in parallel. Relative to GAMP, experiments in [47] show that SwAMP is much more robust to difficult $\boldsymbol{A}$, but it is slower and cannot facilitate fast implementations of $\boldsymbol{A}$ like an FFT. As another example, the public-domain GAMPmatlab implementation [48] has included "adaptive damping" and "mean removal" mechanisms for some time, but they have never been described in the literature.

In this chapter, we detail the most recent versions of GAMPmatlab's adaptive damping and mean-removal mechanisms, and we experimentally characterize their performance on non-zero-mean, rank-deficient, column-correlated, and ill-conditioned $\boldsymbol{A}$ matrices. Our results show improved robustness relative to SwAMP and enhanced convergence speed.

3.1 Adaptively Damped GAMP

Damping is commonly used in loopy belief propagation to "slow down" the updates in an effort to promote convergence. (See, e.g., [49] for damping applied to the sum-product algorithm and [39, 48, 50] for damping applied to GAMP.) However, since not enough damping allows divergence while too much damping unnecessarily slows convergence, we are motivated to develop an *adaptive* damping scheme that applies just the right amount of damping at each iteration.

Table 3.1 details the proposed *adaptively damped GAMP* (AD-GAMP) algorithm. Lines (R3)-(R6) and (R10) use an iteration-t-dependent damping parameter $\beta(t) \in (0, 1]$ to slow the updates,[8] and lines (R12)-(R18) adapt the parameter $\beta(t)$. When $\beta(t) = 1 \; \forall t$, AD-GAMP reduces to the original GAMP from Table 2.1.

[8] The GAMPmatlab implementation [48] allows one to disable damping in (R6) and/or (R10).

definitions for MMSE-GAMP:

$$g_{\mathsf{z}_m}(\hat{p},\nu^p) \triangleq \int z\, f_{\mathsf{z}_m}(z;\hat{p},\nu^p)dz \tag{D1}$$
$$\text{for } f_{\mathsf{z}_m}(z;\hat{p},\nu^p) \triangleq \tfrac{p_{\mathsf{y}_m|\mathsf{z}_m}(y_m|z)\mathcal{N}(z;\hat{p},\nu^p)}{B_m(\hat{p},\nu^p)}$$
$$\text{and } B_m(\hat{p},\nu^p) \triangleq \int p_{\mathsf{y}_m|\mathsf{z}_m}(y_m|z)\,\mathcal{N}(z;\hat{p},\nu^p)dz$$
$$g_{\mathsf{x}_n}(\hat{r},\nu^r) \triangleq \int x\, f_{\mathsf{x}_n}(x;\hat{r},\nu^r)dx \tag{D2}$$
$$\text{for } f_{\mathsf{x}_n}(x;\hat{r},\nu^r) \triangleq \tfrac{p_{\mathsf{x}_n}(x)\mathcal{N}(x;\hat{r},\nu^r)}{C_n(\hat{r},\nu^r)}$$
$$\text{and } C_n(\hat{r},\nu^r) \triangleq \int p_{\mathsf{x}_n}(x)\,\mathcal{N}(x;\hat{r},\nu^r)dx$$

definitions for MAP-GAMP:

$$g_{\mathsf{z}_m}(\hat{p},\nu^p) \triangleq \arg\max_z \ln p_{\mathsf{y}_m|\mathsf{z}_m}(y_m|z) + \tfrac{1}{2\nu^p}|z-\hat{p}|^2 \tag{D3}$$
$$g_{\mathsf{x}_n}(\hat{r},\nu^r) \triangleq \arg\max_x \ln p_{\mathsf{x}_n}(x) + \tfrac{1}{2\nu^r}|x-\hat{r}|^2 \tag{D4}$$

inputs:

$$\forall m,n\colon g_{\mathsf{z}_m}, g_{\mathsf{x}_n}, \hat{x}_n(1), \nu^x_n(1), a_{mn}, T_{\max}\geq 1, \epsilon \geq 0$$
$$T_\beta \geq 0, \beta_{\max}\in(0,1], \beta_{\min}\in[0,\beta_{\max}], G_{\text{pass}}\geq 1, G_{\text{fail}}<1$$

initialize:

$$\forall m\colon \nu^p_m(1) = \textstyle\sum_{n=1}^N |a_{mn}|^2\nu^x_n(1), \quad \hat{p}_m(1) = \sum_{n=1}^N a_{mn}\hat{x}_n(1) \tag{I2}$$
$$J(1)=\infty,\ \beta(1)=1,\ t=1 \tag{I3}$$

while $t \leq T_{\max}$,

$$\forall m\colon \nu^z_m(t) = \nu^p_m(t)\, g'_{\mathsf{z}_m}(\hat{p}_m(t),\nu^p_m(t)) \tag{R1}$$
$$\forall m\colon \hat{z}_m(t) = g_{\mathsf{z}_m}(\hat{p}_m(t),\nu^p_m(t)) \tag{R2}$$
$$\forall m\colon \nu^s_m(t) = \beta(t)\Big(1-\tfrac{\nu^z_m(t)}{\nu^p_m(t)}\Big)\tfrac{1}{\nu^p_m(t)} + \big(1-\beta(t)\big)\nu^s_m(t-1) \tag{R3}$$
$$\forall m\colon \hat{s}_m(t) = \beta(t)\tfrac{\hat{z}_m(t)-\hat{p}_m(t)}{\nu^p_m(t)} + \big(1-\beta(t)\big)\hat{s}_m(t-1) \tag{R4}$$
$$\forall n\colon \tilde{x}_n(t) = \beta(t)\hat{x}_n(t) + \big(1-\beta(t)\big)\tilde{x}_n(t-1) \tag{R5}$$
$$\forall n\colon \nu^r_n(t) = \beta(t)\tfrac{1}{\sum_{m=1}^M |a_{mn}|^2\nu^s_m(t)} + \big(1-\beta(t)\big)\nu^r_n(t-1) \tag{R6}$$
$$\forall n\colon \hat{r}_n(t) = \tilde{x}_n(t) + \nu^r_n(t)\textstyle\sum_{m=1}^M a^{\mathsf{H}}_{mn}\hat{s}_m(t) \tag{R7}$$
$$\forall n\colon \nu^x_n(t+1) = \nu^r_n(t)\, g'_{\mathsf{x}_n}(\hat{r}_n(t),\nu^r_n(t)) \tag{R8}$$
$$\forall n\colon \hat{x}_n(t+1) = g_{\mathsf{x}_n}(\hat{r}_n(t),\nu^r_n(t)) \tag{R9}$$
$$\forall m\colon \nu^p_m(t+1) = \beta(t)\textstyle\sum_{n=1}^N |a_{mn}|^2\nu^x_n(t+1) + \big(1-\beta(t)\big)\nu^p_m(t) \tag{R10}$$
$$\forall m\colon \hat{p}_m(t+1) = \textstyle\sum_{n=1}^N a_{mn}\hat{x}_n(t+1) - \nu^p_m(t+1)\,\hat{s}_m(t) \tag{R11}$$
$$J(t+1) = \text{eqn (3.1) for MAP-GAMP or eqn (3.10) for MMSE-GAMP} \tag{R12}$$
$$\text{if}\quad J(t+1) \leq \max_{\tau=\max\{t-T_\beta,1\},\ldots,t} J(\tau) \text{ or } \beta(t)=\beta_{\min} \tag{R13}$$
$$\text{then}\quad \text{if}\quad \|\hat{\boldsymbol{x}}(t)-\hat{\boldsymbol{x}}(t+1)\|/\|\hat{\boldsymbol{x}}(t+1)\| < \epsilon, \tag{R14}$$
$$\text{then}\quad \text{stop} \tag{R15}$$
$$\text{else}\quad \beta(t+1) = \min\{\beta_{\max}, G_{\text{pass}}\beta(t)\} \tag{R16}$$
$$t = t+1 \tag{R17}$$
$$\text{else}\quad \beta(t) = \max\{\beta_{\min}, G_{\text{fail}}\beta(t)\}, \tag{R18}$$

end

outputs: $\forall m,n\colon \hat{r}_n(t), \nu^r_n(t), \hat{p}_m(t+1), \nu^p_m(t+1), \hat{x}_n(t+1), \nu^x_n(t+1)$

Table 3.1: The adaptively damped GAMP algorithm. In lines (R1) and (R8), g'_{z_m} and g'_{x_n} denote the derivatives of g_{z_m} and g_{x_n} w.r.t their first arguments.

3.1.1 Damping Adaptation

The damping adaptation mechanism in AD-GAMP works as follows. Line (R12) computes the current cost $J(t{+}1)$, as described in the sequel. Line (R13) then checks evaluates whether the current iteration "passes" or "fails": it passes if the current cost is at least as good as the worst cost over the last $T_\beta \geq 0$ iterations or if $\beta(t)$ is already at its minimum allowed value $\beta_{\min}$, else it fails. If the iteration passes, (R14)-(R15) implement a stopping condition, (R16) increases $\beta(t)$ by a factor $G_{\text{pass}} \geq 1$ (up to the maximum value $\beta_{\max}$), and (R17) increments the counter t. If the iteration fails, (R18) decreases $\beta(t)$ by a factor $G_{\text{fail}} < 1$ (down to the minimum value $\beta_{\min}$) and the counter t is *not* advanced, causing AD-GAMP to re-try the tth iteration with the new value of $\beta(t)$.

In the MAP case, line (R12) simply computes the cost $J(t{+}1) = J_{\mathsf{MAP}}(\hat{\boldsymbol{x}}(t{+}1))$ for J_{MAP} from (3.1). The MMSE case, which is more involved, will be described next.

3.1.2 MMSE-GAMP Cost Evaluation

As proven in [38] and interpreted in the context of Bethe free entropy in [45], the fixed points of MMSE-GAMP are critical points of the optimization problem

$$(f_{\boldsymbol{x}}, f_{\boldsymbol{z}}) = \underset{b_{\boldsymbol{x}}, b_{\boldsymbol{z}}}{\arg\min}\, J_{\text{Bethe}}(b_{\boldsymbol{x}}, b_{\boldsymbol{z}}) \text{ s.t. } \mathrm{E}\{\boldsymbol{z}|b_{\boldsymbol{z}}\} = \boldsymbol{A}\,\mathrm{E}\{\boldsymbol{x}|b_{\boldsymbol{x}}\} \tag{3.2}$$

$$J_{\text{Bethe}}(b_{\boldsymbol{x}}, b_{\boldsymbol{z}}) \triangleq D\Big(b_{\boldsymbol{x}} \| p_{\boldsymbol{x}}\Big) + D\Big(b_{\boldsymbol{z}} \| p_{\boldsymbol{y}|\boldsymbol{z}} Z^{-1}\Big) + H\Big(b_{\boldsymbol{z}}, \boldsymbol{\nu}^p\Big) \tag{3.3}$$

$$H\Big(b_{\boldsymbol{z}}, \boldsymbol{\nu}^p\Big) \triangleq \frac{1}{2}\sum_{m=1}^{M}\left(\frac{\mathrm{var}\{z_m|b_{z_m}\}}{\nu_m^p} + \ln 2\pi\nu_m^p\right), \tag{3.4}$$

where $b_{\boldsymbol{x}}$ and $b_{\boldsymbol{z}}$ are separable pdfs, $Z^{-1} \triangleq \int p_{\boldsymbol{y}|\boldsymbol{z}}(\boldsymbol{y}|\boldsymbol{z})d\boldsymbol{z}$ is the scaling factor that renders $p_{\boldsymbol{y}|\boldsymbol{z}}(\boldsymbol{y}|\boldsymbol{z})Z^{-1}$ a valid pdf over $\boldsymbol{z} \in \mathbb{R}^M$, $D(\cdot\|\cdot)$ denotes Kullback-Leibler (KL) divergence, and $H(b_{\boldsymbol{z}})$ is an upper bound on the entropy of $b_{\boldsymbol{z}}$ that is tight when $b_{\boldsymbol{z}}$ is

independent Gaussian with variances in $\boldsymbol{\nu}^p$. In other words, the pdfs $f_{\boldsymbol{x}}(\boldsymbol{x};\hat{\boldsymbol{r}},\boldsymbol{\nu}^{\boldsymbol{r}}) = \prod_{n=1}^N f_{\mathsf{x}_n}(x_n;\hat{r}_n,\nu_n^r)$ and $f_{\boldsymbol{z}}(\boldsymbol{z};\hat{\boldsymbol{p}},\boldsymbol{\nu}^{\boldsymbol{p}}) = \prod_{m=1}^M f_{\mathsf{z}_m}(z_m;\hat{p}_m,\nu_m^p)$ given in lines (D1) and (D2) of Table 3.1 are critical points of (3.2) for fixed-point versions of $\hat{\boldsymbol{r}},\boldsymbol{\nu}^{\boldsymbol{r}},\hat{\boldsymbol{p}},\boldsymbol{\nu}^{\boldsymbol{p}}$.

Since $f_{\boldsymbol{x}}$ and $f_{\boldsymbol{z}}$ are functions of $\hat{\boldsymbol{r}},\boldsymbol{\nu}^{\boldsymbol{r}},\hat{\boldsymbol{p}},\boldsymbol{\nu}^{\boldsymbol{p}}$, the cost J_{Bethe} can be written in terms of these quantities as well. For this, we first note

$$D\Big(f_{\mathsf{x}_n}\Big\|p_{\mathsf{x}_n}\Big) = \int f_{\mathsf{x}_n}(x;\hat{r}_n,\nu_n^r)\ln\frac{p_{\mathsf{x}_n}(x)\mathcal{N}(x;\hat{r}_n,\nu_n^r)}{p_{\mathsf{x}_n}(x)C_n(\hat{r}_n,\nu_n^r)}dx \tag{3.5}$$

$$= -\ln C_n(\hat{r}_n,\nu_n^r) - \frac{\ln 2\pi\nu_n^r}{2} - \int f_{\mathsf{x}_n}(x;\hat{r}_n,\nu_n^r)\frac{|x-\hat{r}_n|^2}{2\nu_n^r}dx \tag{3.6}$$

$$= -\ln C_n(\hat{r}_n,\nu_n^r) - \frac{\ln 2\pi\nu_n^r}{2} - \frac{|\hat{x}_n-\hat{r}_n|^2+\nu_n^x}{2\nu_n^r}, \tag{3.7}$$

where $\hat{x}_n$ and ν_n^x are the mean and variance of $f_{\mathsf{x}_n}(\cdot;\hat{r}_n,\nu_n^r)$ from (R9) and (R8). Following a similar procedure,

$$D\Big(f_{\mathsf{z}_m}\|p_{\mathsf{y}_m|\mathsf{z}_m}Z_m^{-1}\Big) = -\ln\frac{B_m(\hat{p}_m,\nu_m^p)}{Z_m} - \frac{\ln 2\pi\nu_m^p}{2} - \frac{|\hat{z}_m-\hat{p}_m|^2+\nu_m^z}{2\nu_m^p}, \tag{3.8}$$

where $\hat{z}_m$ and ν_m^z are the mean and variance of $f_{\mathsf{z}_m}(\cdot;\hat{p}_m,\nu_m^p)$ from (R2) and (R1). Then, by separability of KL divergences, we have $D(f_{\boldsymbol{x}}\|p_{\boldsymbol{x}}) = \sum_{n=1}^N D\Big(f_{\mathsf{x}_n}\|p_{\mathsf{x}_n}\Big)$ and $D(f_{\boldsymbol{z}}\|p_{\boldsymbol{y}|\boldsymbol{z}}Z^{-1}) = \sum_{m=1}^M D\Big(f_{\mathsf{z}_m}\|p_{\mathsf{y}_m|\mathsf{z}_m}Z_m^{-1}\Big)$, (3.3) and (3.4) imply

$$\begin{aligned} J_{\text{Bethe}}(\hat{\boldsymbol{r}},\boldsymbol{\nu}^r,\hat{\boldsymbol{p}},\boldsymbol{\nu}^p) = &-\sum_{m=1}^M\left(\ln B_m(\hat{p}_m,\nu_m^p) + \frac{|\hat{z}_m-\hat{p}_m|^2}{2\nu_m^p}\right) \\ &-\sum_{n=1}^N\left(\ln C_n(\hat{r}_n,\nu_n^r) + \frac{\ln\nu_n^r}{2} + \frac{\nu_n^x+|\hat{x}_n-\hat{r}_n|^2}{2\nu_n^r}\right) + \mathsf{const}, \end{aligned} \tag{3.9}$$

where we have written $J_{\text{Bethe}}(f_{\boldsymbol{x}},f_{\boldsymbol{z}})$ as "$J_{\text{Bethe}}(\hat{\boldsymbol{r}},\boldsymbol{\nu}^r,\hat{\boldsymbol{p}},\boldsymbol{\nu}^p)$" to make the $(\hat{\boldsymbol{r}},\boldsymbol{\nu}^r,\hat{\boldsymbol{p}},\boldsymbol{\nu}^p)$-dependence clear, and where const collects terms invariant to $(\hat{\boldsymbol{r}},\boldsymbol{\nu}^r,\hat{\boldsymbol{p}},\boldsymbol{\nu}^p)$.

Note that the iteration-t MMSE-GAMP cost is not obtained simply by plugging $(\hat{\boldsymbol{r}}(t),\boldsymbol{\nu}^{\boldsymbol{r}}(t),\hat{\boldsymbol{p}}(t{+}1),\boldsymbol{\nu}^{\boldsymbol{p}}(t{+}1))$ into (3.9), because the latter quantities do not necessarily

yield $(f_{\boldsymbol{x}}, f_{\boldsymbol{z}})$ satisfying the moment-matching constraint $\mathrm{E}\{\boldsymbol{z}|f_{\boldsymbol{z}}\} = \boldsymbol{A}\,\mathrm{E}\{\boldsymbol{x}|f_{\boldsymbol{x}}\}$ from (3.2). Thus, it was suggested in [45] to compute the cost as

$$J_{\mathsf{MSE}}(\hat{\boldsymbol{r}}(t), \boldsymbol{\nu}^{\boldsymbol{r}}(t)) = J_{\mathrm{Bethe}}(\hat{\boldsymbol{r}}(t), \boldsymbol{\nu}^{\boldsymbol{r}}(t), \tilde{\boldsymbol{p}}, \boldsymbol{\nu}^{\boldsymbol{p}}(t+1)), \tag{3.10}$$

for $\tilde{\boldsymbol{p}}$ chosen to match the moment-matching constraint, i.e., for

$$[\boldsymbol{A}\hat{\boldsymbol{x}}(t+1)]_m = g_{z_m}\Big(\tilde{p}_m, \nu^p_m(t+1)\Big) \text{ for } m = 1, \ldots, M \tag{3.11}$$

where $\hat{x}_n(t+1) = g_{\mathsf{x}_n}\Big(\hat{r}_n(t), \mu^r_n(t)\Big)$ for $n = 1, \ldots, N$ from (R9). Note that, since $\boldsymbol{\nu}^{\boldsymbol{p}}(t+1)$ can be computed from $(\hat{\boldsymbol{r}}(t), \boldsymbol{\nu}^{\boldsymbol{r}}(t))$ via (R8) and (R10), the left side of (3.10) uses only $(\hat{\boldsymbol{r}}(t), \boldsymbol{\nu}^{\boldsymbol{r}}(t))$.

In the case of an additive white Gaussian noise (AWGN), i.e., $p_{y_m|z_m}(y_m|z_m) = \mathcal{N}(z_m; y_m, \nu^w)$ with $\nu^w > 0$, the function $g_{z_m}(\tilde{p}_m, \nu^p_m)$ is linear in $\tilde{p}_m$. In this case, [45] showed that (3.11) can be solved in closed-form, yielding the solution

$$\tilde{p}_m = \Big((\nu^p_m(t+1) + \nu^w)[\boldsymbol{A}\hat{\boldsymbol{x}}(t+1)]_m - \nu^p_m(t+1)y_m\Big)/\nu^w. \tag{3.12}$$

For general $p_{y_m|z_m}$, however, the function $g_{z_m}(\tilde{p}_m, \nu^p_m)$ is non-linear in $\tilde{p}_m$ and difficult to invert in closed-form. Thus, we propose to solve (3.11) numerically using the regularized Newton's method detailed in Table 3.2. There, $\alpha \in (0, 1]$ is a stepsize, $\phi \geq 0$ is a regularization parameter that keeps the update's denominator positive, and $I_{\max}$ is a maximum number of iterations, all of which should be tuned in accordance with $p_{y_m|z_m}$. Meanwhile, $\tilde{p}_m(1)$ is an initialization that can be set at $\hat{p}_m(t+1)$ or $[\boldsymbol{A}\hat{\boldsymbol{x}}(t+1)]_m$ and ϵ_{inv} is a stopping tolerance. Note that the functions g_{z_m} and g'_{z_m} employed in Table 3.2 are readily available from Table 3.1.

inputs:
$g_{z_m}, [\boldsymbol{A}\hat{\boldsymbol{x}}]_m, \nu_m^p, \tilde{p}_m(1), I_{\max} \geq 1, \epsilon_{\text{inv}} \geq 0, \alpha \in (0,1], \phi \geq 0$
for $i = 1 : I_{\max}$,
$e_m(i) = [\boldsymbol{A}\hat{\boldsymbol{x}}]_m - g_{z_m}\big(\tilde{p}_m(i), \nu_m^p\big)$ (F1)
if $\big|e_m(i)/g_{z_m}\big(\tilde{p}_m(i), \nu_m^p\big)\big| < \epsilon_{\text{inv}}$, stop (F2)
$\nabla_m(i) = g'_{z_m}\big(\tilde{p}_m(i), \nu_m^p\big)$ (F3)
$\tilde{p}_m(i+1) = \tilde{p}_m(i) + \alpha \frac{e_m(i)\nabla_m(i)}{\nabla_m^2(i)+\phi}$ (F4)
end
outputs: $\tilde{p}_m(i)$

Table 3.2: A regularized Newton's method to find the value of $\tilde{p}_m$ that solves $[\boldsymbol{A}\hat{\boldsymbol{x}}]_m = g_{z_m}(\tilde{p}_m, \nu_m^p)$ for a given $[\boldsymbol{A}\hat{\boldsymbol{x}}]_m$ and ν_m^p.

3.1.3 Mean Removal

To mitigate the difficulties caused by $\boldsymbol{A}$ with non-zero mean entries, we propose to rewrite the linear system "$\boldsymbol{z} = \boldsymbol{A}\boldsymbol{x}$" as

$$\underbrace{\begin{bmatrix} \boldsymbol{z} \\ z_{M+1} \\ z_{M+2} \end{bmatrix}}_{\triangleq\, \bar{\boldsymbol{z}}} = \underbrace{\begin{bmatrix} \tilde{\boldsymbol{A}} & b_{12}\boldsymbol{\gamma} & b_{13}\mathbf{1}_M \\ b_{21}\mathbf{1}_N^{\mathsf{H}} & -b_{21}b_{12} & 0 \\ b_{31}\boldsymbol{c}^{\mathsf{H}} & 0 & -b_{31}b_{13} \end{bmatrix}}_{\triangleq\, \bar{\boldsymbol{A}}} \underbrace{\begin{bmatrix} \boldsymbol{x} \\ x_{N+1} \\ x_{N+2} \end{bmatrix}}_{\triangleq\, \bar{\boldsymbol{x}}} \tag{3.13}$$

where $(\cdot)^{\mathsf{H}}$ is conjugate transpose, $\mathbf{1}_P \triangleq [1, \ldots, 1]^{\mathsf{H}} \in \mathbb{R}^P$, and

$$\mu \triangleq \tfrac{1}{MN}\mathbf{1}_M^{\mathsf{H}}\boldsymbol{A}\mathbf{1}_N \tag{3.14}$$

$$\boldsymbol{\gamma} \triangleq \tfrac{1}{N}\boldsymbol{A}\mathbf{1}_N \tag{3.15}$$

$$\boldsymbol{c}^{\mathsf{H}} \triangleq \tfrac{1}{M}\mathbf{1}_M^{\mathsf{H}}\big(\boldsymbol{A} - \mu\mathbf{1}_M\mathbf{1}_N^{\mathsf{H}}\big) \tag{3.16}$$

$$\tilde{\boldsymbol{A}} \triangleq \boldsymbol{A} - \boldsymbol{\gamma}\mathbf{1}_N^{\mathsf{H}} - \mathbf{1}_M\boldsymbol{c}^{\mathsf{H}}. \tag{3.17}$$

The advantage of (3.13) is that the rows and columns of $\bar{\boldsymbol{A}}$ are approximately zero-mean. This can be seen by first verifying, via the definitions above, that $\boldsymbol{c}^{\mathsf{H}}\mathbf{1}_N = 0$, $\tilde{\boldsymbol{A}}\mathbf{1}_N = \mathbf{0}$, and $\mathbf{1}_M^{\mathsf{H}}\tilde{\boldsymbol{A}} = \mathbf{0}^{\mathsf{H}}$, which implies that the elements in every row and column of $\tilde{\boldsymbol{A}}$ are zero-mean. Thus, for large N and M, the elements in all but a vanishing

fraction of the rows and columns in $\bar{\boldsymbol{A}}$ will also be zero-mean. The mean-square coefficient size in the last two rows and columns of $\bar{\boldsymbol{A}}$ can be made to match that in $\tilde{\boldsymbol{A}}$ via choice of $b_{12}, b_{13}, b_{21}, b_{31}$.

To understand the construction of (3.13), note that (3.17) implies

$$\boldsymbol{z} = \boldsymbol{A}\boldsymbol{x} = \tilde{\boldsymbol{A}}\boldsymbol{x} + b_{12}\boldsymbol{\gamma}\underbrace{\mathbf{1}_N^{\mathsf{H}}\boldsymbol{x}/b_{12}}_{\triangleq\, x_{N+1}} + b_{13}\mathbf{1}_M\underbrace{\boldsymbol{c}^{\mathsf{H}}\boldsymbol{x}/b_{13}}_{\triangleq\, x_{N+2}}, \tag{3.18}$$

which explains the first M rows of (3.13). To satisfy the definitions in (3.18), we then require that $z_{M+1} = 0$ and $z_{M+2} = 0$ in (3.13), which can be ensured through the Dirac-delta likelihood

$$p_{y_m|\mathsf{z}_m}(y_m|z_m) \triangleq \delta(z_m) \quad \text{for} \quad m \in \{M+1, M+2\}. \tag{3.19}$$

Meanwhile, we make no assumption about the newly added elements x_{N+1} and x_{N+2}, and thus adopt the improper uniform prior

$$p_{\mathsf{x}_n}(x_n) \propto 1 \quad \text{for} \quad n \in \{N+1, N+2\}. \tag{3.20}$$

In summary, the mean-removal approach suggested here runs GAMP or AD-GAMP (as in Table 3.1) with $\bar{\boldsymbol{A}}$ in place of $\boldsymbol{A}$ and with the likelihoods and priors augmented by (3.19) and (3.20). It is important to note that, if multiplication by $\boldsymbol{A}$ and $\boldsymbol{A}^{\mathsf{H}}$ can be implemented using a fast transform (e.g., FFT), then multiplication by $\bar{\boldsymbol{A}}$ and $\bar{\boldsymbol{A}}^{\mathsf{H}}$ can too; for details, see the GAMPmatlab implementation [48].

3.2 Numerical Results

We numerically studied the recovery $\mathsf{NMSE} \triangleq \|\hat{\boldsymbol{x}} - \boldsymbol{x}\|^2/\|\boldsymbol{x}\|^2$ of SwAMP [47] and the MMSE version of the original GAMP from [28] relative to the proposed mean-removed (M-GAMP) and adaptively damped (AD-GAMP) modifications, as

well as their combination (MAD-GAMP). In all experiments, the signal $\boldsymbol{x}$ was drawn Bernoulli-Gaussian (BG) with sparsity rate τ and length $N=1000$, and performance was averaged over 100 realizations. Average NMSE was clipped to 0 dB for plotting purposes. The matrix $\boldsymbol{A}$ was drawn in one of four ways:

(a) **Non-zero mean**: i.i.d $\mathsf{a}_{mn} \sim \mathcal{N}(\mu, \frac{1}{N})$ for a specified $\mu \neq 0$.

(b) **Low-rank product**: $\boldsymbol{A} = \frac{1}{N}\mathbf{U}\mathbf{V}$ with $\mathbf{U} \in \mathbb{R}^{M\times R}$, $\mathbf{V} \in \mathbb{R}^{R\times N}$, and i.i.d $\mathsf{u}_{mr}, \mathsf{v}_{rn} \sim \mathcal{N}(0,1)$, for a specified R. Note $\boldsymbol{A}$ is rank deficient when $R < \min\{M, N\}$.

(c) **Column-correlated**: Rows of $\boldsymbol{A}$ are independent zero-mean stationary Gauss-Markov processes with correlation coefficient $\rho = \mathrm{E}\{\mathsf{a}_{mn}\mathsf{a}^{\mathsf{H}}_{m,n+1}\}/\mathrm{E}\{|\mathsf{a}_{mn}|^2\}$.

(d) **Ill-conditioned**: $\boldsymbol{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\mathsf{H}}$ where $\mathbf{U}$ and $\mathbf{V}^{\mathsf{H}}$ are the left and right singular vector matrices of an i.i.d $\mathcal{N}(0,1)$ matrix and $\boldsymbol{\Sigma}$ is a singular value matrix such that $[\boldsymbol{\Sigma}]_{i,i}/[\boldsymbol{\Sigma}]_{i+1,i+1} = (\kappa)^{1/\min\{M,N\}}$ for $i = 1, \ldots, \min\{M, N\}$–1, with a specified condition number $\kappa > 1$.

For all algorithms, we used $T_{\max} = 1000$ and $\epsilon = 10^{-5}$. Unless otherwise noted, for adaptive damping, we used $T_\beta = 0$, $G_{\text{pass}} = 1.1$, $G_{\text{fail}} = 0.5$, $\beta_{\max} = 1$, and $\beta_{\min} = 0.01$. For SwAMP, we used the authors' publicly available code [51].

First we experiment with CS in AWGN at $\mathsf{SNR} \triangleq \mathrm{E}\{\|\boldsymbol{z}\|^2\}/\mathrm{E}\{\|\boldsymbol{y}-\boldsymbol{z}\|^2\} = 60$ dB. For this, we used $M = 500 = N/2$ measurements and sparsity rate $\tau = 0.2$. As a reference, we compute a lower-bound on the achievable NMSE using a genie who knows the support of $\boldsymbol{x}$. For non-zero-mean matrices, Fig. 3.1(a) shows that the proposed M-GAMP and MAD-GAMP provided near-genie performance for all tested means μ. In contrast, GAMP only worked with zero-mean $\boldsymbol{A}$ and SwAMP with small-mean $\boldsymbol{A}$. For low-rank product, correlated, and ill-conditioned matrices, Fig. 3.1(b)-(d) show

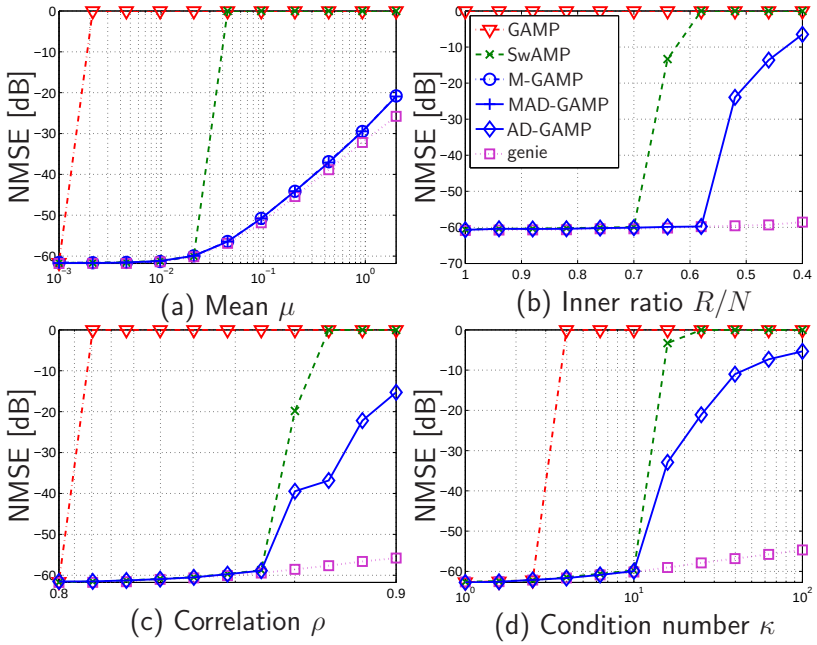


Figure 3.1: AWGN compressive sensing under (a) non-zero-mean, (b) low-rank product, (c) column-correlated, and (d) ill-conditioned $\boldsymbol{A}$.

that AD-GAMP is slightly more robust than SwAMP and significantly more robust than GAMP.

Next, we tried "robust" CS by repeating the previous experiment with sparsity rate $\tau = 0.15$ and with 10% of the observations (selected uniformly at random) replaced by "outliers" corrupted by AWGN at $\mathsf{SNR} = 0$ dB. For (M)AD-GAMP, we set $\beta_{\max} = 0.1$ and $T_{\max} = 2000$. With non-zero-mean $\boldsymbol{A}$, Fig. 3.2(a) shows increasing performance as we move from GAMP to M-GAMP to SwAMP to MAD-GAMP. For low-rank product, correlated, and ill-conditioned matrices, Fig. 3.2(b)-(d) show that SwAMP was slightly more robust than AD-GAMP, and both where much more robust than GAMP.

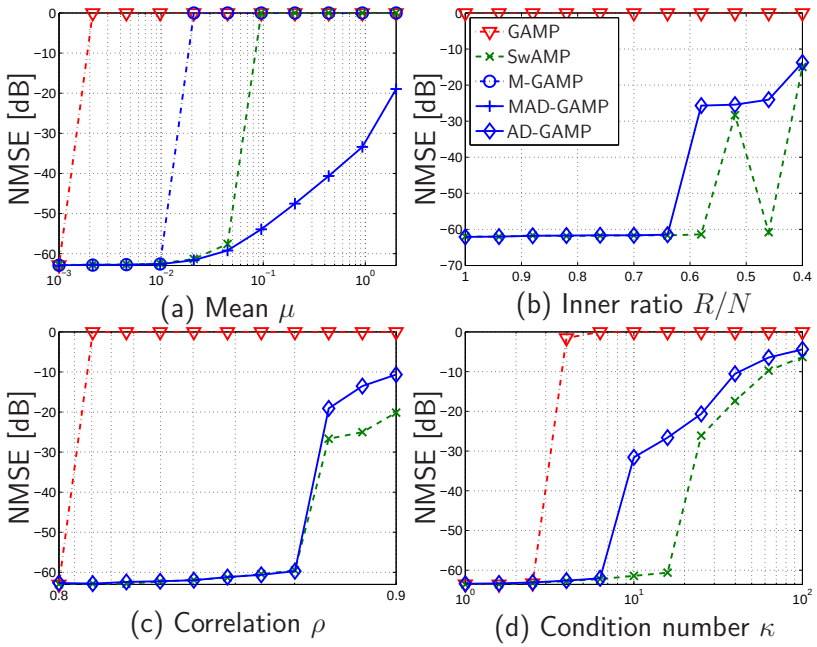


Figure 3.2: "Robust" compressive sensing under (a) non-zero-mean, (b) low-rank product, (c) column-correlated, and (d) ill-conditioned $\boldsymbol{A}$.

Next, we experimented with noiseless 1-bit CS [52], where $\boldsymbol{y} = \text{sgn}(\boldsymbol{A}\boldsymbol{x})$, using $M = 3000$ measurements and sparsity ratio $\tau = 0.125$. In each realization, the empirical mean was subtracted from the non-zero entries of $\boldsymbol{x}$ to prevent $y_m = 1\ \forall m$. For (M)AD-GAMP, we used $\beta_{\text{max}} = 0.5$. For SwAMP, we increased the stopping tolerance to $\epsilon = 5 \times 10^{-5}$, as it significantly improved runtime without degrading accuracy. For non-zero-mean $\boldsymbol{A}$, Fig. 3.3(a) shows that M-GAMP and MAD-GAMP were more robust than SwAMP, which was in turn much more robust than GAMP. For low-rank product, correlated, and ill-conditioned matrices, Fig. 3.3(b)-(d) show that MAD-GAMP and SwAMP gave similarly robust performance, while the original GAMP was very fragile.

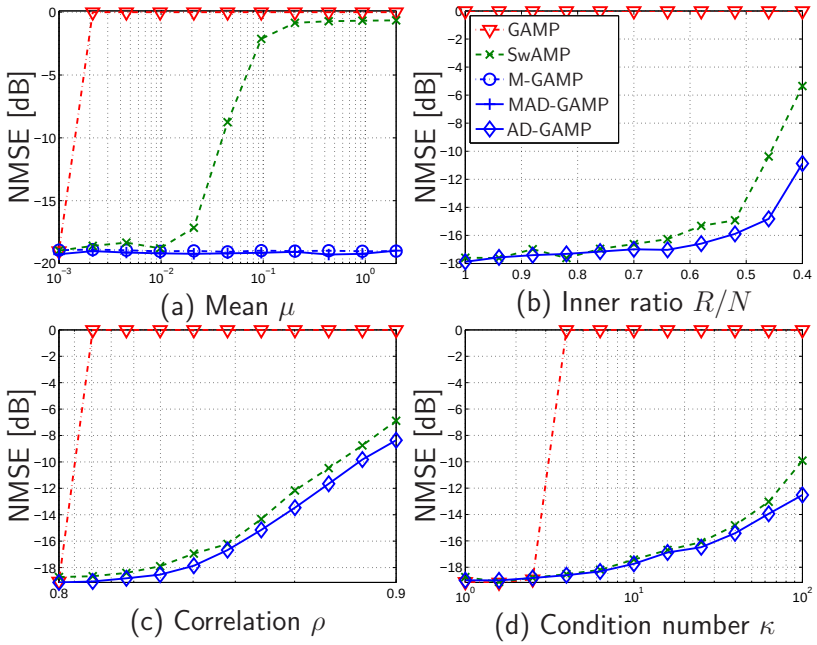


Figure 3.3: 1-bit compressive sensing under (a) non-zero-mean, (b) low-rank product, (c) column-correlated, and (d) ill-conditioned $\boldsymbol{A}$.

Finally, we compare the convergence speed of MAD-GAMP to SwAMP. For each problem, we chose a setting that allowed MAD-GAMP and SwAMP to converge for each matrix type. Table 3.3 shows that, on the whole, MAD-GAMP ran several times faster than SwAMP but used more iterations. Thus, it may be possible to reduce SwAMP's runtime to below that of MAD-GAMP using a more efficient (e.g., BLAS-based) implementation, at least for explicit $\boldsymbol{A}$. When $\boldsymbol{A}$ has a fast $O(N \log N)$ implementation (e.g., FFT), only (M)AD-GAMP will be able to exploit the reduced complexity.

		$\mu = 0.021$		$R/N = 0.64$		$\rho = 0.8$		$\log_{10}\kappa = 1$	
		MAD-GAMP	SwAMP	AD-GAMP	SwAMP	AD-GAMP	SwAMP	AD-GAMP	SwAMP
seconds	AWGN	**1.06**	1.90	**0.88**	2.74	**1.36**	3.84	**0.81**	1.49
	1-bit	**53.34**	83.21	**49.22**	137.46	**42.32**	149.40	**50.25**	117.62
	Robust	**3.47**	8.81	**2.66**	11.13	**3.33**	15.70	**2.38**	12.22
# iters	AWGN	42.9	**39.2**	130.0	**109.5**	221.9	**153.2**	121.4	**58.8**
	1-bit	947.8	**97.4**	942.7	**160.8**	866.2	**175.8**	927.3	**136.3**
	Robust	187.3	**42.2**	208.7	**56.1**	269.1	**79.2**	187.7	**61.7**

Table 3.3: Average runtime (in seconds) and # iterations of MAD-GAMP and SwAMP for various problem types and matrix types.

3.3 Conclusions

We proposed adaptive damping and mean-removal modifications of GAMP that help prevent divergence in the case of "difficult" $\boldsymbol{A}$ matrices. We then numerically demonstrated that the resulting modifications significantly increase GAMP's robustness to non-zero-mean, low-rank product, column-correlated, and ill-conditioned $\boldsymbol{A}$ matrices. Moreover, they provide robustness similar to the recently proposed SwAMP algorithm, whilerunning faster than the current SwAMP implementation. For future work, we note that the sequential update of SwAMP could in principle be combined with the proposed mean-removal and/or adaptive damping to perhaps achieve a level robustness greater than either SwAMP or (M)AD-GAMP.

Chapter 4: Expectation-Maximization Gaussian-Mixture Approximate Message Passing

4.1 Introduction

We consider estimating a K-sparse (or compressible) signal $\boldsymbol{x} \in \mathbb{R}^N$ from $M < N$ linear measurements $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{w} \in \mathbb{R}^M$, where $\boldsymbol{A}$ is known and $\boldsymbol{w}$ is additive white Gaussian noise (AWGN).[9] For this problem, accurate (relative to the noise variance) signal recovery is known to be possible with polynomial-complexity algorithms when $\boldsymbol{x}$ is sufficiently sparse and when $\boldsymbol{A}$ satisfies certain restricted isometry properties [10], or when $\boldsymbol{A}$ is large with i.i.d zero-mean sub-Gaussian entries [11] as discussed below.

LASSO [13] (or, equivalently, Basis Pursuit Denoising [14]), is a well-known approach to the sparse-signal recovery problem that solves the convex problem (1.3) with λ_{lasso} a tuning parameter that trades between the sparsity and measurement-fidelity of the solution. When $\boldsymbol{A}$ is constructed from i.i.d zero-mean sub-Gaussian entries, the performance of LASSO can be sharply characterized in the large system limit (i.e., as $K, M, N \to \infty$ with fixed undersampling ratio M/N and sparsity ratio K/M) using the so-called phase transition curve (PTC) [11, 53]. When the observations are noiseless, the PTC bisects the M/N-versus-K/M plane into the region where LASSO

[9] This chapter is excerpted from our work published in [34].

reconstructs the signal perfectly (with high probability) and the region where it does not. (See Figs. 4.3–4.5.) When the observations are noisy, the same PTC bisects the plane into the regions where LASSO's noise sensitivity (i.e., the ratio of estimation-error power to measurement-noise power under the worst-case signal distribution) is either finite or infinite [54]. An important fact about LASSO's noiseless PTC is that it is invariant to the distribution of the nonzero signal coefficients. While this implies that LASSO is robust to "difficult" instances of p_{act} in (1.4), it also implies that LASSO cannot benefit from the case that p_{act} is an "easy" distribution. For example, when the signal is known apriori to be non-negative, polynomial-complexity algorithms exist with PTCs that are better than LASSO's [16].

At the other end of the spectrum is minimum mean-squared error (MMSE)-optimal signal recovery under *known* marginal pdfs of the form (1.4) and *known* noise variance. The PTC of MMSE recovery has been recently characterized [17] and shown to be well above that of LASSO. In particular, for *any* $p_{\mathsf{act}}(\cdot)$, the PTC on the M/N-versus-K/M plane reduces to the line $K/M = 1$ in both the noiseless and noisy cases. Moreover, as described in Chapter 2, efficient algorithms for approximate MMSE-recovery have been proposed, such as the Bayesian version of Donoho, Maleki, and Montanari's *approximate message passing* (AMP) algorithm from [27], which performs loopy belief-propagation on the underlying factor graph using central-limit-theorem approximations that become exact in the large-system limit under i.i.d zero-mean sub-Gaussian $\boldsymbol{A}$. In fact, in this regime, AMP obeys [30] a state-evolution whose fixed points, when unique, are optimal. To handle arbitrary noise distributions and a wider class of matrices $\boldsymbol{A}$, Rangan proposed a *generalized AMP* (GAMP) [28]

that forms the starting point of this work. (See Table 2.1.) For more details and background on GAMP, we refer the reader to Chapter 2 and [28].

In practice, one ideally wants a recovery algorithm that does not need to know $p_{\mathsf{x}}(\cdot)$ and the noise variance a priori, yet offers performance on par with MMSE recovery, which (by definition) requires knowing these prior statistics. Towards this goal, we propose a recovery scheme that aims to *learn* the prior signal distribution $p_{\mathsf{x}}(\cdot)$, as well as the variance of the AWGN, while simultaneously recovering the signal vector $\boldsymbol{x}$ from the noisy compressed measurements $\boldsymbol{y}$. To do so, we model the active component $p_{\mathsf{act}}(\cdot)$ in (1.4) using a generic L-term Gaussian mixture (GM) and then learn the GM parameters and noise variance using the expectation-maximization (EM) algorithm [31]. As we will see, all of the quantities needed for the EM updates are already computed by the GAMP algorithm, making the overall process very computationally efficient. Moreover, GAMP provides approximately MMSE estimates of $\boldsymbol{x}$ that suffice for signal recovery, as well as posterior activity probabilities that suffice for support recovery.

Since, in our approach, the prior pdf parameters are treated as deterministic unknowns, our proposed EM-GM-GAMP algorithm can be classified as an "empirical-Bayesian" approach [3]. Compared with previously proposed empirical-Bayesian approaches to compressive sensing (e.g., [55–57]), ours has a more flexible signal model, and thus is able to better match a wide range of signal pdfs $p_{\mathsf{x}}(\cdot)$, as we demonstrate through a detailed numerical study. In addition, the complexity scaling of our algorithm is superior to that in [55–57], implying lower complexity in the high dimensional regime, as we confirm numerically. Supplemental experiments demonstrate that our excellent results hold for a wide range of sensing operators $\boldsymbol{A}$, with some exceptions.

Although this chapter does not contain any convergence guarantees or a rigorous analysis/justification of the proposed EM-GM-GAMP, Kamilov et al. showed in [58] that a generalization of EM-GM-GAMP yields asymptotically (i.e., in the large system limit) consistent parameter estimates when $\boldsymbol{A}$ is i.i.d zero-mean Gaussian, when the parameterized signal and noise distributions match the true signal and noise distributions, and when those distributions satisfy certain identifiability conditions. We refer interested readers to [58] for more details.

4.2 Gaussian-Mixture GAMP

We first introduce Gaussian-mixture (GM) GAMP, a key component of our overall approach, where the coefficients in $\boldsymbol{x} = [\mathsf{x}_1, \ldots, \mathsf{x}_N]^{\mathsf{T}}$ are assumed to be i.i.d with marginal pdf

$$p_{\mathsf{x}}(x; \lambda, \boldsymbol{\omega}, \boldsymbol{\theta}, \boldsymbol{\phi}) = (1-\lambda)\delta(x) + \lambda \sum_{\ell=1}^{L} \omega_\ell \mathcal{N}(x; \theta_\ell, \phi_\ell), \tag{4.1}$$

where $\delta(\cdot)$ is the Dirac delta, λ is the sparsity rate, and, for the k^{th} GM component, ω_k, θ_k, and ϕ_k are the weight, mean, and variance, respectively. In the sequel, we use $\boldsymbol{\omega} \triangleq [\omega_1, \ldots, \omega_L]^{\mathsf{T}}$ and similar definitions for $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$. By definition, $\sum_{\ell=1}^{L} \omega_\ell = 1$. The noise $\boldsymbol{w} = [\mathsf{w}_1, \ldots, \mathsf{w}_M]^{\mathsf{T}}$ is assumed to be i.i.d Gaussian, with mean zero and variance ψ, i.e.,

$$p_{\mathsf{w}}(w; \psi) = \mathcal{N}(w; 0, \psi), \tag{4.2}$$

and independent of $\boldsymbol{x}$. Although above and in the sequel we assume real-valued quantities, all expressions in the sequel can be converted to the circular-complex case by replacing $\mathcal{N}$ with $\mathcal{CN}$ and removing the $\frac{1}{2}$'s from (A.3), (A.14), and (A.24). We note that, from the perspective of GM-GAMP, the prior parameters $\boldsymbol{q} \triangleq [\lambda, \boldsymbol{\omega}, \boldsymbol{\theta}, \boldsymbol{\phi}, \psi]$ and the number of mixture components, L, are treated as fixed and known.

GAMP models the relationship between the m^{th} observed output y_m and the corresponding noiseless output $z_m \triangleq \boldsymbol{a}_m^{\mathsf{T}}\boldsymbol{x}$, where $\boldsymbol{a}_m^{\mathsf{T}}$ denotes the m^{th} row of $\boldsymbol{A}$, using the conditional pdf $p_{y|z}(y_m|z_m;\boldsymbol{q})$. It then approximates the true marginal posterior $p(z_m|\boldsymbol{y};\boldsymbol{q})$ by

$$p_{z|y}(z_m|\boldsymbol{y};\hat{p}_m,\mu_m^p,\boldsymbol{q}) \triangleq \frac{p_{y|z}(y_m|z_m;\boldsymbol{q})\,\mathcal{N}(z_m;\hat{p}_m,\mu_m^p)}{\int_z p_{y|z}(y_m|z;\boldsymbol{q})\,\mathcal{N}(z;\hat{p}_m,\mu_m^p)} \tag{4.3}$$

using quantities $\hat{p}_m$ and μ_m^p that change with iteration t (see Table 2.1), although here we suppress the t notation for brevity. Under the AWGN assumption[10] (4.2) we have $p_{y|z}(y|z;\boldsymbol{q}) = \mathcal{N}(y;z,\psi)$, and thus the pdf (4.3) has moments [28]

$$E_{z|y}\{z_m|\boldsymbol{y};\hat{p}_m,\mu_m^p,\boldsymbol{q}\} = \hat{p}_m + \tfrac{\mu_m^p}{\mu_m^p+\psi}(y_m-\hat{p}_m) \tag{4.4}$$

$$\mathrm{var}_{z|y}\{z_m|\boldsymbol{y};\hat{p}_m,\mu_m^p,\boldsymbol{q}\} = \frac{\mu_m^p\psi}{\mu_m^p+\psi}. \tag{4.5}$$

GAMP then approximates the true marginal posterior $p(x_n|\boldsymbol{y};\boldsymbol{q})$ by

$$p_{x|y}(x_n|\boldsymbol{y};\hat{r}_n,\mu_n^r,\boldsymbol{q}) \triangleq \frac{p_x(x_n;\boldsymbol{q})\,\mathcal{N}(x_n;\hat{r}_n,\mu_n^r)}{\int_x p_x(x;\boldsymbol{q})\,\mathcal{N}(x;\hat{r}_n,\mu_n^r)} \tag{4.6}$$

where again $\hat{r}_n$ and μ_n^r vary with the GAMP iteration t.

[10]Because GAMP can handle an arbitrary $p_{y|z}(\cdot|\cdot)$, the extension of EM-GM-GAMP to additive non-Gaussian noise, and even non-additive measurement channels (such as with quantized outputs [59] or logistic regression [28]), is straightforward. Moreover, the parameters of the pdf $p_{y|z}(\cdot|\cdot)$ could be learned using a method similar to that which we propose for learning the AWGN variance ψ, as will be evident from the derivation in Chapter 4.3.1. Finally, one could even model $p_{y|z}(\cdot|\cdot)$ as a Gaussian mixture and learn the corresponding parameters.

Plugging the sparse GM prior (4.1) into (4.6) and simplifying, one can obtain[11] the GM-GAMP approximated posterior

$$\begin{aligned} & p_{\mathsf{x}|\boldsymbol{y}}(x_n|\boldsymbol{y};\hat{r}_n,\mu_n^r,\boldsymbol{q}) \\ &= \Big((1-\lambda)\delta(x_n)+\lambda\sum_{\ell=1}^{L}\omega_\ell\mathcal{N}(x_n;\theta_\ell,\phi_\ell)\Big)\frac{\mathcal{N}(x_n;\hat{r}_n,\mu_n^r)}{\zeta_n} \qquad (4.7) \\ &= \big(1-\pi_n\big)\delta(x_n)+\pi_n\sum_{\ell=1}^{L}\overline{\beta}_{n,\ell}\mathcal{N}\Big(x_n;\gamma_{n,\ell},\nu_{n,\ell}\Big) \qquad (4.8) \end{aligned}$$

with normalization factor

$$\begin{aligned} \zeta_n &\triangleq \int_x p_{\mathsf{x}}(x;\boldsymbol{q})\,\mathcal{N}(x;\hat{r}_n,\mu_n^r) \qquad (4.9) \\ &= (1-\lambda)\mathcal{N}(0;\hat{r}_n,\mu_n^r)+\lambda\sum_{\ell=1}^{L}\omega_\ell\mathcal{N}(0;\hat{r}_n-\theta_\ell,\mu_n^r+\phi_\ell) \qquad (4.10) \end{aligned}$$

and $(\hat{r}_n,\mu_n^r,\boldsymbol{q})$-dependent quantities

$$\begin{aligned} \beta_{n,\ell} &\triangleq \lambda\omega_\ell\mathcal{N}(\hat{r}_n;\theta_\ell,\phi_\ell+\mu_n^r) \qquad (4.11) \\ \overline{\beta}_{n,\ell} &\triangleq \frac{\beta_{n,\ell}}{\sum_{k=1}^{L}\beta_{n,k}} \qquad (4.12) \\ \pi_n &\triangleq \frac{1}{1+\Big(\frac{\sum_{\ell=1}^{L}\beta_{n,\ell}}{(1-\lambda)\mathcal{N}(0;\hat{r}_n,\mu_n^r)}\Big)^{-1}} \qquad (4.13) \\ \gamma_{n,\ell} &\triangleq \frac{\hat{r}_n/\mu_n^r+\theta_\ell/\phi_\ell}{1/\mu_n^r+1/\phi_\ell} \qquad (4.14) \\ \nu_{n,\ell} &\triangleq \frac{1}{1/\mu_n^r+1/\phi_\ell}. \qquad (4.15) \end{aligned}$$

The posterior mean and variance of $p_{\mathsf{x}|\boldsymbol{y}}$ are given in steps (R13)-(R14) of Table 2.1, and (4.8) makes it clear that π_n is GM-GAMP's approximation of the posterior support probability $\Pr\{\mathsf{x}_n\neq 0\,|\,\boldsymbol{y};\boldsymbol{q}\}$.

In principle, one could specify GAMP for an arbitrary signal prior $p_{\mathsf{x}}(\cdot)$. However, if the integrals in (R9)–(R10) are not computable in closed form (e.g., when $p_{\mathsf{x}}(\cdot)$ is

[11] Both (4.8) and (4.10) can be derived from (4.7) via the Gaussian-pdf multiplication rule: $\mathcal{N}(x;a,A)\mathcal{N}(x;b,B)=\mathcal{N}(x;\frac{a/A+b/B}{1/A+1/B},\frac{1}{1/A+1/B})\mathcal{N}(0;a-b,A+B)$.

Student's-t), then they would need to be computed numerically, thereby drastically increasing the computational complexity of GAMP. In contrast, for GM signal models, we see above that all steps can be computed in closed form. Thus, a practical approach to the use of GAMP with an intractable signal prior $p_{\mathsf{x}}(\cdot)$ is to *approximate* $p_{\mathsf{x}}(\cdot)$ using an L-term GM, after which all GAMP steps can be easily implemented. The same approach could also be used to ease the implementation of intractable output priors $p_{\mathsf{y}|\mathsf{z}}(\cdot|\cdot)$.

4.3 EM Learning of the Prior Parameters

We now propose an expectation-maximization (EM) algorithm [31] to learn the prior parameters $\boldsymbol{q} \triangleq [\lambda, \boldsymbol{\omega}, \boldsymbol{\theta}, \boldsymbol{\phi}, \psi]$. The EM algorithm is an iterative technique that increases a lower bound on the likelihood $p(\boldsymbol{y}; \boldsymbol{q})$ at each iteration, thus guaranteeing that the likelihood converges to a local maximum or at least a saddle point [60]. In our case, the EM algorithm manifests as follows. Writing, for arbitrary pdf $\hat{p}(\boldsymbol{x})$,

$$\ln p(\boldsymbol{y}; \boldsymbol{q}) = \int_{\boldsymbol{x}} \hat{p}(\boldsymbol{x}) \ln p(\boldsymbol{y}; \boldsymbol{q}) \tag{4.16}$$

$$= \int_{\boldsymbol{x}} \hat{p}(\boldsymbol{x}) \ln \Big(\frac{p(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{q})}{\hat{p}(\boldsymbol{x})} \frac{\hat{p}(\boldsymbol{x})}{p(\boldsymbol{x}|\boldsymbol{y}; \boldsymbol{q})} \Big) \tag{4.17}$$

$$= \underbrace{E_{\hat{p}(\boldsymbol{x})}\{\ln p(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{q})\} + H(\hat{p})}_{\triangleq\, \mathcal{L}_{\hat{p}}(\boldsymbol{y}; \boldsymbol{q})} + \underbrace{D(\hat{p} \,\|\, p_{\boldsymbol{x}|\boldsymbol{y}}(\cdot|\boldsymbol{y}; \boldsymbol{q}))}_{\geq 0} \tag{4.18}$$

where $\mathrm{E}_{\hat{p}(\boldsymbol{x})}\{\cdot\}$ denotes expectation over $\boldsymbol{x} \sim \hat{p}(\boldsymbol{x})$, $H(\hat{p})$ denotes the entropy of pdf $\hat{p}$, and $D(\hat{p} \,\|\, p)$ denotes the Kullback-Leibler (KL) divergence between $\hat{p}$ and p. The non-negativity of the KL divergence implies that $\mathcal{L}_{\hat{p}}(\boldsymbol{y}; \boldsymbol{q})$ is a lower bound on $\ln p(\boldsymbol{y}; \boldsymbol{q})$, and thus the EM algorithm iterates over two steps: E) choosing $\hat{p}$ to maximize the lower bound for fixed $\boldsymbol{q} = \boldsymbol{q}^i$, and M) choosing $\boldsymbol{q}$ to maximize the lower bound for fixed $\hat{p} = \hat{p}^i$. For the E step, since $\mathcal{L}_{\hat{p}}(\boldsymbol{y}; \boldsymbol{q}^i) = \ln p(\boldsymbol{y}; \boldsymbol{q}^i) - D(\hat{p} \,\|\, p_{\boldsymbol{x}|\boldsymbol{y}}(\cdot|\boldsymbol{y}; \boldsymbol{q}^i))$, the

maximizing pdf would clearly be $\hat{p}^i(\boldsymbol{x}) = p_{\boldsymbol{x}|\boldsymbol{y}}(\boldsymbol{x}|\boldsymbol{y};\boldsymbol{q}^i)$, i.e., the true posterior under prior parameters $\boldsymbol{q}^i$. Then, for the M step, since $\mathcal{L}_{\hat{p}^i}(\boldsymbol{y};\boldsymbol{q}) = \mathrm{E}_{\hat{p}^i(\boldsymbol{x})}\{\ln p(\boldsymbol{x},\boldsymbol{y};\boldsymbol{q})\} + H(\hat{p}^i)$, the maximizing $\boldsymbol{q}$ would clearly be $\boldsymbol{q}^{i+1} = \arg\max_{\boldsymbol{q}} \mathrm{E}\{\ln p(\boldsymbol{x},\boldsymbol{y};\boldsymbol{q}) \,|\, \boldsymbol{y};\boldsymbol{q}^i\}$.

In our case, because the true posterior is very difficult to calculate, we instead construct our lower-bound $\mathcal{L}_{\hat{p}}(\boldsymbol{y};\boldsymbol{q})$ using the GAMP approximated posteriors, i.e., we set $\hat{p}^i(\boldsymbol{x}) = \prod_n p_{\mathsf{x}|\boldsymbol{y}}(x_n|\boldsymbol{y};\boldsymbol{q}^i)$ for $p_{\mathsf{x}|\boldsymbol{y}}$ defined in (4.6), resulting in

$$\boldsymbol{q}^{i+1} = \arg\max_{\boldsymbol{q}} \hat{\mathrm{E}}\{\ln p(\boldsymbol{x},\boldsymbol{y};\boldsymbol{q}) \,|\, \boldsymbol{y};\boldsymbol{q}^i\}, \tag{4.19}$$

where "$\hat{\mathrm{E}}$" indicates the use of the GAMP's posterior approximation. Moreover, since the joint optimization in (4.19) is difficult to perform, we update $\boldsymbol{q}$ one component at a time (while holding the others fixed), which is the well known "incremental" variant on EM from [61]. In the sequel, we use "$\boldsymbol{q}^i_{\backslash\lambda}$" to denote the vector $\boldsymbol{q}^i$ with the element λ removed (and similar for the other parameters).

4.3.1 EM Update of the Gaussian Noise Variance

We first derive the EM update for the noise variance ψ given a previous parameter estimate $\boldsymbol{q}^i$. For this, we write $p(\boldsymbol{x},\boldsymbol{y};\boldsymbol{q}) = Cp(\boldsymbol{y}|\boldsymbol{x};\psi) = C\prod_{m=1}^M p_{Y|Z}(y_m|\boldsymbol{a}_m^{\mathsf{T}}\boldsymbol{x};\psi)$ for a ψ-invariant constant C, so that

$$\psi^{i+1} = \arg\max_{\psi>0} \sum_{m=1}^M \hat{\mathrm{E}}\Big\{ \ln p_{y|z}(y_m|\boldsymbol{a}_m^{\mathsf{T}}\boldsymbol{x};\psi) \,\Big|\, \boldsymbol{y};\boldsymbol{q}^i \Big\} \tag{4.20}$$

Following Appendix A.1, we see that the noise variance update becomes

$$\psi^{i+1} = \frac{1}{M}\sum_{m=1}^M \Big(|y_m - \hat{z}_m|^2 + \mu_m^z \} \Big), \tag{4.21}$$

where the quantities $\hat{z}_m$ and μ_m^z are given in (R5)-(R6) in Table 2.1.

4.3.2 EM Updates of the Signal Parameters: BG Case

Suppose that the signal distribution $p_{\mathsf{x}}(\cdot)$ is modeled using an $L = 1$-term GM, i.e., a Bernoulli-Gaussian (BG) pdf. In this case, the marginal signal prior in (4.1) reduces to

$$p_{\mathsf{x}}(x; \lambda, \omega, \theta, \phi) = (1-\lambda)\delta(x) + \lambda \mathcal{N}(x; \theta, \phi). \tag{4.22}$$

Note that, in the BG case, the mixture weight ω is, by definition, unity and does not need to be learned.

We now derive the EM update for λ given previous parameters $\boldsymbol{q}^i \triangleq [\lambda^i, \theta^i, \phi^i, \psi^i]$. Because we can write $p(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{q}) = C \prod_{n=1}^N p_{\mathsf{x}}(x_n; \lambda, \theta, \phi)$ for a λ-invariant constant C,

$$\lambda^{i+1} = \arg\max_{\lambda \in (0,1)} \sum_{n=1}^N \hat{\mathrm{E}}\Big\{ \ln p_{\mathsf{x}}(x_n; \lambda, \boldsymbol{q}^i_{\backslash\lambda}) \,\Big|\, \boldsymbol{y}; \boldsymbol{q}^i \Big\}. \tag{4.23}$$

From Appendix A.2.1, we see that the update for the sparsity rate λ is

$$\lambda^{i+1} = \frac{1}{N} \sum_{n=1}^N \pi_n. \tag{4.24}$$

Conveniently, the posterior support probabilities $\{\pi_n\}_{n=1}^N$ are easily calculated from the GM-GAMP outputs via (4.13).

Similar to (4.23), the EM update for θ can be written as

$$\theta^{i+1} = \arg\max_{\theta \in \mathbb{R}} \sum_{n=1}^N \hat{\mathrm{E}}\Big\{ \ln p_{\mathsf{x}}(x_n; \theta, \boldsymbol{q}^i_{\backslash\theta}) \,\Big|\, \boldsymbol{y}; \boldsymbol{q}^i \Big\}. \tag{4.25}$$

Following Appendix A.2.2, the update for θ becomes

$$\theta^{i+1} = \frac{1}{\lambda^{i+1} N} \sum_{n=1}^N \pi_n \gamma_{n,1} \tag{4.26}$$

where $\{\gamma_{n,1}\}_{n=1}^N$ defined in (4.14) are easily computed from the GM-GAMP outputs.

Similar to (4.23), the EM update for ϕ can be written as

$$\hat{\phi}^{i+1} = \arg\max_{\phi > 0} \sum_{n=1}^N \hat{\mathrm{E}}\Big\{ \ln p_{\mathsf{x}}(x_n; \phi, \boldsymbol{q}^i_{\backslash\phi}) \,\Big|\, \boldsymbol{y}; \boldsymbol{q}^i \Big\}. \tag{4.27}$$

Following the procedure detailed in Appendix A.2.3, we get the update

$$\phi^{i+1} = \frac{1}{\lambda^{i+1} N} \sum_{n=1}^{N} \pi_n \left(\left| \theta^i - \gamma_{n,1} \right|^2 + \nu_{n,1} \right) \tag{4.28}$$

where $\{\nu_{n,1}\}_{n=1}^N$ from (4.15) are easily computed from the GAMP outputs.

4.3.3 EM Updates of the Signal Parameters: GM Case

We now generalize the EM updates derived in Chapter 4.3.2 to the GM prior given in (4.1) for $L \geq 1$. As we shall see, it is not possible to write the exact EM updates in closed-form when $L > 1$, and so some approximations will be made.

We begin by deriving the EM update for λ given the previous parameters $\boldsymbol{q}^i \triangleq [\lambda^i, \boldsymbol{\omega}^i, \boldsymbol{\theta}^i, \boldsymbol{\phi}^i, \psi^i]$. The first two steps are identical to the steps (4.23) and (A.5) presented for the BG case, and for brevity we do not repeat them here. In the third step, use of the GM prior (4.1) yields

$$\frac{d}{d\lambda} \ln p_{\mathsf{x}}(x_n; \lambda, \boldsymbol{q}^i_{\setminus\lambda}) = \frac{\sum_{\ell=1}^L \omega_\ell^i \mathcal{N}(x_n; \theta_\ell^i, \phi_\ell^i) - \delta(x_n)}{p_{\mathsf{x}}(x_n; \lambda, \boldsymbol{q}^i_{\setminus\lambda})} = \begin{cases} \frac{1}{\lambda} & x_n \neq 0 \\ \frac{-1}{1-\lambda} & x_n = 0 \end{cases}, \tag{4.29}$$

which coincides with the BG expression (A.6). The remaining steps also coincide with those in the BG case, and so the final EM update for λ, in the case of a GM,[12] is given by (4.24).

We next derive the EM updates for the GM parameters $\boldsymbol{\omega}, \boldsymbol{\theta}$, and $\boldsymbol{\phi}$. For each $k = 1, \ldots, L$, we incrementally update θ_k, then ϕ_k, and then the entire vector $\boldsymbol{\omega}$,

[12] The arguments in this section reveal that, under signal priors of the form $p_{\mathsf{x}}(x) = (1-\lambda)\delta(x) + \lambda p_{\mathsf{act}}(x)$, where $p_{\mathsf{act}}(\cdot)$ can be arbitrary, the EM update for λ is that given in (4.24).

while holding all other parameters fixed. The EM updates are thus

$$\theta_k^{i+1} = \arg\max_{\theta_k \in \mathbb{R}} \sum_{n=1}^{N} \hat{\mathrm{E}}\Big\{ \ln p_{\mathsf{x}}(x_n; \theta_k, \boldsymbol{q}^i_{\backslash \theta_k}) \,\Big|\, \boldsymbol{y}; \boldsymbol{q}^i \Big\}, \tag{4.30}$$

$$\phi_k^{i+1} = \arg\max_{\phi_k > 0} \sum_{n=1}^{N} \hat{\mathrm{E}}\Big\{ \ln p_{\mathsf{x}}(x_n; \phi_k, \boldsymbol{q}^i_{\backslash \phi_k}) \,\Big|\, \boldsymbol{y}; \boldsymbol{q}^i \Big\} \tag{4.31}$$

$$\boldsymbol{\omega}^{i+1} = \arg\max_{\boldsymbol{\omega} > 0: \sum_k \omega_k = 1} \sum_{n=1}^{N} \hat{\mathrm{E}}\Big\{ \ln p_{\mathsf{x}}(x_n; \boldsymbol{\omega}, \boldsymbol{q}^i_{\backslash \boldsymbol{\omega}}) \,\Big|\, \boldsymbol{y}; \boldsymbol{q}^i \Big\}. \tag{4.32}$$

Following Appendices A.3.1–A.3.3, the updates of the GM parameters are

$$\theta_k^{i+1} = \frac{\sum_{n=1}^{N} \pi_n \overline{\beta}_{n,k} \gamma_{n,k}}{\sum_{n=1}^{N} \pi_n \overline{\beta}_{n,k}} \tag{4.33}$$

$$\phi_k^{i+1} = \frac{\sum_{n=1}^{N} \pi_n \overline{\beta}_{n,k} \Big(|\theta_k^i - \gamma_{n,k}|^2 + \nu_{n,k} \Big)}{\sum_{n=1}^{N} \pi_n \overline{\beta}_{n,k}} \tag{4.34}$$

$$\omega_k^{i+1} = \frac{\sum_{n=1}^{N} \pi_n \overline{\beta}_{n,k}}{\sum_{n=1}^{N} \pi_n}, \tag{4.35}$$

respectively, where $\overline{\beta}_{n,k}, \pi_n, \gamma_{n,k}$ and $\nu_{n,k}$ are given by (4.12)–(4.15).

For sparse signals $\boldsymbol{x}$, we find that learning the GM means $\{\theta_k\}$ using the above EM procedure yields excellent recovery MSE. However, for "heavy-tailed" signals (i.e., whose pdfs have tails that are not exponentially bounded, such as Student's-t), our experience indicates that the EM-learned values of $\{\theta_k\}$ tend to gravitate towards the outliers in $\{x_n\}_{n=1}^N$, resulting in an overfitting of $p_{\mathsf{x}}(\cdot)$ and thus poor reconstruction MSE. For such heavy-tailed signals, we find that better reconstruction performance is obtained by fixing the means at zero (i.e., $\theta_k^i = 0 \; \forall k, i$). Thus, in the remainder of the chapter, we consider two modes of operation: a "sparse" mode where $\boldsymbol{\theta}$ is learned via the above EM procedure, and a "heavy-tailed" mode that fixes $\boldsymbol{\theta} = \mathbf{0}$.

Although, for the case of GM priors, approximations were used in the derivation of the EM updates (4.33), (4.34), and (4.35), it is interesting to note that, in the case of $L = 1$ mixture components, these approximate EM-GM updates coincide with the

exact EM-BG updates derived in Chapter 4.3.2. In particular, the approximate-EM update of the GM parameter θ_1 in (4.33) coincides with the exact-EM update of the BG parameter θ in (4.26), the approximate-EM update of the GM parameter ϕ_1 in (4.34) coincides with the exact-EM update of the BG parameter ϕ in (4.28), and the approximate-EM update of the GM parameter ω_1 in (4.35) reduces to the fixed value 1. Thus, one can safely use the GM updates above in the BG setting without any loss of optimality.

4.3.4 EM Initialization

Since the EM algorithm may converge to a local maximum or at least a saddle point of the likelihood function, proper initialization of the unknown parameters $\boldsymbol{q}$ is essential. Here, we propose initialization strategies for both the "sparse" and "heavy-tailed" modes of operation, for a given value of L. Regarding the value of L, we prescribe a method to learn it in Chapter 4.3.6. However, the fixed choices $L = 3$ for "sparse" mode and $L = 4$ for "heavy tailed" mode usually perform well, as shown in Chapter 4.4.

For the "sparse" mode, we set the initial sparsity rate λ^0 equal to the theoretical noiseless LASSO PTC, i.e., $\lambda^0 = \frac{M}{N}\rho_{\text{SE}}(\frac{M}{N})$, where [16]

$$\rho_{\mathsf{SE}}(\tfrac{M}{N}) = \max_{c>0} \frac{1 - \frac{2N}{M}[(1+c^2)\Phi(-c) - c\phi(c)]}{1 + c^2 - 2[(1+c^2)\Phi(-c) - c\phi(c)]} \tag{4.36}$$

describes the maximum value of $\frac{K}{M}$ supported by LASSO for a given $\frac{M}{N}$, and where $\Phi(\cdot)$ and $\phi(\cdot)$ denote the cdf and pdf of the $\mathcal{N}(0,1)$ distribution, respectively. Using the energies $||\boldsymbol{y}||_2^2$ and $||\boldsymbol{A}||_F^2$ and an assumed value of SNR^0, we initialize the noise and signal variances, respectively, as

$$\psi^0 = \frac{\|\boldsymbol{y}\|_2^2}{(\mathsf{SNR}^0 + 1)M}, \;\; \varphi^0 = \frac{\|\boldsymbol{y}\|_2^2 - M\psi^0}{||\boldsymbol{A}||_F^2 \lambda^0}, \tag{4.37}$$

where, in the absence of (user provided) knowledge about the $\mathsf{SNR} \triangleq \|\boldsymbol{A}\boldsymbol{x}\|_2^2/\|\boldsymbol{w}\|_2^2$, we suggest $\mathsf{SNR}^0 = 100$, because in our experience this value works well over a wide range of true SNR. Then, we uniformly space the initial GM means $\boldsymbol{\theta}^0$ over $[\frac{-L+1}{2L}, \frac{L-1}{2L}]$, and subsequently fit the mixture weights $\boldsymbol{\omega}^0$ and variances $\boldsymbol{\phi}^0$ to the uniform pdf supported on $[-0.5, 0.5]$ (which can be done offline using the standard approach to EM-fitting of GM parameters, e.g., [22, p. 435]). Finally, we multiply $\boldsymbol{\theta}^0$ by $\sqrt{12\varphi^0}$ and $\boldsymbol{\phi}^0$ by $12\varphi^0$ to ensure that the resulting signal variance equals φ^0.

For the "heavy-tailed" mode, we initialize λ^0 and ψ^0 as above and set, for $k = 1, \ldots, L$,

$$\omega_k^0 = \frac{1}{L}, \ \phi_k^0 = \frac{k}{\sqrt{L}} \frac{(\|\boldsymbol{y}\|_2^2 - M\psi^0)}{\|\boldsymbol{A}\|_F^2 \lambda^0}, \text{ and } \theta_k^0 = 0. \tag{4.38}$$

4.3.5 EM-GM-GAMP Summary and Demonstration

The fixed-L EM-GM-GAMP[13] algorithm developed in the previous sections is summarized in Table 4.1. For EM-BG-GAMP (as previously described in [62]), one would simply run EM-GM-GAMP with $L = 1$.

To demonstrate EM-GM-GAMP's ability to learn the underlying signal distribution, Fig. 4.1 shows examples of the GM-modeled signal distributions learned by EM-GM-GAMP in both "sparse" and "heavy-tailed" modes. To create the figure, we first constructed the true signal vector $\boldsymbol{x} \in \mathbb{R}^N$ using $N = 2000$ independent draws of the true distribution $p_{\mathsf{x}}(\cdot)$ shown in each of the subplots. Then, we constructed measurements $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{w}$ by drawing $\boldsymbol{A} \in \mathbb{R}^{M \times N}$ with i.i.d $\mathcal{N}(0, M^{-1})$ elements and $\boldsymbol{w} \in \mathbb{R}^M$ with i.i.d $\mathcal{N}(0, \sigma^2)$ elements, with $M = 1000$ and σ^2 chosen to achieve $\mathsf{SNR} = 25$ dB. Finally, we ran EM-GM-GAMP according to Table 4.1,

[13] Matlab code at http://www.ece.osu.edu/~schniter/EMturboGAMP.

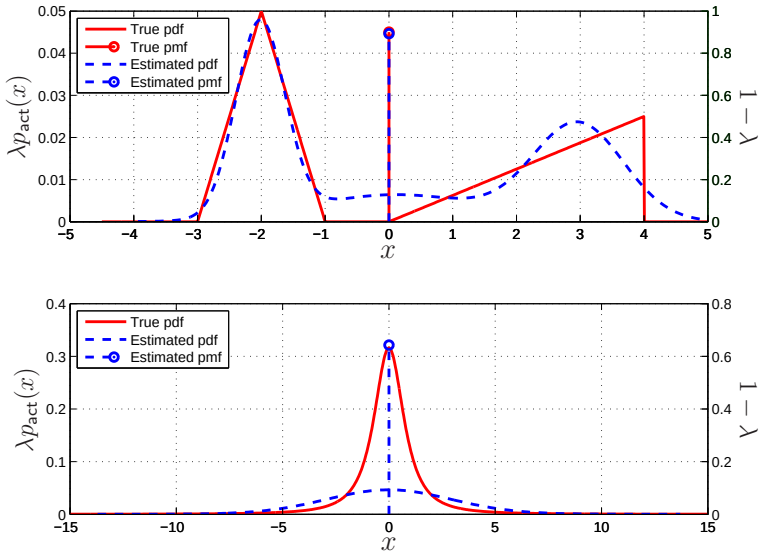


Figure 4.1: True and EM-GM-GAMP-learned versions of the signal distribution $p_{\mathsf{x}}(x) = \lambda p_{\mathsf{act}}(x) + (1-\lambda)\delta(x)$. The top subplot shows "sparse" mode EM-GM-GAMP run using GM-order $L = 3$ on a sparse signal whose non-zero components were generated according to a triangular mixture, whereas the bottom subplot shows "heavy-tailed" EM-GM-GAMP run using $L = 4$ on a Student's-t signal with rate parameter $q = 1.67$ (defined in (4.48)). The density of the continuous component $\lambda p_{\mathsf{act}}(x)$ is marked on the left axis, while the mass of the discrete component $(1-\lambda)\delta(x)$ is marked on the right axis.

and plotted the GM approximation $p_{\mathsf{x}}(x; \boldsymbol{q}^i)$ from (4.1) using the learned pdf parameters $\boldsymbol{q}^i = [\lambda^i, \boldsymbol{\omega}^i, \boldsymbol{\theta}^i, \boldsymbol{\phi}^i, \psi^i]$. Figure 4.1 confirms that EM-GM-GAMP is successful in learning a reasonable approximation of the unknown true pdf $p_{\mathsf{x}}(\cdot)$ from the noisy compressed observations $\boldsymbol{y}$, in both sparse and heavy-tailed modes.

Initialize L and $\boldsymbol{q}^0$ as described in Chapter 4.3.4.
Initialize $\hat{\boldsymbol{x}}^0 = \mathbf{0}$.
for $i = 1$ to $I_{\max}$ **do**
  Generate $\hat{\boldsymbol{x}}^i$, $\hat{\boldsymbol{z}}^i$, $(\boldsymbol{\mu}^z)^i$, $\boldsymbol{\pi}^i$, $\{\boldsymbol{\beta}_k^i, \boldsymbol{\gamma}_k^i, \boldsymbol{\nu}_k^i\}_{k=1}^L$ using GM-GAMP with $\boldsymbol{q}^{i-1}$ (see Table 2.1).
  if $\|\hat{\boldsymbol{x}}^i - \hat{\boldsymbol{x}}^{i-1}\|_2^2 < \tau_{\mathsf{em}} \|\hat{\boldsymbol{x}}^{i-1}\|_2^2$ **then**
    break.
  end if
  Compute λ^i from $\boldsymbol{\pi}^{i-1}$ as described in (4.24).
  for $k = 1$ to L **do**
    if sparse mode enabled **then**
      Compute θ_k^i from $\boldsymbol{\pi}^{i-1}$, $\boldsymbol{\gamma}_k^{i-1}$, $\{\boldsymbol{\beta}_l^{i-1}\}_{l=1}^L$ as described in (4.33).
    else if heavy-tailed mode enabled **then**
      Set $\theta_k^i = 0$.
    end if
    Compute ϕ_k^i from θ_k^{i-1}, $\boldsymbol{\pi}^{i-1}$, $\boldsymbol{\gamma}_k^{i-1}$, $\boldsymbol{\nu}_k^{i-1}$, $\{\boldsymbol{\beta}_l^{i-1}\}_{l=1}^L$ as described in (4.34).
    Compute $\boldsymbol{\omega}^i$ from $\boldsymbol{\pi}^{i-1}$ and $\{\boldsymbol{\beta}_l^{i-1}\}_{l=1}^L$ as described in (4.35).
  end for
  Compute ψ^i from $\hat{\boldsymbol{z}}^i$ and $(\boldsymbol{\mu}^z)^i$ as in (4.21).
end for

Table 4.1: The EM-GM-GAMP algorithm (fixed-L case)

4.3.6 Selection of GM Model Order

We now propose a method to learn the number of GM components, L, based on standard maximum likelihood (ML)-based model-order-selection methodology [63], i.e.,

$$\hat{L} = \underset{L \in \mathbb{Z}^+}{\arg\max} \ \ln p(\boldsymbol{y}; \hat{\boldsymbol{q}}_L) - \eta(L), \tag{4.39}$$

where $\hat{\boldsymbol{q}}_L$ is the ML estimate of $\boldsymbol{q}$ under the hypothesis L and $\eta(L)$ is a penalty term. For $\eta(L)$, there are several possibilities, but we focus on the Bayesian information criterion (BIC) [63]:

$$\eta_{\mathsf{BIC}}(L) = |\hat{\boldsymbol{q}}_L| \ln U, \tag{4.40}$$

where $|\hat{\boldsymbol{q}}_L|$ denotes the number[14] of real-valued parameters affected by L, and U is the sample size (see below).

Because $\ln p(\boldsymbol{y};\hat{\boldsymbol{q}}_L)$ is difficult to evaluate, we work with the lower bound (where for now L^j, $\hat{\boldsymbol{q}}_L$, and $\hat{\boldsymbol{q}}_{L^j}$ are arbitrary)

$$\ln p(\boldsymbol{y};\hat{\boldsymbol{q}}_L) = \ln \int_{\boldsymbol{x}} p(\boldsymbol{x}|\boldsymbol{y};\hat{\boldsymbol{q}}_{L^j}) \frac{p(\boldsymbol{x},\boldsymbol{y};\hat{\boldsymbol{q}}_L)}{p(\boldsymbol{x}|\boldsymbol{y};\hat{\boldsymbol{q}}_{L^j})} \tag{4.41}$$

$$\geq \int_{\boldsymbol{x}} p(\boldsymbol{x}|\boldsymbol{y};\hat{\boldsymbol{q}}_{L^j}) \ln \frac{p(\boldsymbol{x},\boldsymbol{y};\hat{\boldsymbol{q}}_L)}{p(\boldsymbol{x}|\boldsymbol{y};\hat{\boldsymbol{q}}_{L^j})} \tag{4.42}$$

$$= \int_{\boldsymbol{x}} p(\boldsymbol{x}|\boldsymbol{y};\hat{\boldsymbol{q}}_{L^j}) \ln p(\boldsymbol{x},\boldsymbol{y};\hat{\boldsymbol{q}}_L) + \text{const} \tag{4.43}$$

$$= \sum_{n=1}^{N} \int_{x_n} p(x_n|\boldsymbol{y};\hat{\boldsymbol{q}}_{L^j}) \ln p_{\mathsf{x}}(x_n;\hat{\boldsymbol{q}}_L) + \text{const} \tag{4.44}$$

$$= \underbrace{\sum_{n=1}^{N} \int_{x_n \neq 0} p(x_n|\boldsymbol{y};\hat{\boldsymbol{q}}_{L^j}) \ln p_{\mathsf{act}}(x_n;\hat{\boldsymbol{q}}_L)}_{\triangleq\, \mathcal{L}_{L^j}(\boldsymbol{y};\hat{\boldsymbol{q}}_L)} + \text{const}, \tag{4.45}$$

where (4.42) applies Jensen's inequality, "const" denotes a constant term w.r.t L, and (4.44) holds because $\ln p(\boldsymbol{x},\boldsymbol{y};\hat{\boldsymbol{q}}_L) = \ln p(\boldsymbol{x};\hat{\boldsymbol{q}}_L) + \ln p(\boldsymbol{y}|\boldsymbol{x};\hat{\psi}) = \sum_{n=1}^{N} \ln p_{\mathsf{x}}(x_n;\hat{\boldsymbol{q}}_L) +$ const. Equation (4.45) can then be obtained integrating (4.44) separately over $\mathcal{B}_\epsilon$ and $\overline{\mathcal{B}_\epsilon}$ and taking $\epsilon \to 0$, as done several times in Chapter 4.3.2. Using this lower bound in place of $\ln p(\boldsymbol{y};\hat{\boldsymbol{q}}_L)$ in (4.39), we obtain the BIC-inspired model order estimate (where now $\hat{\boldsymbol{q}}_L$ is specifically the ML estimate of $\boldsymbol{q}_L$)

$$L^{j+1} \triangleq \underset{L\in\mathbb{Z}^+}{\arg\max}\ \mathcal{L}_{L^j}(\boldsymbol{y};\hat{\boldsymbol{q}}_L) - \eta_{\mathsf{BIC}}(L). \tag{4.46}$$

We in fact propose to perform (4.46) iteratively, with $j = 0, 1, 2, \ldots$ denoting the iteration index. Notice that (4.46) can be interpreted as a "penalized" EM update

[14] In our case, the parameters affected by L are the GM means, variances, and weights, so that, for real-valued signals, we use $|\hat{\boldsymbol{q}}_L| = 3L - 1$ in "sparse" mode and $|\hat{\boldsymbol{q}}_L| = 2L - 1$ in heavy-tailed mode, and for complex-valued signals, we use $|\hat{\boldsymbol{q}}_L| = 4L - 1$ in "sparse" mode and $|\hat{\boldsymbol{q}}_L| = 2L - 1$ in heavy-tailed mode.

for L; if we neglect the penalty term $\eta(L)$, then (4.41)-(4.45) becomes a standard derivation for the EM-update of L (recall, e.g., the EM derivation in Chapter 4.3). The penalty term is essential, though, because the unpenalized log-likelihood lower bound $\mathcal{L}_{L^j}(\boldsymbol{y};\hat{\boldsymbol{q}}_L)$ is non-decreasing[15] in L.

We now discuss several practical aspects of our procedure. First, we are forced to approximate the integral in (4.45). To start, we use GM-GAMP's approximation of the posterior $p(x_n|\boldsymbol{y};\hat{\boldsymbol{q}}_{L^j})$ from (4.7), and the EM approximations of the ML-estimates $\hat{\boldsymbol{q}}_{L^j}$ and $\hat{\boldsymbol{q}}_L$ outlined in Chapter 4.3.3. In this case, the integral in (4.45) takes the form

$$\int_{x_n} \pi_n \sum_{l=1}^{L^j} \overline{\beta}_{n,l} \mathcal{N}(x_n;\gamma_{n,l},\nu_{n,l}) \ln \sum_{k=1}^{L} \omega_k \mathcal{N}(x_n;\theta_k,\phi_k) \tag{4.47}$$

which is still difficult due to the log term. Hence, we evaluate (4.47) using the point-mass approximation $\mathcal{N}(x_n;\gamma_{n,l},\nu_{n,l}) \approx \delta(x_n - \gamma_{n,l})$. Second, for the BIC penalty (4.40), we use the sample size $U = \sum_{n=1}^{N} \pi_n$, which is the effective number of terms in the sum in (4.45). Third, when maximizing L over $\mathbb{Z}^+$ in (4.46), we start with $L = 1$ and increment L in steps of one until the penalized metric decreases. Fourth, for the initial model order L^0, we recommend using $L^0 = 3$ in "sparse" mode and $L^0 = 4$ in "heavy-tailed" mode, i.e., the fixed-L defaults from Chapter 4.3.4. Finally, (4.46) is iterated until either $L^{j+1} = L^j$ or a predetermined maximum number of allowed model-order iterations $J_{\max}$ has been reached.

As a demonstration of the proposed model-order selection procedure, we estimated a realization of $\boldsymbol{x}$ with $N = 1000$ coefficients drawn i.i.d from the triangular mixture pdf shown in Fig. 4.1 (top, red) with $\lambda = 0.1$, from the $M = 500$ noisy measurements

[15]Note that $\mathcal{L}_{L^j}(\boldsymbol{y};\hat{\boldsymbol{q}}_L)$ can be written as a constant plus a scaled value of the negative KL divergence between $p(\boldsymbol{x} \,|\, \boldsymbol{x}\neq\boldsymbol{0}, \boldsymbol{y};\hat{\boldsymbol{q}}_{L^j})$ and the GMM $p_{\mathsf{act}}(\boldsymbol{x};\hat{\boldsymbol{q}}_L)$, where the KL divergence is clearly non-increasing in L.

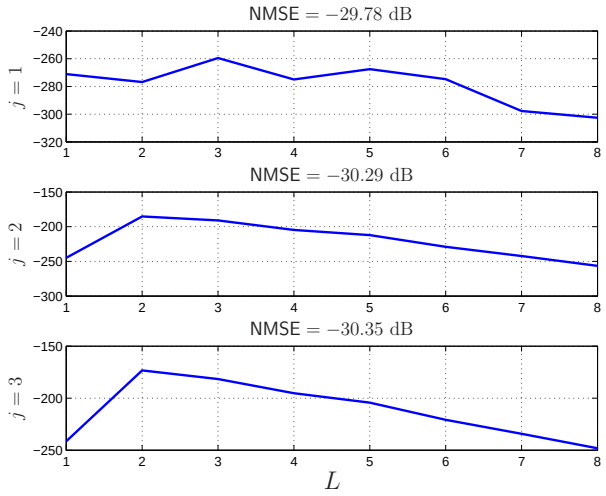


Figure 4.2: An example of the model-order metric in (4.46) over several iterations $j = 1, 2, 3$ using initial model-order $L^j|_{j=0} = 1$, together with the NMSE of the resulting estimates.

$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{w}$, where $\boldsymbol{A}$ was drawn i.i.d $\mathcal{N}(0, M^{-1})$, and $\boldsymbol{w}$ was AWGN such that SNR $= 20$ dB. For illustrative purposes, we set the initial model order at $L^0 = 1$. Iteration $j = 1$ yielded the metric $\mathcal{L}_{L^j}(\boldsymbol{y}; \hat{\boldsymbol{q}}_L) - \eta_{\mathsf{BIC}}(L)$ shown at the top of Fig. 4.2, which was maximized by $L = 3 \triangleq L^1$. The metric resulting from iteration $j = 2$ is shown in the middle of Fig. 4.2, which was maximized by $L = 2 \triangleq L^2$. At iteration $j = 3$, we obtained the metric at the bottom of Fig. 4.2, which is also maximized by $L = 2 \triangleq L^3$. Since $L^3 = L^2$, the algorithm terminates with final model order estimate $L = 2$. Figure 4.2 also indicates the per-iteration MSE, which is best at the final model order.

4.4 Numerical Results

In this section we report the results of a detailed numerical study that investigate the performance of EM-GM-GAMP under both noiseless and noisy settings. For all experiments, we set the GM-GAMP tolerance to $\tau_{\mathsf{gamp}} = 10^{-5}$ and the maximum

GAMP-iterations to $T_{\max} = 20$ (recall Table 2.1), and we set the EM tolerance to $\tau_{\mathsf{em}} = 10^{-5}$ and the maximum EM-iterations to $I_{\max} = 20$ (recall Table 4.1). For fixed-L EM-GM-GAMP, we set $L = 3$ in "sparse" and $L = 4$ in "heavy-tailed" modes.

4.4.1 Noiseless Phase Transitions

We first describe the results of experiments that computed noiseless empirical phase transition curves (PTCs) under three sparse-signal distributions. To evaluate each empirical PTC, we fixed $N = 1000$ and constructed a 30×30 grid where (M, K) were chosen to yield a uniform sampling of oversampling ratios $\frac{M}{N} \in [0.05, 0.95]$ and sparsity ratios $\frac{K}{M} \in [0.05, 0.95]$. At each grid point, we generated $R = 100$ independent realizations of a K-sparse signal $\boldsymbol{x}$ from a specified distribution and an $M \times N$ measurement matrix $\boldsymbol{A}$ with i.i.d $\mathcal{N}(0, M^{-1})$ entries. From the noiseless measurements $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$, we recovered the signal $\boldsymbol{x}$ using several algorithms. A recovery $\hat{\boldsymbol{x}}$ from realization $r \in \{1, \ldots, R\}$ was defined a success if the $\mathsf{NMSE} \triangleq \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2^2 / \|\boldsymbol{x}\|_2^2 < 10^{-6}$, and the average success rate was defined as $\overline{S} \triangleq \frac{1}{R} \sum_{r=1}^{R} S_r$, where $S_r = 1$ for a success and $S_r = 0$ otherwise. The empirical PTC was then plotted, using Matlab's `contour` command, as the $\overline{S} = 0.5$ contour over the sparsity-undersampling grid.

Figures 4.3–4.5 show the empirical PTCs for five recovery algorithms: the proposed EM-GM-GAMP algorithm (in "sparse" mode) for both L fixed and L learned through model-order selection (MOS), the proposed EM-BG-GAMP algorithm, a genie-tuned[16] GM-GAMP that uses the true parameters $\boldsymbol{q} = [\lambda, \boldsymbol{\omega}, \boldsymbol{\theta}, \boldsymbol{\phi}, \psi]$, and the Donoho, Maleki, Montanari (DMM) LASSO-style AMP from [16]. For comparison,

[16]For genie-tuned GM-GAMP, for numerical reasons, we set the noise variance at $\psi = 10^{-6}$ and, with Bernoulli and BR signals, the mixture variances at $\phi_k = 10^{-2}$.

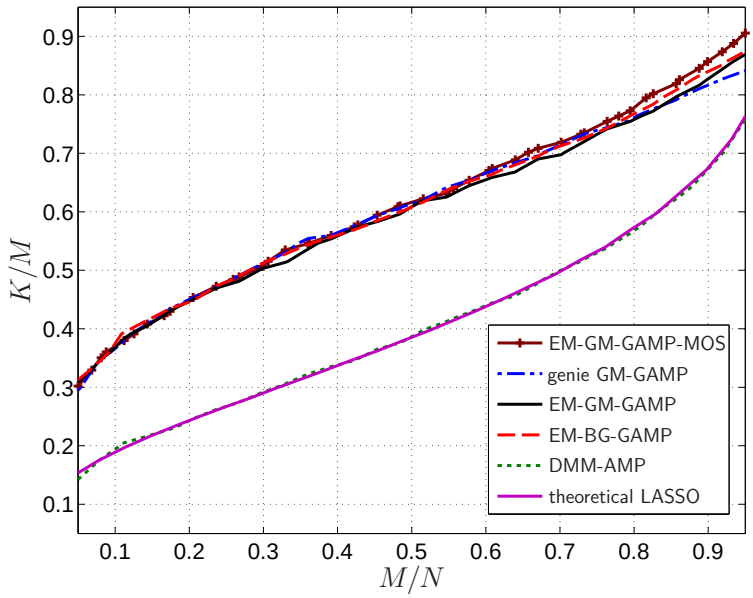


Figure 4.3: Empirical PTCs and LASSO theoretical PTC for noiseless recovery of Bernoulli-Gaussian signals.

Figs. 4.3–4.5 also display the theoretical LASSO PTC (4.36). The signals were generated as Bernoulli-Gaussian (BG) in Fig. 4.3 (using mean $\theta = 0$ and variance $\phi = 1$ for the Gaussian component), as Bernoulli in Fig. 4.4 (i.e., all non-zero coefficients set equal to 1), and as Bernoulli-Rademacher (BR) in Fig. 4.5.

For all three signal types, Figs. 4.3–4.5 show that the empirical PTC of EM-GM-GAMP significantly improves on the empirical PTC of DMM-AMP as well as the theoretical PTC of LASSO. (The latter two are known to converge in the large system limit [16].) For BG signals, Fig. 4.3 shows that EM-GM-GAMP-MOS, EM-GM-GAMP, and EM-BG-GAMP all yield PTCs that are nearly identical to that of genie-GM-GAMP, suggesting that our EM-learning procedures are working well. For Bernoulli signals, Fig. 4.4 shows EM-GM-GAMP-MOS performing very close to genie-GM-GAMP, and both EM-GM-GAMP and EM-BG-GAMP performing slightly

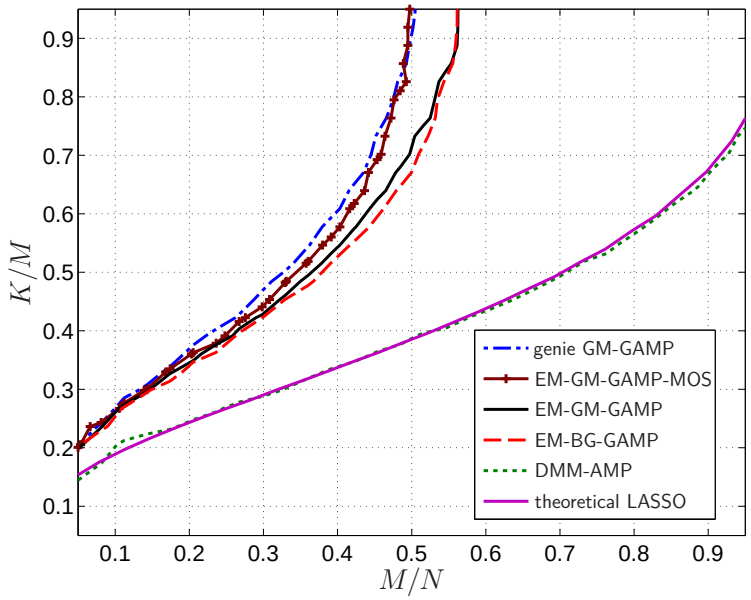


Figure 4.4: Empirical PTCs and LASSO theoretical PTC for noiseless recovery of Bernoulli signals.

worse but far better than DMM-AMP. Finally, for BR signals, Fig. 4.5 shows EM-GM-GAMP performing significantly better than EM-BG-GAMP, since the former is able to accurately model the BR distribution (with $L \geq 2$ mixture components) whereas the latter (with a single mixture component) is not, and on par with genie-GM-GAMP, whereas EM-GM-GAMP-MOS performs noticeably better than genie-GM-GAMP. The latter is due to EM-GM-GAMP-MOS doing per-realization parameter tuning, while genie-GM-GAMP employs the best set of *fixed* parameters over all realizations.

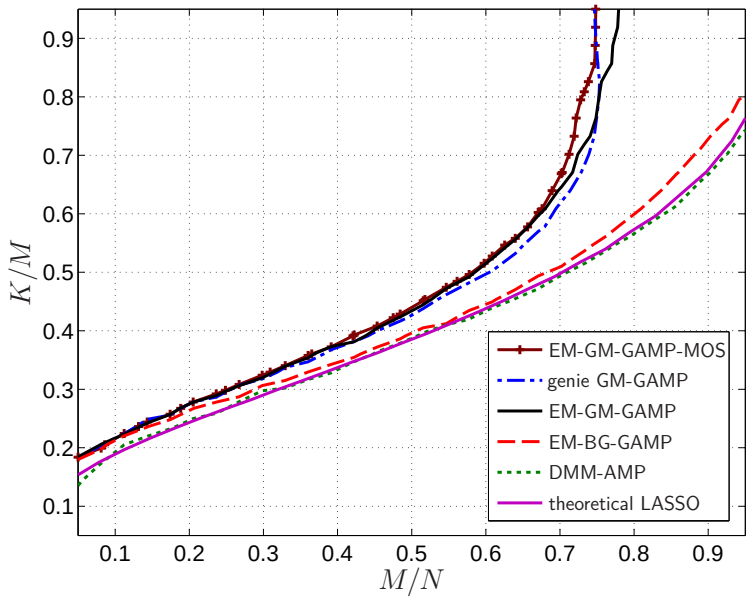


Figure 4.5: Empirical PTCs and LASSO theoretical PTC for noiseless recovery of Bernoulli-Rademacher signals.

To better understand the performance of EM-GM-GAMP when $\frac{M}{N} \ll 1$, we fixed $N = 8192$ and constructed a 12×9 grid of (M, K) values spaced uniformly in the log domain. At each grid point, we generated $R = 100$ independent realizations of a K-sparse BG signal and an i.i.d $\mathcal{N}(0, M^{-1})$ matrix $\boldsymbol{A}$. We then recovered $\boldsymbol{x}$ from the noiseless measurements using EM-GM-GAMP-MOS, EM-GM-GAMP, EM-BG-GAMP, genie-GM-GAMP, and the Lasso-solver[17] FISTA[18] [64]. Figure 4.6 shows that the PTCs of EM-GM-GAMP-MOS and EM-GM-GAMP are nearly identical, slightly better than those of EM-BG-GAMP and genie-GM-GAMP (especially at very small M), and much better than FISTA's.

[17]For this experiment, we also tried DMM-AMP but found that it had convergence problems, and we tried SPGL1 but found performance degradations at small M.

[18]For FISTA, we used the regularization parameter $\lambda_{\mathsf{FISTA}} = 10^{-5}$, which is consistent with the values used for the noiseless experiments in [64].

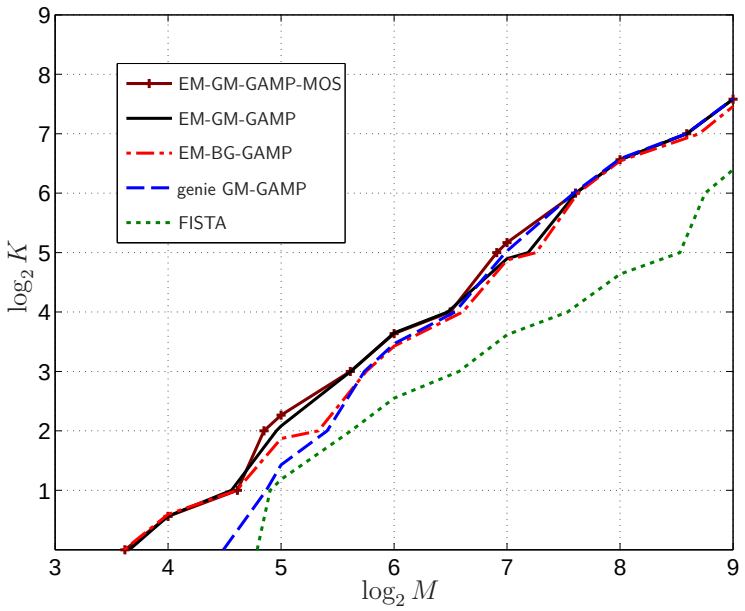


Figure 4.6: Empirical PTCs for noiseless recovery of Bernoulli-Gaussian signals of length $N = 8192$ when $M \ll N$.

Next, we studied the effect of the measurement matrix construction on the performance of EM-GM-GAMP in "sparse" mode with fixed $L = 3$. For this, we plotted EM-GM-GAMP empirical PTCs for noiseless recovery of a length-$N = 1000$ BG signal under several types of measurement matrix $\boldsymbol{A}$: i.i.d $\mathcal{N}(0,1)$, i.i.d Uniform $[-\frac{1}{2}, \frac{1}{2}]$, i.i.d centered Cauchy with scale 1, i.i.d Bernoulli[19] (i.e., $a_{mn} \in \{0,1\}$) with $\lambda_A \triangleq \Pr\{a_{mn} \neq 0\} = 0.15$, i.i.d zero-mean BR (i.e., $a_{mn} \in \{0,1,-1\}$) with $\lambda_A \in \{0.05, 0.15, 1\}$, and randomly row-sampled Discrete Cosine Transform (DCT). Figure 4.7 shows that the EM-GM-GAMP PTC with i.i.d $\mathcal{N}(0,1)$ matrices also holds with the other i.i.d zero-mean sub-Gaussian examples (i.e., Uniform and BR with $\lambda_A = 1$). This is not surprising given that AMP itself has rigorous guarantees for i.i.d zero-mean sub-Gaussian matrices [11]. Figure 4.7 shows that the i.i.d-$\mathcal{N}$ PTC is also

[19]For the Bernoulli and BR matrices, we ensured that no two columns of a given realization $\boldsymbol{A}$ were identical.

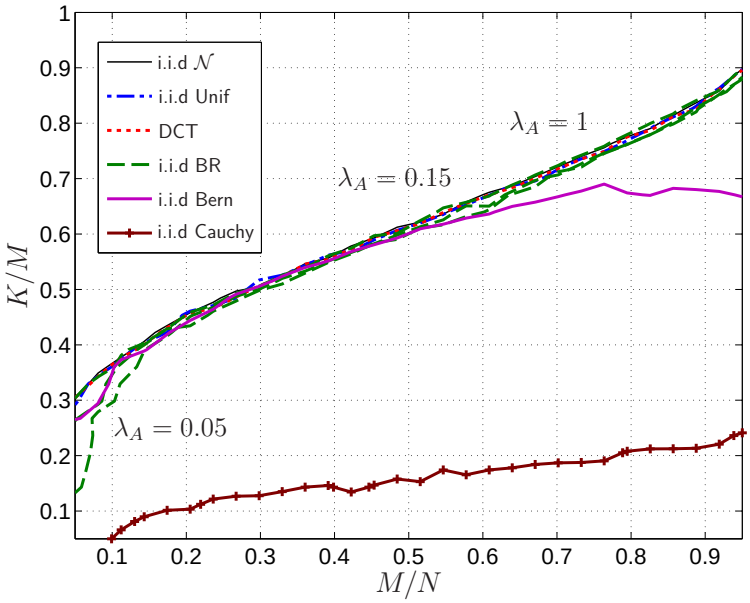


Figure 4.7: Empirical PTCs for EM-GM-GAMP noiseless recovery of Bernoulli-Gaussian signals under various $\boldsymbol{A}$: i.i.d $\mathcal{N}(0,1)$, i.i.d Uniform $[-\frac{1}{2}, \frac{1}{2}]$, i.i.d Bernoulli with $\lambda_A \triangleq \Pr\{a_{mn} \neq 0\} = 0.15$, i.i.d zero-mean Bernoulli-Rademacher with $\lambda_A \in \{0.05, 0.15, 1\}$, i.i.d Cauchy, and randomly row-sampled DCT.

preserved with randomly row-sampled DCT matrices, which is not surprising given AMP's excellent empirical performance with many types of deterministic $\boldsymbol{A}$ [65] even in the absence of theoretical guarantees. Figure 4.7 shows, however, that EM-GM-GAMP's PTC can degrade with non-zero-mean i.i.d matrices (as in the Bernoulli example) or with super-Gaussian i.i.d matrices (as in the BR example with sparsity rate $\lambda_A = 0.05$ and the Cauchy example). Surprisingly, the i.i.d-$\mathcal{N}$ PTC is preserved by i.i.d-BR matrices with sparsity rate $\lambda_A = 0.15$, even though $\lambda_A > \frac{1}{3}$ is required for a BR matrix to be sub-Gaussian [66].

4.4.2 Noisy Sparse Signal Recovery

Figures 4.8–4.10 show NMSE for noisy recovery of BG, Bernoulli, and BR signals, respectively. To construct these plots, we fixed $N = 1000$, $K = 100$, SNR $= 25$ dB, and varied M. Each data point represents NMSE averaged over $R = 500$ realizations, where in each realization we drew an $\boldsymbol{A}$ with i.i.d $\mathcal{N}(0, M^{-1})$ elements, an AWGN noise vector, and a random signal vector. For comparison, we show the performance of the proposed EM-GM-GAMP (in "sparse" mode) for both MOS and $L = 3$ versions, EM-BG-GAMP, genie-tuned[20] Orthogonal Matching Pursuit (OMP) [67], genie-tuned[20] Subspace Pursuit (SP) [68], Bayesian Compressive Sensing (BCS) [57], Sparse Bayesian Learning [56] (via the more robust T-MSBL [69]), de-biased genie-tuned[21] LASSO (via SPGL1 [70]), and Smoothed-ℓ_0 (SL0) [71]. All algorithms were run under the suggested defaults, with `noise=small` in T-MSBL.

[20] We ran both the SP and OMP algorithms (using the publicly available implementation from http://sparselab.stanford.edu/OptimalTuning/code.htm) under 10 different sparsity assumptions, spaced uniformly from 1 to $2K$, and reported the lowest NMSE among the results.

[21] We ran SPGL1 in 'BPDN' mode: $\min_{\hat{\boldsymbol{x}}} \|\boldsymbol{x}\|_1$ s.t. $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2 \leq \sigma$, for hypothesized tolerances $\sigma^2 \in \{0.1, 0.2, \ldots, 1.5\} \times M\psi$, and reported the lowest NMSE among the results.

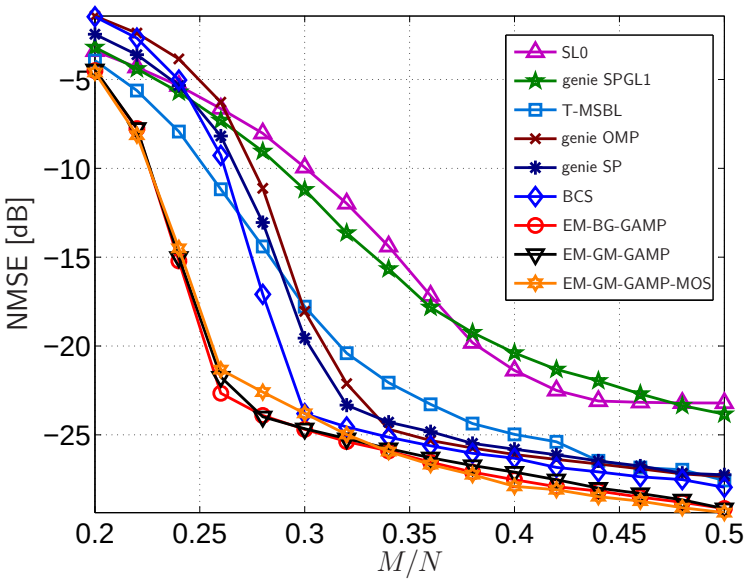


Figure 4.8: NMSE versus undersampling ratio M/N for noisy recovery of Bernoulli-Gaussian signals.

For BG signals, Fig. 4.8 shows that EM-GM-GAMP-MOS, EM-GM-GAMP, and EM-BG-GAMP together exhibit the best performance among the tested algorithms, reducing the M/N breakpoint (i.e., the location of the knee in the NMSE curve, which represents a sort of phase transition) from 0.3 down to 0.26, but also improving NMSE by ≈ 1 dB relative to the next best algorithm, which was BCS. Relative to the other EM-GAMP variants, MOS resulted in a slight degradation of performance for $\frac{M}{N}$ between 0.26 and 0.31, but was otherwise identical. For Bernoulli signals, Fig. 4.9 shows much more significant gains for EM-GM-GAMP-MOS, EM-GM-GAMP and EM-BG-GAMP over the other algorithms: the M/N breakpoint was reduced from 0.4 down to 0.32 (and even 0.3 with MOS), and the NMSE was reduced by ≈ 8 dB relative to the next best algorithm, which was T-MSBL in this case. Finally, for BR signals, Fig. 4.10 shows a distinct advantage for EM-GM-GAMP and EM-GM-GAMP-MOS

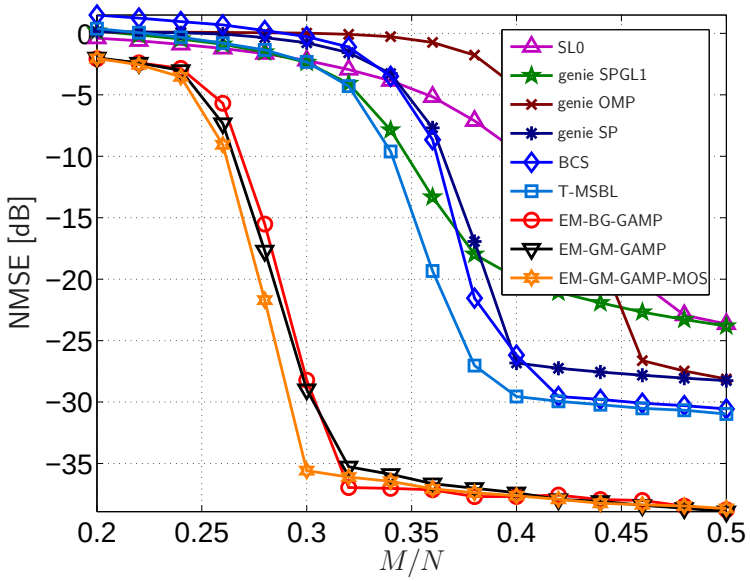


Figure 4.9: NMSE versus undersampling ratio M/N for noisy recovery of Bernoulli signals.

over the other algorithms, including EM-BG-GAMP, due to the formers' ability to accurately model the BR signal prior. In particular, for $M/N \geq 0.36$, EM-GM-GAMP-MOS reduces the NMSE by 10 dB relative to the best of the other algorithms (which was either EM-BG-GAMP or T-MSBL depending on the value of M/N) and reduces the M/N breakpoint from 0.38 down to 0.35.

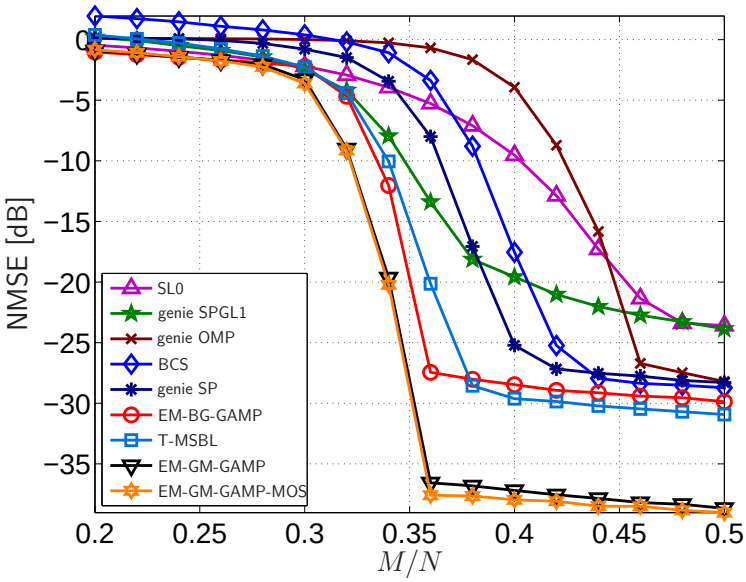


Figure 4.10: NMSE versus undersampling ratio M/N for noisy recovery of Bernoulli-Rademacher signals.

To investigate each algorithm's robustness to AWGN, we plotted the NMSE attained in the recovery of BR signals with $N = 1000$, $M = 500$, and $K = 100$ as a function of SNR in Fig. 4.11, where each point represents an average over $R = 100$ problem realizations, where in each realization we drew an $\boldsymbol{A}$ with i.i.d $\mathcal{N}(0, M^{-1})$ elements, an AWGN noise vector, and a random signal vector. All algorithms were under the same conditions as those reported previously, except that T-MSBL used `noise=small` when SNR $>$ 22dB and `noise=mild` when SNR $\leq$ 22 dB, as recommended in [72]. From Fig. 4.11, we see that the essential behavior observed in the fixed-SNR BR plot Fig. 4.10 holds over a wide range of SNRs. In particular, Fig. 4.11 shows that EM-GM-GAMP and EM-GM-GAMP-MOS yield significantly lower NMSE than all other algorithms over the full SNR range, while EM-BG-GAMP and T-MSBL yield the second lowest NMSE (also matched by BCS for SNRs between 30 and 40

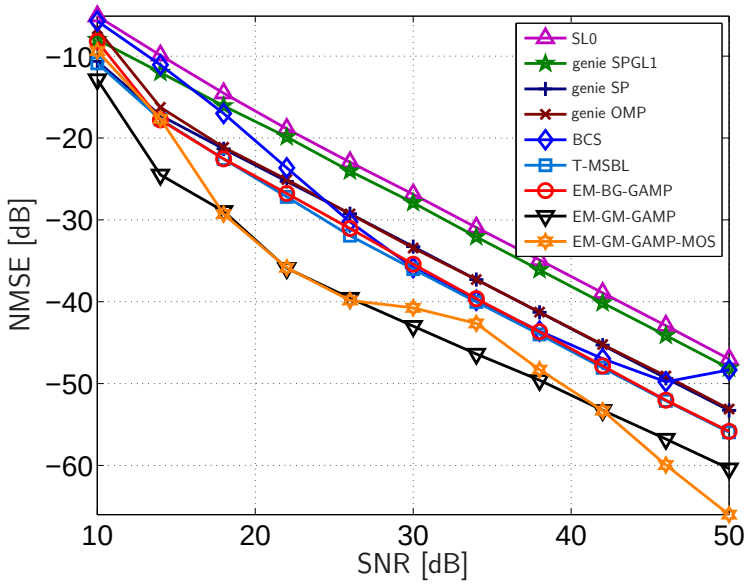


Figure 4.11: NMSE versus SNR for noisy recovery of Bernoulli-Rademacher signals.

dB). Note, however, than T-MSBL must be given some knowledge about the true noise variance in order to perform well [72], unlike the proposed algorithms.

4.4.3 Heavy-Tailed Signal Recovery

In many applications of compressive sensing, the signal to be recovered is not perfectly sparse, but instead contains a few large coefficients and many small ones. While the literature often refers to such signals as "compressible," there are many real-world signals that do not satisfy the technical definition of compressibility (see, e.g., [4]), and so we refer to such signals more generally as "heavy tailed."

To investigate algorithm performance for these signals, we first consider an i.i.d Student's-t signal, with prior pdf

$$p_{\mathsf{x}}(x;q) \triangleq \frac{\Gamma((q+1)/2))}{\sqrt{\pi}\Gamma(q/2)}\left(1+x^2\right)^{-(q+1)/2} \tag{4.48}$$

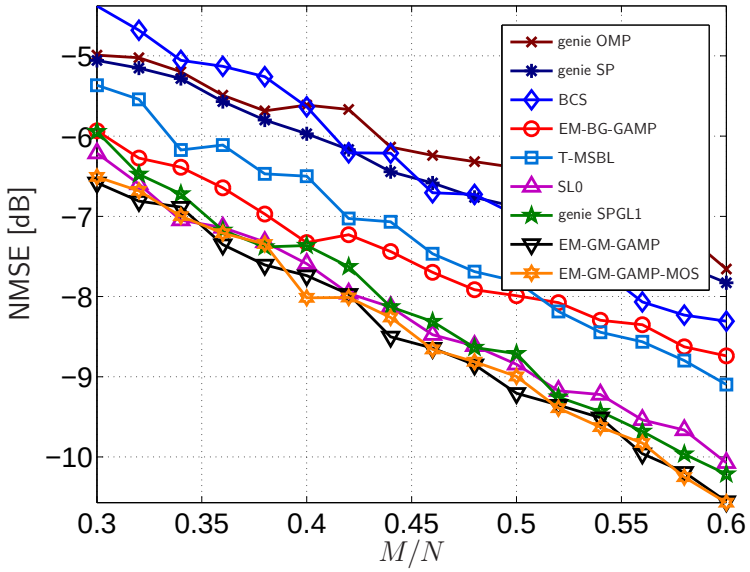


Figure 4.12: NMSE versus undersampling ratio M/N for noisy recovery of Student-t signals with rate parameter 1.67.

under the (non-compressible) rate $q = 1.67$, which has been shown to be an excellent model for wavelet coefficients of natural images [4]. For such signals, Fig. 4.12 plots NMSE versus the number of measurements M for fixed $N = 1000$, SNR $= 25$ dB, and an average of $R = 500$ realizations, where in each realization we drew an $\boldsymbol{A}$ with i.i.d $\mathcal{N}(0, M^{-1})$ elements, an AWGN noise vector, and a random signal vector. Figure 4.12 shows both variants of EM-GM-GAMP (here run in "heavy-tailed" mode) outperforming all other algorithms under test.[22] We have also verified (in experiments not shown here) that "heavy-tailed" EM-GM-GAMP exhibits similarly good performance with other values of the Student's-t rate parameter q, as well as for i.i.d centered Cauchy signals.

[22]In this experiment, we ran both OMP and SP under 10 different sparsity hypotheses, spaced uniformly from 1 to $K_{\text{lasso}} = M\rho_{\text{SE}}(\frac{M}{N})$, and reported the lowest NMSE among the results.

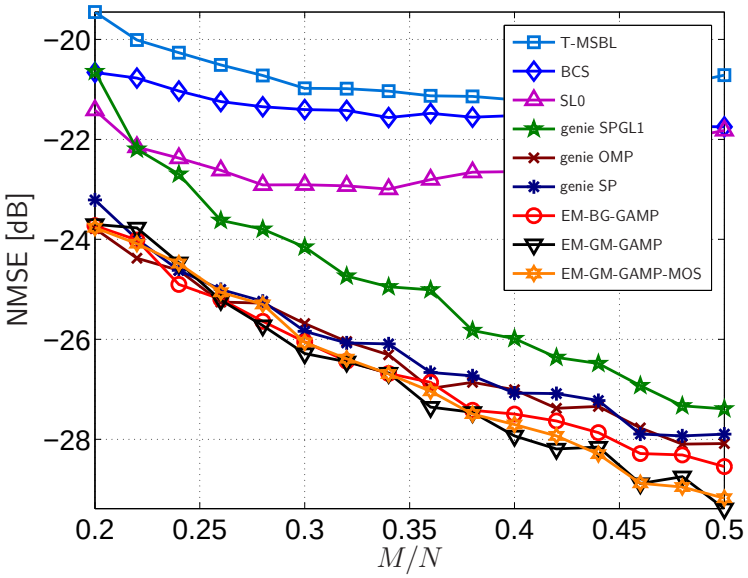


Figure 4.13: NMSE versus undersampling ratio M/N for noisy recovery of log-normal signals with location parameter 0 and scale parameter 1.

To investigate the performance for positive heavy-tailed signals, we conducted a similar experiment using i.i.d log-normal $\boldsymbol{x}$, generated using the distribution

$$p_{\mathsf{x}}(x;\mu,\sigma^2) = \tfrac{1}{x\sqrt{2\pi\sigma^2}} \exp -\tfrac{(\ln x-\mu)}{2\sigma^2} \tag{4.49}$$

with location parameter $\mu = 0$ and scale parameter $\sigma^2 = 1$. Figure 4.13 confirms the excellent performance of EM-GM-GAMP-MOS, EM-GM-GAMP, and EM-BG-GAMP over all tested undersampling ratios M/N. We postulate that, for signals known apriori to be positive, EM-GM-GAMP's performance could be further improved through the use of a prior p_{x} with support restricted to the the positive reals, via a mixture of positively truncated Gaussians.

It may be interesting to notice that, with the perfectly sparse signals examined in Figs. 4.8–4.10, SL0 and SPGL1 performed relatively poorly, the relevance-vector-machine (RVM)-based approaches (i.e., BCS, T-MSBL) performed relatively well, and

the greedy approaches (OMP and SP) performed in-between. With the heavy-tailed signals in Figs. 4.12–4.13, it is more difficult to see a consistent pattern. For example, with the Student's-t signal, the greedy approaches performed the worse, the RVM approaches were in the middle, and SL0 and SPGL1 performed very well. But with the log-normal signal, the situation was very different: the greedy approaches performed very well, SPGL1 performed moderately well, but SL0 and the RVM approaches performed very poorly.

In conclusion, for *all* of the many signal types tested above, the best recovery performance came from EM-GM-GAMP and its MOS variant. We attribute this behavior to EM-GM-GAMP's ability to tune itself to the signal (and in fact the realization) at hand.

4.4.4 Runtime and Complexity Scaling with Problem Size

Next we investigated how complexity scales with signal length N by evaluating the runtime of each algorithm on a typical personal computer. For this, we fixed $K/N = 0.1$, $M/N = 0.5$, $\mathsf{SNR} = 25$ dB and varied the signal length N. Figure 4.14 shows the runtimes for noisy recovery of a Bernoulli-Rademacher signal, while Fig. 4.15 shows the corresponding NMSEs. In these plots, each datapoint represents an average over $R = 50$ realizations. The algorithms that we tested are the same ones that we described earlier. However, to fairly evaluate runtime, we configured some a bit differently than before. In particular, for genie-tuned SPGL1, in order to yield a better runtime-vs-NMSE tradeoff, we reduced the tolerance grid (recall footnote 21) to $\sigma^2 \in \{0.6, 0.8, \ldots, 1.4\} \times M\psi$ and turned off debiasing. For OMP and SP, we used the fixed support size $K_{\mathsf{lasso}} = M\rho_{\mathsf{SE}}(\frac{M}{N})$ rather than searching for the size that

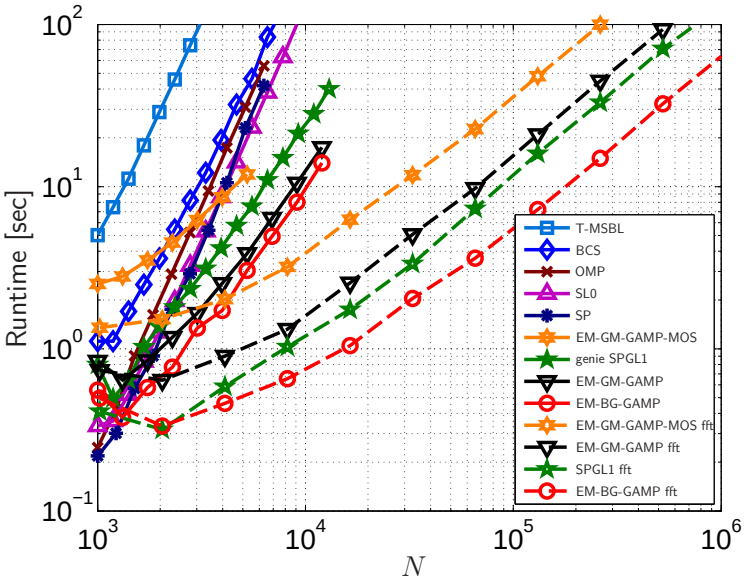


Figure 4.14: Runtime versus signal length N for noisy recovery of Bernoulli-Rademacher signals.

minimizes NMSE over a grid of 10 hypotheses, as before. Otherwise, all algorithms were run under the suggested defaults, with T-MSBL run under `noise=small` and EM-GM-GAMP run in "sparse" mode.

The complexities of the proposed EM-GM-GAMP methods are dominated by one matrix multiplication by $\boldsymbol{A}$ and $\boldsymbol{A}^{\mathsf{T}}$ per iteration. Thus, when these matrix multiplications are explicitly implemented and $\boldsymbol{A}$ is dense, the total complexity of EM-GM-GAMP should scale as $\mathcal{O}(MN)$. This scaling is indeed visible in the runtime curves of Fig. 4.14. There, $\mathcal{O}(MN)$ becomes $\mathcal{O}(N^2)$ since the ratio M/N was fixed, and the horizontal axis plots N on a logarithmic scale, so that this complexity scaling manifests, at sufficiently large values of N, as a line with slope 2. Figure 4.14 confirms that genie-tuned SPGL1 also has the same complexity scaling, albeit with longer overall runtimes. Meanwhile, Fig. 4.14 shows T-MSBL, BCS, SL0, OMP, and

SP exhibiting a complexity scaling of $\mathcal{O}(N^3)$ (under fixed K/N and M/N), which results in orders-of-magnitude larger runtimes for long signals (e.g., $N \geq 10^4$). With short signals (e.g., $N < 1300$), though, OMP, SP, SL0, and SPGL1 are faster than EM-GM-GAMP. Finally, Fig. 4.15 verifies that, for most of the algorithms, the NMSEs are relatively insensitive to signal length N when the undersampling ratio M/N and sparsity ratio K/M are both fixed, although the performance of EM-GM-GAMP improves with N (which is not surprising in light of AMP's large-system-limit optimality properties [30]) and the performance of BCS degrades with N.

Both the proposed EM-GM-GAMP methods and SPGL1 can exploit the case where multiplication by $\boldsymbol{A}$ and $\boldsymbol{A}^{\mathsf{T}}$ is implemented using a fast algorithm like the fast Fourier transform (FFT)[23], which reduces the complexity to $\mathcal{O}(N \log N)$, and avoids the need to store $\boldsymbol{A}$ in memory—a potentially serious problem when MN is large. The dashed lines in Figs. 4.14–4.15 (labeled "fft") show the average runtime and NMSE of the proposed algorithms and SPGL1 in case that $\boldsymbol{A}$ was a randomly row-sampled FFT. As expected, the runtimes are dramatically reduced. While EM-BG-GAMP retains its place as the fastest algorithm, SPGL1 now runs 1.5× faster than EM-GM-GAMP (at the cost of 14 dB higher NMSE). The MOS version of EM-GM-GAMP yields slightly better NMSE, but takes ≈ 2.5 times as long to run as the fixed-L version.

[23]For our FFT-based experiments, we used the complex-valued versions of EM-BG-GAMP, EM-GM-GAMP, and SPGL1.

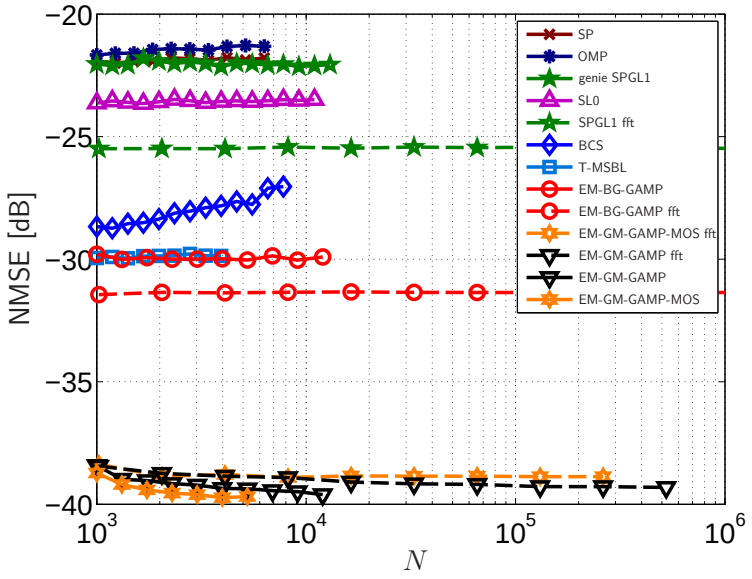


Figure 4.15: NMSE versus signal length N for noisy recovery of Bernoulli-Rademacher signals.

4.4.5 Example: Compressive Recovery of Audio

As a practical example, we experimented with the recovery of an audio signal from compressed measurements. The full length-81920 audio signal was first partitioned into T blocks $\{\boldsymbol{u}_t\}_{t=1}^T$ of length N. Noiseless compressed measurements $\boldsymbol{y}_t = \boldsymbol{\Phi}\boldsymbol{u}_t \in \mathbb{R}^M$ were then collected using $M = N/2$ samples per block. Rather than reconstructing $\boldsymbol{u}_t$ directly from $\boldsymbol{y}_t$, we first reconstructed[24] the transform coefficients $\boldsymbol{x}_t = \boldsymbol{\Psi}^\mathsf{T}\boldsymbol{u}_t$, using the (orthogonal) discrete cosine transform (DCT) $\boldsymbol{\Psi} \in \mathbb{R}^{N\times N}$, and later reconstructed $\boldsymbol{u}_t$ via $\boldsymbol{u}_t = \boldsymbol{\Psi}\boldsymbol{x}_t$. Our effective sparse-signal model can thus be written as $\boldsymbol{y}_t = \boldsymbol{A}\boldsymbol{x}_t$ with $\boldsymbol{A} = \boldsymbol{\Phi}\boldsymbol{\Psi}$. We experimented with two types of measurement matrix $\boldsymbol{\Phi}$: i.i.d zero-mean Gaussian and random selection (i.e., containing rows

[24] Although one could exploit additional structure among the multiple-timestep coefficients $\{\boldsymbol{x}_t\}_{t=1}^T$ for improved recovery (e.g., sparsity clustering in the time and/or frequency dimensions, as well as amplitude correlation in those dimensions) as demonstrated in [73], such techniques are outside the scope of this chapter.

of the identity matrix selected uniformly at random), noting that the latter allows a fast implementation of $\boldsymbol{A}$ and $\boldsymbol{A}^{\mathsf{T}}$. Table 4.2 shows the resulting time-averaged NMSE, i.e., $\mathsf{TNMSE} \triangleq \frac{1}{T}\sum_{t=1}^{T} ||\boldsymbol{u}_t - \hat{\boldsymbol{u}}_t||^2/||\boldsymbol{u}_t||^2$, and total runtime achieved by the previously described algorithms at block lengths $N = 1024, 2048, 4096, 8192$, which correspond to $T = 80, 40, 20, 10$ blocks, respectively. The numbers reported in the table represent an average over 50 realizations of $\boldsymbol{\Phi}$. For these experiments, we configured the algorithms as described in Chapter 4.4.3 for the heavy-tailed experiment except that, for genie-SPGL1, rather than using $\psi = 0$, we used $\psi = 10^{-6}$ for the tolerance grid (recall footnote 21) because we found that this value minimized TNMSE and, for T-MSBL, we used the setting `prune_gamma` $= 10^{-12}$ as recommended in a personal correspondence with the author. For certain combinations of algorithm and blocklength, excessive runtimes prevented us from carrying out the experiment, and thus no result appears in the table.

Table 4.2 shows that, for this audio experiment, the EM-GM-GAMP methods and SL0 performed best in terms of TNMSE. As in the synthetic examples presented earlier, we attribute EM-GM-GAMP's excellent TNMSE to its ability to tune itself to whatever signal is at hand. As for SL0's excellent TNMSE, we reason that it had the good fortune of being particularly well-tuned to this audio signal, given that it performed relatively poorly with the signal types used for Figs. 4.8–4.11 and Fig. 4.13. From the runtimes reported in Table 4.2, we see that, with i.i.d Gaussian $\boldsymbol{\Phi}$ and the shortest block length ($N = 1024$), genie-OMP is by far the fastest, whereas the EM-GM-GAMP methods are the slowest. But, as the block length grows, the EM-GM-GAMP methods achieve better and better runtimes as a consequence of their

excellent complexity scaling, and eventually EM-BG-GAMP and fixed-L EM-GM-GAMP become the two fastest algorithms under test (as shown with i.i.d Gaussian $\mathbf{\Phi}$ at $N = 8192$). For this audio example, the large-block regime may be the more important, because that is where all algorithms give their smallest TNMSE. Next, looking at the runtimes under random-selection $\mathbf{\Phi}$, we see dramatic speed improvements for the EM-GM-GAMP methods and SPGL1, which were all able to leverage Matlab's fast DCT. In fact, the total runtimes of these four algorithms *decrease* as N is increased from 1024 to 8192. We conclude by noting that EM-BG-GAMP (at $N = 8192$ with random selection $\mathbf{\Phi}$) achieves the fastest runtime in the entire table while yielding a TNMSE that is within 1.3 dB of the best value in the entire table. Meanwhile, fixed-L EM-GM-GAMP (at $N = 8192$ with random selection $\mathbf{\Phi}$) gives TNMSE only 0.3 dB away from the best in the entire table with a runtime of only about twice the best in the entire table. Finally, the best TNMSEs in the entire table are achieved by EM-GM-GAMP-MOS (at $N = 8192$), which takes ≈ 2.5 times as long to run as its fixed-L counterpart.

		$N = 1024$		$N = 2048$		$N = 4096$		$N = 8192$	
		TNMSE	time	TNMSE	time	TNMSE	time	TNMSE	time
i.i.d Gaussian $\mathbf{\Phi}$	EM-GM-GAMP-MOS	**-17.3**	468.9	**-18.3**	487.2	**-21.0**	967.9	**-21.8**	2543
	EM-GM-GAMP	-16.9	159.2	-18.0	213.2	-20.7	434.0	-21.4	1129
	EM-BG-GAMP	-15.9	115.2	-17.0	174.1	-19.4	430.2	-20.0	**1116**
	SL0	-16.8	41.6	-17.9	128.5	-20.6	629.0	-21.3	2739
	genie SPGL1	-14.3	90.9	-16.2	200.6	-18.6	514.3	-19.5	1568
	BCS	-15.0	67.5	-15.8	149.1	-18.4	**428.0**	-18.8	2295
	T-MSBL	-16.3	1.2e4	–	–	–	–	–	–
	genie OMP	-13.9	**20.1**	-14.9	**109.9**	-17.6	527.0	–	–
	genie SP	-14.5	87.7	-15.5	305.9	-18.0	1331	–	–
random selection $\mathbf{\Phi}$	EM-GM-GAMP-MOS	-16.6	233.0	-17.5	136.1	**-20.5**	109.6	**-21.6**	93.9
	EM-GM-GAMP	**-16.7**	56.1	**-17.7**	43.7	**-20.5**	38.0	-21.5	37.8
	EM-BG-GAMP	-16.2	29.6	-17.2	**22.3**	-19.7	**19.4**	-20.5	**18.0**
	SL0	**-16.7**	35.7	-17.6	119.5	-20.4	597.8	-21.2	2739
	genie SPGL1	-14.0	34.4	-15.9	24.5	-18.4	21.7	-19.7	19.6
	BCS	-15.5	60.5	-16.1	126.2	-19.4	373.8	-20.2	2295
	T-MSBL	-15.5	1.2e4	–	–	–	–	–	–
	genie OMP	-15.1	**20.1**	-15.7	106.8	-18.9	506.0	–	–
	genie SP	-15.2	104.5	-16.1	395.3	-18.7	1808	–	–

Table 4.2: Average TNMSE (in dB) and total runtime (in seconds) for compressive audio recovery.

4.5 Conclusions

Those interested in practical compressive sensing face the daunting task of choosing among literally hundreds of signal reconstruction algorithms (see, e.g., [74]). In testing these algorithms, they are likely to find that some work very well with particular signal classes, but not with others. They are also likely to get frustrated by those algorithms that require the tuning of many parameters. Finally, they are likely to find that some of the algorithms that are commonly regarded as "very fast" are actually very slow in high-dimensional problems. Meanwhile, those familiar with the theory of compressive sensing know that the workhorse LASSO is nearly minimax optimal,

and that its phase transition curve is robust to the nonzero-coefficient distribution of sparse signals. However, they also know that, for most signal classes, there is a large gap between the MSE performance of LASSO and that of the MMSE estimator derived under full knowledge of the signal and noise statistics [17]. Thus, they may wonder whether there is a way to close this gap by designing a signal reconstruction algorithm that *both learns and exploits* the signal and noise statistics.

With these considerations in mind, we proposed an empirical Bayesian approach to compressive signal recovery that merges two powerful inference frameworks: expectation maximization (EM) and approximate message passing (AMP). We then demonstrated—through a detailed numerical study—that our approach, when used with a flexible Gaussian-mixture signal prior, achieves a state-of-the-art combination of reconstruction error and runtime on a very wide range of signal and matrix types in the high-dimensional regime.

Chapter 5: Enforcing Non-negativity and Linear Equality Constraints

5.1 Introduction

We consider the recovery of an (approximately) sparse signal $\boldsymbol{x} \in \mathbb{R}^N$ from the noisy linear measurements

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{w} \in \mathbb{R}^M, \tag{5.1}$$

where $\boldsymbol{A}$ is a known sensing matrix, $\boldsymbol{w}$ is noise, and M may be $\ll N$.[25] In this chapter, we focus on non-negative (NN) signals (i.e., $x_n \geq 0 \ \forall n$) that obey known linear equality constraints $\boldsymbol{B}\boldsymbol{x} = \boldsymbol{c} \in \mathbb{R}^P$. A notable example is *simplex*-constrained signals, i.e., $\boldsymbol{x} \in \Delta_+^N \triangleq \{\boldsymbol{x} \in \mathbb{R}^N : x_n \geq 0 \ \forall n, \mathbf{1}^\mathsf{T}\boldsymbol{x} = 1\}$, occurring in hyperspectral image unmixing [18], portfolio optimization [75, 76], density estimation [77, 78], and other applications. We also consider the recovery of NN sparse signals without the linear constraint $\boldsymbol{B}\boldsymbol{x} = \boldsymbol{c}$ [79–81], which arises in imaging applications [82] and elsewhere [83].

As previously discussed in Chapter 1.1.3, one approach to recovering linearly constrained NN sparse $\boldsymbol{x}$ is to solve the ℓ_1-penalized constrained NN least-squares (LS) problem (5.2) (see, e.g., [76]) for some $\lambda \geq 0$:

$$\hat{\boldsymbol{x}}_{\text{c-lasso}} = \underset{\boldsymbol{x} \geq 0}{\arg\min} \tfrac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_1 \ \text{ s.t. } \ \boldsymbol{B}\boldsymbol{x} = \boldsymbol{c}. \tag{5.2}$$

[25] This chapter is excerpted from our work published in [35].

Although this problem is convex [84], finding a solution can be computationally challenging in the high-dimensional regime. Also, while a larger λ is known to promote more sparsity in $\hat{\boldsymbol{x}}$, determining the best choice of λ can be difficult in practice. For example, methods based on cross-validation, the L-curve, or Stein's unbiased risk estimator can be used (see [85] for discussions of all three), but they require much more computation than solving (5.2) for a fixed λ. For this reason, (5.2) is often considered under the special case $\lambda = 0$ [86], where it reduces to linearly constrained NN-LS.

For the recovery of K-sparse simplex-constrained signals, a special case of the general problem under consideration, the Greedy Selector and Simplex Projector (GSSP) was proposed in [78]. GSSP, an instance of projected gradient descent, iterates

$$\hat{\boldsymbol{x}}^{i+1} = \mathcal{P}_K\Big(\hat{\boldsymbol{x}}^i - \mathsf{step}^i \nabla_{\boldsymbol{x}} \|\boldsymbol{y} - \boldsymbol{A}\hat{\boldsymbol{x}}^{\boldsymbol{i}}\|_2^2\Big), \tag{5.3}$$

where $\mathcal{P}_K(\cdot)$ is the Euclidean projection onto the K-sparse simplex, $\hat{\boldsymbol{x}}^i$ is the iteration-i estimate, step^i is the iteration-i step size, and $\nabla_{\boldsymbol{x}}$ is the gradient w.r.t $\boldsymbol{x}$. For algorithms of this sort, rigorous approximation guarantees can be derived when $\boldsymbol{A}$ obeys the restricted isometry property [87]. Determining the best choice of K can, however, be difficult in practice.

In this chapter, we propose two methods for recovering a linearly constrained NN sparse vector $\boldsymbol{x}$ from noisy linear observations $\boldsymbol{y}$ of the form (5.1), both of which are based on the Generalized Approximate Message Passing (GAMP) algorithm [28], an approximation of loopy belief propagation that has close connections to primal-dual optimization algorithms [38, 39]. When run in "max-sum" mode, GAMP can be used to solve optimization problems of the form $\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x}} \sum_{m=1}^{M} h_m([\boldsymbol{A}\boldsymbol{x}]_m) +$

$\sum_{n=1}^{N} g_n(x_n)$, where $\hat{\boldsymbol{x}}$ can be interpreted as the *maximum a posteriori* (MAP) estimate[26] of $\boldsymbol{x}$ under the assumed signal prior (5.4) and likelihood (5.5):

$$p(\boldsymbol{x}) \propto \prod_{n=1}^{N} \exp(-g_n(x_n)) \tag{5.4}$$

$$p(\boldsymbol{y}|\boldsymbol{A}\boldsymbol{x}) \propto \prod_{m=1}^{M} \exp(-h_m([\boldsymbol{A}\boldsymbol{x}]_m)). \tag{5.5}$$

When run in "sum-product" mode, GAMP returns an approximation of the minimum mean-squared error (MMSE) estimate of $\boldsymbol{x}$ under the same assumptions. In either case, the linear equality constraints $\boldsymbol{B}\boldsymbol{x} = \boldsymbol{c}$ can be enforced through the use of noiseless pseudo-measurements, as described in the sequel.

The first of our proposed approaches solves (5.2) using max-sum GAMP while tuning λ using a novel expectation-maximization (EM) [31] procedure. We henceforth refer to this approach as EM-NNL-GAMP, where NNL is short for "non-negative LASSO.[27]" We demonstrate, via extensive numerical experiments, that 1) the *runtime* of our approach is much faster than the state-of-the-art TFOCS solver [88] for a fixed λ, and that 2) the MSE *performance* of our λ-tuning procedure is on par with TFOCS under *oracle* tuning. We also consider the special case of $\lambda = 0$, yielding "non-negative least squares GAMP" (NNLS-GAMP), whose performance and runtime compare favorably to Matlab's `lsqlin` routine. In addition, we consider a variation on (5.2) that replaces the quadratic loss $\frac{1}{2}\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|_2^2$ with the absolute loss $\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|_1$ for improved robustness to outliers in $\boldsymbol{w}$ [89], and demonstrate the potential advantages of this technique on a practical dataset.

[26] When the optimization problem is convex, max-sum GAMP yields exact MAP estimates for arbitrary $\boldsymbol{A}$ if it converges.

[27] In the absence of the constraint $\boldsymbol{B}\boldsymbol{x}=\boldsymbol{c}$, the optimization problem (5.2) can be recognized as a non-negatively constrained version of the LASSO [13] (also known as basis-pursuit denoising [14]). Similarly, in the special case of $\lambda=0$, (5.2) reduces to non-negative LS [86].

The second of our proposed approaches aims to solve not an optimization problem like (5.2) but rather an inference problem: compute the *MMSE estimate* of a linearly constrained NN sparse vector $\boldsymbol{x}$ from noisy linear observations $\boldsymbol{y}$. This is in general a daunting task, since computing the true MMSE estimate requires i) knowing both the true signal prior $p(\boldsymbol{x})$ and likelihood $p(\boldsymbol{y}|\boldsymbol{A}\boldsymbol{x})$, which are rarely available in practice, and ii) performing optimal inference w.r.t that prior and likelihood, which is rarely possible in practice for computational reasons.

However, when the coefficients in $\boldsymbol{x}$ are i.i.d and the observation matrix $\boldsymbol{A}$ in (5.1) is sufficiently large and random, recent work [34] has demonstrated that near-MMSE estimation is indeed possible via the following methodology: place an i.i.d Gaussian-mixture (GM) model with parameters $\boldsymbol{q}$ on the coefficients $\{x_n\}$, run sum-product GAMP based on that model, and tune the model parameters $\boldsymbol{q}$ using an appropriately designed EM algorithm. For such $\boldsymbol{A}$, the asymptotic optimality of GAMP as an MMSE-inference engine was established in [28, 40], and the ability of EM-GAMP to achieve consistent estimates of $\boldsymbol{q}$ was established in [58].

In this work, we show that the EM-GM-GAMP approach from [34] can be extended to *linearly constrained non-negative* signal models through the use of a non-negative Gaussian-mixture (NNGM) model and noiseless pseudo-measurements, and we detail the derivation and implementation of the resulting algorithm. Moreover, we demonstrate, via extensive numerical experiments, that EM-NNGM-GAMP's reconstruction MSE is state-of-the-art and that its runtime compares favorably to existing methods.

Both of our proposed approaches can be classified as "empirical-Bayes" [3] in the sense that they combine Bayesian and frequentist approaches: GAMP performs (MAP

or MMSE) Bayesian inference with respect to a given prior, where the parameters of the prior are treated as deterministic and learned using the EM algorithm, a maximum-likelihood (ML) approach.

5.2 Observation Models

To enforce the linear equality constraint $\boldsymbol{B}\boldsymbol{x} = \boldsymbol{c} \in \mathbb{R}^P$ using GAMP, we extend the observation model (5.1) to

$$\underbrace{\begin{bmatrix}\boldsymbol{y}\\ \boldsymbol{c}\end{bmatrix}}_{\triangleq\, \bar{\boldsymbol{y}}} = \underbrace{\begin{bmatrix}\boldsymbol{A}\\ \boldsymbol{B}\end{bmatrix}}_{\triangleq\, \bar{\boldsymbol{A}}} \boldsymbol{x} + \begin{bmatrix}\boldsymbol{w}\\ \boldsymbol{0}\end{bmatrix} \tag{5.6}$$

and exploit the fact that GAMP supports a likelihood function that varies with the measurement index m. Defining $\bar{\boldsymbol{z}} \triangleq \bar{\boldsymbol{A}}\boldsymbol{x}$, the likelihood associated with the augmented model (5.6) can be written as

$$p_{\overline{\mathsf{y}}_m|\overline{\mathsf{z}}_m}(\overline{y}_m|\overline{z}_m) = \begin{cases} p_{\mathsf{y}|\mathsf{z}}(\overline{y}_m|\overline{z}_m) & m = 1,\dots,M \\ \delta(\overline{y}_m - \overline{z}_m) & m = M+1,\dots,M+P, \end{cases} \tag{5.7}$$

where $p_{\mathsf{y}|\mathsf{z}}$ corresponds to the first M measurements, i.e., (5.1).

Note that, for either max-sum or sum-product GAMP, the quantities in (R3)-(R6) of Table 2.1 then become

$$\hat{z}_m(t) = c_{m-M} \qquad m = M+1,\dots,M+P \tag{5.8}$$

$$\mu^z_m(t) = 0 \qquad m = M+1,\dots,M+P, \tag{5.9}$$

where c_{m-M} are elements of $\boldsymbol{c}$.

5.2.1 Additive white Gaussian noise

When the noise $\boldsymbol{w}$ is modeled as additive white Gaussian noise (AWGN) with variance ψ, the likelihood $p_{y|z}$ in (5.7) takes the form

$$p_{y|z}(y|z) = \mathcal{N}(y; z, \psi). \tag{5.10}$$

In this case, for either max-sum or sum-product GAMP, the quantities in (R3)-(R6) of Table 2.1 become [28] (omitting the t index for brevity)

$$\hat{z}_m = \hat{p}_m + \tfrac{\mu^p_m}{\mu^p_m+\psi}(y_m - \hat{p}_m) \qquad m = 1, \dots, M \tag{5.11}$$

$$\mu^z_m = \frac{\mu^p_m \psi}{\mu^p_m + \psi} \qquad m = 1, \dots, M. \tag{5.12}$$

5.2.2 Additive white Laplacian noise

The additive white Laplacian noise (AWLN) observation model is an alternative to the AWGN model that is more robust to outliers [89]. Here, the noise $\boldsymbol{w}$ is modeled as AWLN with rate parameter $\psi > 0$, and the corresponding likelihood $p_{y|z}$ in (5.7) takes the form

$$p_{y|z}(y|z) = \mathcal{L}(y; z, \psi) \triangleq \tfrac{\psi}{2} \exp(-\psi|y - z|), \tag{5.13}$$

and so, for the max-sum case, (R3) in Table 2.1 becomes

$$\hat{z}_m = \arg\min_{z_m \in \mathbb{R}} |z_m - y_m| + \frac{(z_m - \hat{p}_m)^2}{2\mu^p_m \psi}. \tag{5.14}$$

The solution to (5.14) can be recognized as a y_m-shifted version of "soft-thresholding" function, and so the max-sum quantities in (R3) and (R4) of Table 2.1 become, using

$\tilde{p}_m \triangleq \hat{p}_m - y_m$,

$$\hat{z}_m = \begin{cases} \hat{p}_m - \psi\mu_m^p & \tilde{p}_m \geq \psi\mu_m^p \\ \hat{p}_m + \psi\mu_m^p & \tilde{p}_m \leq -\psi\mu_m^p \\ y_m & \text{else} \end{cases} \qquad m = 1, \dots, M, \tag{5.15}$$

$$\mu_m^z = \begin{cases} 0 & |\tilde{p}_m| \leq \psi\mu_m^p \\ \mu_m^p & \text{else} \end{cases} \qquad m = 1, \dots, M. \tag{5.16}$$

Meanwhile, as shown in Appendix B.2.1, the sum-product GAMP quantities (R5) and (R6) (i.e., the mean and variance of the GAMP approximated z_m posterior (2.4)) become

$$\hat{z}_m = y_m + \frac{\underline{C}_m}{C_m}\left(\underline{p}_m - \sqrt{\mu_m^p}h(\underline{\kappa}_m)\right) + \frac{\overline{C}_m}{C_m}\left(\overline{p}_m + \sqrt{\mu_m^p}h(\overline{\kappa}_m)\right) \tag{5.17}$$

$$\begin{aligned} \mu_m^z = {} & \frac{\underline{C}_m}{C_m}\left(\mu_m^p g(\underline{\kappa}_m) + \left(\underline{p}_m - \sqrt{\mu_m^p}h(\underline{\kappa}_m)\right)^2\right) \\ & + \frac{\overline{C}_m}{C_m}\left(\mu_m^p g(\overline{\kappa}_m) + \left(\overline{p}_m + \sqrt{\mu_m^p}h(\overline{\kappa}_m)\right)^2\right) - (y_m - \hat{z}_m)^2, \end{aligned} \tag{5.18}$$

where $\underline{p}_m \triangleq \tilde{p}_m + \psi\mu_m^p$, $\overline{p}_m \triangleq \tilde{p}_m - \psi\mu_m^p$,

$$\underline{C}_m \triangleq \tfrac{\psi}{2}\exp\left(\psi\tilde{p}_m + \tfrac{1}{2}\psi^2\mu_m^p\right)\Phi_c(\underline{\kappa}_m) \tag{5.19}$$

$$\overline{C}_m \triangleq \tfrac{\psi}{2}\exp\left(-\psi\tilde{p}_m + \tfrac{1}{2}\psi^2\mu_m^p\right)\Phi_c(\overline{\kappa}_m), \tag{5.20}$$

$C_m \triangleq \underline{C}_m + \overline{C}_m$, $\underline{\kappa}_m \triangleq \underline{p}_m/\sqrt{\mu_m^p}$, $\overline{\kappa}_m \triangleq -\overline{p}_m/\sqrt{\mu_m^p}$ and

$$h(a) \triangleq \frac{\varphi(a)}{\Phi_c(a)} \tag{5.21}$$

$$g(a) \triangleq 1 - h(a)\big(h(a) - a\big). \tag{5.22}$$

5.3 Non-Negative GAMP

5.3.1 NN Least Squares GAMP

We first detail the NNLS-GAMP algorithm, which uses max-sum GAMP to solve the $\lambda = 0$ case of (5.2). Noting that the $\boldsymbol{x} \geq \mathbf{0}$ constraint in (5.2) can be thought of as

adding an infinite penalty to the quadratic term when any $x_n < 0$ and no additional penalty otherwise, we model the elements of $\boldsymbol{x}$ as i.i.d random variables with the (improper) NN prior pdf

$$p_{\mathsf{x}}(x) = \begin{cases} 1 & x \geq 0 \\ 0 & x < 0 \end{cases}, \tag{5.23}$$

and we assume the augmented model (5.7) with AWGN likelihood (5.10) (of variance $\psi = 1$), in which case max-sum GAMP performs the unconstrained optimization

$$\arg\min_{\boldsymbol{x}} -\sum_{n=1}^{N} \ln \mathbb{1}_{x_n \geq 0} - \ln \mathbb{1}_{\boldsymbol{B}\boldsymbol{x}=\boldsymbol{c}} + \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2, \tag{5.24}$$

where $\mathbb{1}_A \in \{0, 1\}$ is the indicator function of the event A. Hence, (5.24) is equivalent to the constrained optimization (5.2) when $\lambda = 0$.

Under the i.i.d NN uniform prior (5.23), it is readily shown that the max-sum GAMP steps (R11) and (R12) become

$$\hat{x}_n = \begin{cases} 0 & \hat{r}_n \leq 0 \\ \hat{r}_n & \hat{r}_n > 0 \end{cases}, \tag{5.25}$$

$$\mu_n^x = \begin{cases} 0 & \hat{r}_n \leq 0 \\ \mu_n^r & \hat{r}_n > 0 \end{cases}. \tag{5.26}$$

5.3.2 NN LASSO GAMP

Next we detail the NNL-GAMP algorithm, which uses max-sum GAMP to solve the $\lambda > 0$ case of (5.2). For this, we again employ the augmented model (5.7) and AWGN likelihood (5.10) (with variance ψ), but we now use i.i.d exponential x_n, i.e.,

$$p_{\mathsf{x}}(x) = \begin{cases} \chi \exp(-\chi x) & x \geq 0 \\ 0 & \text{else} \end{cases} \tag{5.27}$$

for $\chi > 0$. With these priors and the augmented observation model (5.6), NNL-GAMP solves the optimization problem

$$\hat{\boldsymbol{x}} = \arg\min_{\boldsymbol{x} \geq 0} \tfrac{1}{2\psi}\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \chi\|\boldsymbol{x}\|_1 \;\text{ s.t. }\; \boldsymbol{B}\boldsymbol{x} = \boldsymbol{c}, \tag{5.28}$$

which reduces to (5.2) under $\lambda = \chi\psi$.

It is then straightforward to show that the max-sum lines (R11) and (R12) in Table 2.1 reduce to

$$\hat{x}_n = \begin{cases} \hat{r}_n - \chi\mu_n^r & \hat{r}_n \geq \chi\mu_n^r \\ 0 & \text{else} \end{cases} \tag{5.29}$$

$$\mu_n^x = \begin{cases} \mu_n^r & \hat{r}_n \geq \chi\mu_n^r \\ 0 & \text{else} \end{cases}. \tag{5.30}$$

5.3.3 NN Gaussian Mixture GAMP

Finally, we detail the NNGM-GAMP algorithm, which employs sum-product GAMP under the i.i.d Bernoulli non-negative Gaussian mixture (NNGM) prior pdf for $\boldsymbol{x}$, i.e.,

$$p_{\mathsf{x}}(x) = (1-\tau)\delta(x) + \tau\sum_{\ell=1}^{L}\omega_\ell\mathcal{N}_+(x;\theta_\ell,\phi_\ell), \tag{5.31}$$

where $\mathcal{N}_+(\cdot)$ denotes the non-negative Gaussian pdf,

$$\mathcal{N}_+(x;\theta,\phi) = \begin{cases} \frac{\mathcal{N}(x;\theta,\phi)}{\Phi_c(-\theta/\sqrt{\phi})} & x \geq 0 \\ 0 & x < 0 \end{cases}, \tag{5.32}$$

$\tau \in (0,1]$ is the sparsity rate, and ω_ℓ, θ_ℓ, and ϕ_ℓ are the weight, location, and scale, respectively, of the ℓ^{th} mixture component. For now, we treat the NNGM parameters $[\tau, \boldsymbol{\omega}, \boldsymbol{\theta}, \boldsymbol{\phi}]$ and the model order L as fixed and known.

As shown in Appendix B.3.1, the sum-product GAMP quantities in (R13) and (R14) of Table 2.1 then become

$$\hat{x}_n = \frac{\tau}{\zeta_n}\sum_{\ell=1}^{L}\beta_{n,\ell}\Big(\gamma_{n,\ell} + \sqrt{\nu_{n,\ell}}h(\alpha_{n,\ell})\Big) \tag{5.33}$$

$$\mu_n^x = \frac{\tau}{\zeta_n}\sum_{\ell=1}^{L}\beta_{n,\ell}\Big(\nu_{n,\ell}g(\alpha_{n,\ell}) + \Big(\gamma_{n,\ell} + \sqrt{\nu_{n,\ell}}h(\alpha_{n,\ell})\Big)^2\Big) - \hat{x}_n^2, \tag{5.34}$$

where ζ_n is the normalization factor

$$\zeta_n \triangleq (1-\tau)\mathcal{N}(0;\hat{r}_n,\mu_n^r) + \tau\sum_{\ell=1}^{L}\beta_{n,\ell}, \tag{5.35}$$

$h(\cdot)$ and $g(\cdot)$ were defined in (5.21) and (5.22), respectively, and

$$\alpha_{n,\ell} \triangleq \frac{-\gamma_{n,\ell}}{\sqrt{\nu_{n,\ell}}} \tag{5.36}$$

$$\gamma_{n,\ell} \triangleq \frac{\hat{r}_n/\mu_n^r + \theta_\ell/\phi_\ell}{1/\mu_n^r + 1/\phi_\ell}, \tag{5.37}$$

$$\nu_{n,\ell} \triangleq \frac{1}{1/\mu_n^r + 1/\phi_\ell} \tag{5.38}$$

$$\beta_{n,\ell} \triangleq \frac{\omega_\ell \mathcal{N}(\hat{r}_n;\theta_\ell,\mu_n^r+\phi_\ell)\Phi_c(\alpha_{n,\ell})}{\Phi_c(-\theta_\ell/\sqrt{\phi_\ell})}. \tag{5.39}$$

From (2.5) and (5.31), it follows that GAMP's approximation to the posterior activity probability $\Pr\{\mathsf{x}_n \neq 0 \,|\, \boldsymbol{y}\}$ is

$$\pi_n = \frac{1}{1+\left(\frac{\tau}{1-\tau}\frac{\sum_{\ell=1}^{L}\beta_{n,\ell}}{\mathcal{N}(0;\hat{r}_n,\mu_n^r)}\right)^{-1}}. \tag{5.40}$$

5.4 EM learning of the prior parameters

In the sequel, we will use $\boldsymbol{q}$ to refer to the collection of prior parameters. For example, if NNGM-GAMP was used with the AWGN observation model, then $\boldsymbol{q} = [\tau, \boldsymbol{\omega}, \boldsymbol{\theta}, \boldsymbol{\phi}, \psi]$. Since the value of $\boldsymbol{q}$ that best fits the true data is typically unknown, we propose to learn it using an EM procedure [31]. The EM algorithm is an iterative technique that is guaranteed to converge to a local maximum of the likelihood $p(\boldsymbol{y};\boldsymbol{q})$.

To understand the EM algorithm, it is convenient to write the log-likelihood as [34]

$$\ln p(\boldsymbol{y};\boldsymbol{q}) = \mathcal{Q}_{\hat{p}}(\boldsymbol{y};\boldsymbol{q}) + D\Big(\hat{p}\,||\,p_{\boldsymbol{x}|\boldsymbol{y}}(\cdot|\boldsymbol{y};\boldsymbol{q})\Big), \tag{5.41}$$

where $\hat{p}$ is an arbitrary distribution on $\boldsymbol{x}$, $D\big(\hat{p}\,||\,\hat{q}\big)$ is the Kullback-Leibler (KL) divergence between $\hat{p}$ and $\hat{q}$, and

$$\mathcal{Q}_{\hat{p}}(\boldsymbol{y};\boldsymbol{q}) \triangleq E_{\hat{p}}\{\ln p_{\boldsymbol{x},\boldsymbol{y}}(\boldsymbol{x},\boldsymbol{y};\boldsymbol{q})\} + H(\hat{p}), \tag{5.42}$$

where $H(\hat{p})$ is the entropy of $\boldsymbol{x} \sim \hat{p}$. Importantly, the non-negativity of KL divergence implies that $\mathcal{Q}_{\hat{p}}(\boldsymbol{y};\boldsymbol{q})$ is a lower bound on (5.41). Starting from the initialization $\boldsymbol{q}^0$, the EM algorithm iteratively improves its estimate $\boldsymbol{q}^i$ at each iteration $i \in \mathbb{N}$: first, it assigns $\hat{p}^i(\cdot) = p_{\boldsymbol{x}|\boldsymbol{y}}(\cdot|\boldsymbol{y};\boldsymbol{q}^i)$ to tighten the bound, and then it sets $\boldsymbol{q}^{i+1}$ to maximize (5.42) with $\hat{p} = \hat{p}^i$.

Since the exact posterior pdf $p_{\boldsymbol{x}|\boldsymbol{y}}(\cdot|\boldsymbol{y};\boldsymbol{q}^i)$ is difficult to calculate, in its place we use GAMP's approximate posterior $\prod_n p_{\mathsf{x}_n|\mathsf{r}_n}(\cdot|\hat{r}_n;\mu_n^r;\boldsymbol{q}^i)$ from (2.5), resulting in the EM update

$$\boldsymbol{q}^{i+1} = \arg\max_{\boldsymbol{q}} \hat{\mathrm{E}}\{\ln p(\boldsymbol{x},\boldsymbol{y};\boldsymbol{q}) \,|\, \boldsymbol{y};\boldsymbol{q}^i\}, \tag{5.43}$$

where $\hat{\mathrm{E}}$ denotes expectation using GAMP's approximate posterior. Also, because calculating the joint update for $\boldsymbol{q}$ in (5.43) can be difficult, we perform the maximization (5.43) one component at a time, known as "incremental EM" [61]. Note that, even when using an approximate posterior and updating incrementally, the EM algorithm iteratively maximizes a lower-bound to the log-likelihood.

Whereas [34] proposed the use of (5.43) to tune *sum-product* GAMP, where the marginal posteriors $p_{\mathsf{x}_n|\mathsf{r}_n}(\cdot|\hat{r}_n;\mu_n^r;\boldsymbol{q}^i)$ from (2.5) are computed for use in steps (R13)-(R14) of Table 2.1, we hereby propose the use of (5.43) to tune *max-sum* GAMP. The reasoning behind our proposal goes as follows. Although max-sum GAMP does not compute marginal posteriors (but rather joint MAP estimates), its large-system-limit analysis (under i.i.d sub-Gaussian $\boldsymbol{A}$) [40] shows that $\hat{r}_n(t)$ can be modeled as an

AWGN-corrupted measurement of the true x_n with AWGN variance $\mu_n^r(t)$, revealing the *opportunity* to compute marginal posteriors via (2.5) as an additional step. Doing so enables the use of (5.43) to tune max-sum GAMP.

5.4.1 EM update of AWGN variance

We first derive the EM update of the AWGN noise variance ψ (recall (5.10)). This derivation differs from the one in [34] in that here we use $\boldsymbol{x}$ as the hidden variable (rather than $\boldsymbol{z}$), since experimentally we have observed gains in the low-SNR regime (e.g., SNR < 10 dB). Because we can write $p(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{q}) = D \prod_{m=1}^M p_{y|z}(y_m|\boldsymbol{a}_m^\mathsf{T}\boldsymbol{x}; \psi)$ with a ψ-invariant term D, the incremental update of ψ from (5.43) becomes

$$\psi^{i+1} = \arg\max_{\psi>0} \sum_{m=1}^M \hat{\mathrm{E}}\Big\{ \ln p_{y|z}(y_m|\boldsymbol{a}_m^\mathsf{T}\boldsymbol{x}; \psi) \,\Big|\, \boldsymbol{y}; \psi^i \Big\}. \tag{5.44}$$

In Appendix B.1, we show that (5.44) reduces to

$$\psi^{i+1} = \frac{1}{M}\|\boldsymbol{y} - \boldsymbol{A}\hat{\boldsymbol{x}}\|_2^2 + \frac{1}{M}\sum_{m=1}^M \sum_{n=1}^N a_{mn}^2 \mu_n^x. \tag{5.45}$$

5.4.2 EM update of Laplacian rate parameter

As in the AWGN case above, the incremental update of the Laplacian rate ψ from (5.43) becomes

$$\psi^{i+1} = \arg\max_{\psi>0} \sum_{m=1}^M \hat{\mathrm{E}}\Big\{ \ln p_{y|z}(y_m|\boldsymbol{a}_m^\mathsf{T}\boldsymbol{x}; \psi) \,\Big|\, \boldsymbol{y}; \psi^i \Big\}, \tag{5.46}$$

but where now $p_{y|z}$ is given by (5.13). In Appendix B.2.2, we show that (5.46) reduces to

$$\psi^{i+1} = M \left(\textstyle\sum_{m=1}^M \hat{\mathrm{E}}\Big\{ |\boldsymbol{a}_m^\mathsf{T}\boldsymbol{x} - y_m| \,\Big|\, \boldsymbol{y}; \psi^i \Big\} \right)^{-1} \tag{5.47}$$

where

$$\hat{\mathrm{E}}\Big\{|\boldsymbol{a}_m^{\mathsf{T}}\boldsymbol{x}-y_m|\,\Big|\,\boldsymbol{y};\psi^i\Big\} \approx \Phi_c\left(\frac{\tilde{z}_m}{\mu_m^p}\right)\left(\tilde{z}_m+\sqrt{\mu_m^p}h\left(\frac{-\tilde{z}_m}{\sqrt{\mu_m^p}}\right)\right) - \Phi_c\left(\frac{-\tilde{z}_m}{\mu_m^p}\right)\left(\tilde{z}_m-\sqrt{\mu_m^p}h\left(\frac{\tilde{z}_m}{\sqrt{\mu_m^p}}\right)\right) \tag{5.48}$$

for $\tilde{z}_m \triangleq \boldsymbol{a}_m^{\mathsf{T}}\hat{\boldsymbol{x}} - y_m$, μ_m^p defined in line (R1) of Table 2.1, and $h(\cdot)$ defined in (5.21).

5.4.3 EM update of exponential rate parameter

Noting that $p(\boldsymbol{x},\boldsymbol{y};\boldsymbol{q}) = D\prod_{n=1}^N p_{\mathsf{x}}(x_n;\chi)$ with χ-invariant D, the incremental EM update of the exponential rate parameter χ is

$$\chi^{i+1} = \arg\max_{\chi>0} \sum_{n=1}^N \hat{\mathrm{E}}\Big\{\ln p_{\mathsf{x}}(\mathsf{x}_n;\chi)\,\Big|\,\boldsymbol{y};\chi^i\Big\}, \tag{5.49}$$

$$= \arg\max_{\chi>0} N\log\chi - \chi\sum_{n=1}^N \hat{\mathrm{E}}\Big\{\mathsf{x}_n\,\Big|\,\boldsymbol{y};\chi^i\Big\} \tag{5.50}$$

which, after zeroing the derivative of (5.50) w.r.t. χ, reduces to

$$\chi^{i+1} = N\left(\sum_{n=1}^N \tilde{r}_n + \sqrt{\mu_n^r}h\left(-\frac{\tilde{r}_n}{\sqrt{\mu_n^r}}\right)\right)^{-1} \tag{5.51}$$

for $\tilde{r}_n \triangleq \hat{r}_n - \chi\mu_n^r$, μ_n^r defined in line (R9) of Table 2.1, and $h(\cdot)$ defined in (5.21). The derivation of (5.51) uses the fact that the posterior used for the expectation in (5.50) simplifies to $p_{\mathsf{x}|\mathsf{r}}(x_n|\hat{r}_n;\mu_n^r) = \mathcal{N}_+(x_n;\tilde{r}_n,\mu_n^r)$. Note that this procedure, when used in conjunction with the AWGN variance learning procedure, automatically "tunes" the LASSO regularization parameter λ in (5.2), a difficult problem (see, e.g., [85]).

5.4.4 EM updates for NNGM parameters and model-order selection

Noting that $p(\boldsymbol{x},\boldsymbol{y};\boldsymbol{q}) = D\prod_{n=1}^{N} p_{\mathsf{x}}(x_n;\boldsymbol{\omega},\boldsymbol{\theta},\boldsymbol{\phi})$ with $[\boldsymbol{\omega},\boldsymbol{\theta},\boldsymbol{\phi}]$-invariant D, the incremental EM updates become

$$\theta_k^{i+1} = \underset{\theta_k\in\mathbb{R}}{\arg\max} \sum_{n=1}^{N} \hat{\mathrm{E}}\Big\{ \ln p_{\mathsf{x}}(\mathsf{x}_n;\theta_k,\boldsymbol{q}^i_{\backslash\theta_k}) \,\Big|\, \boldsymbol{y};\boldsymbol{q}^i \Big\}, \tag{5.52}$$

$$\phi_k^{i+1} = \underset{\phi_k>0}{\arg\max} \sum_{n=1}^{N} \hat{\mathrm{E}}\Big\{ \ln p_{\mathsf{x}}(\mathsf{x}_n;\phi_k,\boldsymbol{q}^i_{\backslash\phi_k}) \,\Big|\, \boldsymbol{y};\boldsymbol{q}^i \Big\}, \tag{5.53}$$

$$\boldsymbol{\omega}^{i+1} = \underset{\boldsymbol{\omega}>0:\,\sum_k\omega_k=1}{\arg\max} \sum_{n=1}^{N} \hat{\mathrm{E}}\Big\{ \ln p_{\mathsf{x}}(\mathsf{x}_n;\boldsymbol{\omega},\boldsymbol{q}^i_{\backslash\boldsymbol{\omega}}) \,\Big|\, \boldsymbol{y};\boldsymbol{q}^i \Big\}, \tag{5.54}$$

where we use "$\boldsymbol{q}^i_{\backslash\boldsymbol{\omega}}$" to denote the vector $\boldsymbol{q}^i$ with $\boldsymbol{\omega}$ components removed (and similar for $\boldsymbol{q}^i_{\backslash\theta_k}$ and $\boldsymbol{q}^i_{\backslash\phi_k}$). As derived in Appendix B.3.2, the updates above can be approximated as

$$\theta_k^{i+1} = \frac{\sum_{n=1}^{N}\pi_n\overline{\beta}_{n,k}\Big(\gamma_{n,k}+\sqrt{\nu_{n,k}}h(\alpha_{n,k})\Big)}{\sum_{n=1}^{N}\pi_n\overline{\beta}_{n,k}} \tag{5.55}$$

$$\begin{aligned}\phi_k^{i+1} = &\frac{\sum_{n=1}^{N}\pi_n\overline{\beta}_{n,k}\Big(\gamma_{n,k}+\sqrt{\nu_{n,k}}h(\alpha_{n,k})-\theta_k\Big)^2}{\sum_{n=1}^{N}\pi_n\overline{\beta}_{n,k}}\\ &+\frac{\sum_{n=1}^{N}\pi_n\overline{\beta}_{n,k}\nu_{n,k}g(\alpha_{n,k})}{\sum_{n=1}^{N}\pi_n\overline{\beta}_{n,k}}\end{aligned} \tag{5.56}$$

$$\omega_k^{i+1} = \frac{\sum_{n=1}^{N}\pi_n\overline{\beta}_{n,k}}{\sum_{n=1}^{N}\pi_n}, \tag{5.57}$$

where the quantities $\alpha_{n,\ell},\gamma_{n,\ell},\nu_{n,\ell},\beta_{n,\ell},\pi_n$ were defined in (5.36)-(5.40) and $\overline{\beta}_{n,k} \triangleq \beta_{n,k}/\sum_\ell \beta_{n,\ell}$. The EM update of the NNGM sparsity rate τ (recall (5.31)) is identical to that for the GM sparsity rate derived in [34]:

$$\tau^{i+1} = \frac{1}{N}\sum_{n=1}^{N}\pi_n. \tag{5.58}$$

Since the quantities in (5.55)-(5.58) are already computed by NNGM-GAMP, the EM updates do not significantly increase the complexity beyond that of NNGM-GAMP itself.

The number of components L in the NNGM model (5.31) can be selected using the standard penalized log-likelihood approach to model-order-selection [63], i.e., by maximizing

$$\ln p(\boldsymbol{y}; \hat{\boldsymbol{q}}_L) - \eta(L), \tag{5.59}$$

where $\hat{\boldsymbol{q}}_L$ is the ML estimate of $\boldsymbol{q}$ under the hypothesis L (for which we would use the EM estimate) and $\eta(L)$ is a penalty term such as that given by the Bayesian information criterion (BIC). Since this model-order-selection procedure is identical to that proposed for EM-GM-GAMP in [34], we refer interested readers to [34] for more details. In practice, we find that the fixed choice of $L = 3$ performs sufficiently well (see Chapter 5.5).

5.4.5 EM initialization

With EM, a good initialization is essential to avoiding bad local minima. For EM-NNL-GAMP, we suggest setting the initial exponential rate parameter $\chi^0 = 10^{-2}$, as this seems to perform well over a wide range of problems (see Chapter 5.5).

For EM-NNGM-GAMP, we suggest the initial sparsity rate

$$\tau^0 = \min\left\{\tfrac{M}{N}\rho_{\text{SE}}(\tfrac{M}{N}), 1-\epsilon\right\} \tag{5.60}$$

where $\epsilon > 0$ is set arbitrarily small and $\rho_{\text{SE}}(\cdot)$ is the theoretical noiseless phase-transition-curve (PTC) for ℓ_1 recovery of sparse non-negative signals, shown in [16] to have the closed-form expression

$$\rho_{\mathsf{SE}}(\delta) = \max_{c\geq 0} \frac{1-(1/\delta)[(1+c^2)\Phi(-c) - c\,\varphi(c)]}{1+c^2-[(1+c^2)\Phi(-c) - c\,\varphi(c)]} \tag{5.61}$$

where $\Phi(\cdot)$ and $\varphi(\cdot)$ denote the cdf and pdf of the standard normal distribution. We then propose to set the initial values of the NNGM weights $\{\omega_\ell\}$, locations $\{\theta_\ell\}$,

and scales $\{\phi_\ell\}$ at the values that best fit the uniform pdf on $[0, \sqrt{3\varphi^0}]$, which can be computed offline similar to the standard EM-based approach described in [22, p. 435]. Under the AWGN model (5.10), we propose to set the initial variance of the noise and signal, respectively, as

$$\psi^0 = \frac{\|\boldsymbol{y}\|_2^2}{(\mathsf{SNR}+1)M}, \ \varphi^0 = \frac{\|\boldsymbol{y}\|_2^2 - M\psi^0}{||\boldsymbol{A}||_F^2 \tau^0}, \tag{5.62}$$

where, without knowledge of the true $\mathsf{SNR} \triangleq \|\boldsymbol{A}\boldsymbol{x}\|_2^2/\|\boldsymbol{w}\|_2^2$, we suggest using the value $\mathsf{SNR}=100$. Meanwhile, under the i.i.d Laplacian noise model (5.13), we suggest to initialize the rate as $\psi^0 = 1$ and φ^0 again as in (5.62).

5.5 Numerical Results

The subsections below describe numerical experiments used to ascertain the performance of the proposed methods[28] to existing methods for non-negative signal recovery.

5.5.1 Validation of NNLS-GAMP and NNL-GAMP

We first examine the performance of our proposed algorithms on the linearly constrained NNLS problem (5.2) with $\lambda=0$. In particular, we compare the performance of NNLS-GAMP to Matlab's solver `lsqlin`. To do this, we drew realizations of K-sparse simplex $\boldsymbol{x} \in \Delta_+^N$, where the nonzero elements $\{\underline{x}_k\}_{k=1}^K$ were placed uniformly at random and drawn from a symmetric Dirichlet distribution with concentration a, i.e.,

$$p(\underline{x}_1, \ldots, \underline{x}_{K-1}) = \begin{cases} \frac{\Gamma(aK)}{\Gamma(a)^K} \prod_{k=1}^K \underline{x}_k^{a-1}, & \underline{x}_k \in [0,1] \\ 0 & \text{else} \end{cases} \tag{5.63a}$$

$$p(\underline{x}_K | \underline{x}_1, \ldots, \underline{x}_{K-1}) = \delta(1 - \underline{x}_1 - \cdots - \underline{x}_K), \tag{5.63b}$$

[28] We implemented the proposed algorithms using the GAMPmatlab [48] package available at http://sourceforge.net/projects/gampmatlab/.

		$N = 100$			$N = 250$			$N = 500$		
			time			time			time	
		$\overline{\mathsf{NMSE}}$	NNLS-GAMP	`lsqlin`	$\overline{\mathsf{NMSE}}$	NNLS-GAMP	`lsqlin`	$\overline{\mathsf{NMSE}}$	NNLS-GAMP	`lsqlin`
SNR	10	-161.8	0.068	**0.050**	-161.8	**0.080**	0.550	-161.8	**0.159**	5.414
	100	-161.7	0.069	**0.021**	-154.3	**0.080**	0.205	-161.5	**0.154**	1.497
	1000	-162.1	0.068	**0.011**	-161.7	0.079	**0.074**	-161.5	**0.151**	0.504

Table 5.1: NNLS-GAMP vs. `lsqlin`: average comparative $\overline{\mathsf{NMSE}}$ [dB] and runtime [sec] for simplex signal recovery.

where $\Gamma(\cdot)$ is the gamma function. For this first experiment, we used $a=1$, in which case $\{\underline{x}_k\}_{k=1}^{K-1}$ are i.i.d uniform on $[0,1]$, as well as $K = N$ (i.e., no sparsity). We then constructed noisy measurements $\boldsymbol{y} \in \mathbb{R}^M$ according to (5.1) using $\boldsymbol{A}$ with i.i.d $\mathcal{N}(0, M^{-1})$ entries, $\mathsf{SNR} \triangleq \|\boldsymbol{A}\boldsymbol{x}\|_2^2/\|\boldsymbol{w}\|_2^2 = [10, 100, 1000]$, and sampling ratio $M/N = 3$. Table 5.1 reports the resulting *comparative* $\overline{\mathsf{NMSE}} \triangleq \|\hat{\boldsymbol{x}}_{\mathsf{NNLS\text{-}GAMP}} - \hat{\boldsymbol{x}}_{\texttt{lsqlin}}\|_2^2/\|\boldsymbol{x}\|_2^2$ and runtime averaged over $R = 100$ realizations for signal lengths $N = [100, 250, 500]$. From the table, we see that NNLS-GAMP and `lsqlin` return identical solutions (up to algorithmic tolerance[29]), but that NNLS-GAMP's runtime scales like $O(N^2)$ while `lsqlin`'s scales like $O(N^3)$, making NNLS-GAMP much faster for larger problem dimensions N. Moreover, we see that NNLS-GAMP's runtime is invariant to SNR, whereas `lsqlin`'s runtime quickly degrades as the SNR decreases.

Next, we examine the performance of our proposed algorithms on the non-negative LASSO problem (5.2) with $\lambda > 0$. In particular, we compare NNL-GAMP to TFOCS[30] [88]. For this, K-sparse non-negative $\boldsymbol{x}$ and noisy observations $\boldsymbol{y}$ were

[29]The algorithms under test include user-adjustable stopping tolerances. As these tolerances are decreased, we observe that the comparative $\overline{\mathsf{NMSE}}$ also decreases, at least down to Matlab's numerical precision limit.

[30]We used Matlab code from http://cvxr.com/tfocs/download/.

		$K=50$			$K=100$			$K=150$		
			time			time			time	
		NMSE	NNL-GAMP	TFOCS	NMSE	NNL-GAMP	TFOCS	NMSE	NNL-GAMP	TFOCS
$\prec$	0.01	-135.7	**0.024**	0.091	-139.9	**0.025**	0.119	-140.8	**0.025**	0.104
	0.001	-125.4	**0.026**	0.130	-122.9	**0.026**	0.148	-117.0	**0.027**	0.175
	0.0001	-113.2	**0.035**	0.256	-113.4	**0.036**	0.262	-112.4	**0.036**	0.292

Table 5.2: NNL-GAMP vs. TFOCS: average comparative $\overline{\mathsf{NMSE}}$ [dB] and runtime [sec] for K-sparse non-negative signal recovery.

constructed as before, but now with $M=1000$, $N=500$, $K<N$, and $\mathsf{SNR}=20$ dB. Table 5.2 shows the runtimes and comparative $\overline{\mathsf{NMSE}}$ between NNL-GAMP and TFOCS for various combinations of sparsity K and regularization weight λ. Table 5.2 shows that the solutions returned by the two algorithms were identical (up to algorithmic tolerance) but that NNL-GAMP ran about 4 to 8 times faster than TFOCS.

5.5.2 Noiseless Empirical Phase Transitions

It has been established (see, e.g., [16]) that, for the recovery of a non-negative K-sparse signal $\boldsymbol{x} \in \mathbb{R}^N$ from noiseless observations $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} \in \mathbb{R}^M$, there exists a sharp phase-transition separating problem sizes (M, N, K) that are perfectly solvable (with very high probability) from those that are not. The precise location of the phase-transition curve (PTC) differs among algorithms, presenting an avenue for comparison.

Below, we present empirical PTCs for the recovery of K-sparse N-length simplex signals from M noiseless measurements. To compute each PTC, we fixed $N=500$ and constructed a 20×20 uniformly spaced grid on the $\frac{M}{N}$-versus-$\frac{K}{M}$ plane for $\frac{M}{N} \in [0.05, 1]$ and $\frac{K}{M} \in [0.05, 1]$. At each grid point, we drew $R=100$ independent realizations of the pair $(\boldsymbol{A}, \boldsymbol{x})$, where $\boldsymbol{A}$ was drawn from i.i.d $\mathcal{N}(0, M^{-1})$ entries and $\boldsymbol{x} \in \mathbb{R}^N$ had

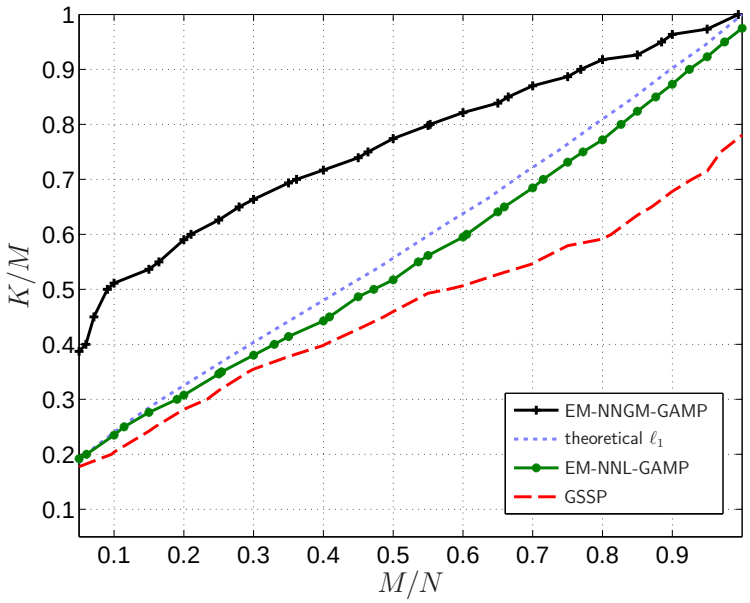


Figure 5.1: Empirical PTCs and ℓ_1-SNN theoretical PTC for noiseless recovery of length-$N=500$, K-sparse, simplex signals with Dirichlet concentration $a=1$ from M measurements.

K nonzero elements $\{\underline{x}_k\}_{k=1}^K$ (placed uniformly at random) drawn from a symmetric Dirichlet distribution (5.63) with concentration parameter a. For the r^{th} realization of $(\boldsymbol{A},\boldsymbol{x})$, we attempted to recover non-negative sparse $\boldsymbol{x}$ from the augmented observations $\left[\begin{smallmatrix}\boldsymbol{y}\\1\end{smallmatrix}\right]=\left[\begin{smallmatrix}\boldsymbol{A}\\\boldsymbol{1}^\mathsf{T}\end{smallmatrix}\right]\boldsymbol{x}$, which implicitly enforce the simplex constraint. The resulting recovery $\hat{\boldsymbol{x}}$ was considered to be "successful" if $\mathsf{NMSE}\triangleq\|\boldsymbol{x}-\hat{\boldsymbol{x}}\|_2^2/\|\boldsymbol{x}\|_2^2<10^{-6}$. Using $S_r=1$ to record a success and $S_r=0$ a failure, the average success rate was then computed as $\overline{S}\triangleq\frac{1}{R}\sum_{r=1}^R S_r$, and the corresponding empirical PTC was plotted as the $\overline{S}=0.5$ level-curve using Matlab's `contour` command.

Figures 5.1 and 5.2 show the empirical PTCs under the Dirichlet concentration $a=1$ (i.e., i.i.d uniform $\{\underline{x}_k\}_{k=1}^{K-1}$) and $a=100$ (i.e., $\underline{x}_k\approx\frac{1}{K}\ \forall k$), respectively, for our proposed EM-tuned NNGM-GAMP and NNL-GAMP algorithms, in comparison

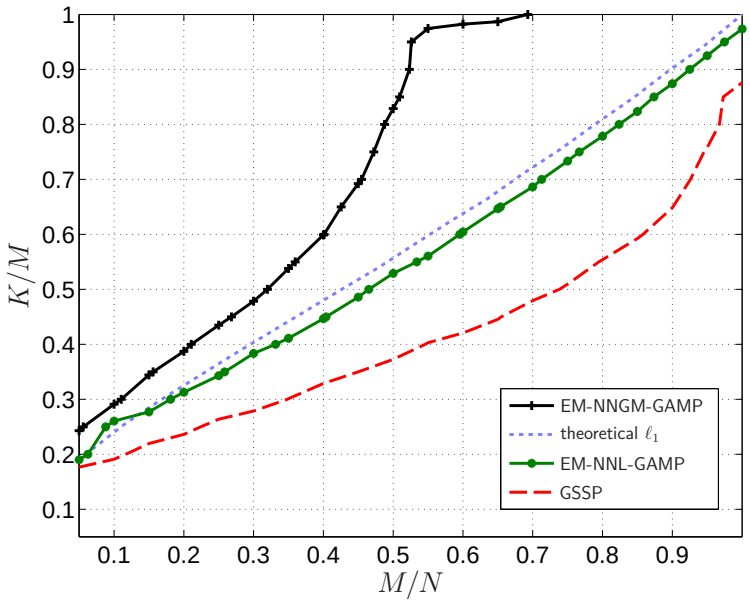


Figure 5.2: Empirical PTCs and ℓ_1-SNN theoretical PTC for noiseless recovery of length-$N=500$, K-sparse, simplex signals with Dirichlet concentration $a = 100$ from M measurements.

to the GSSP[31] approach (5.3) proposed in [78]. We did not consider NNLS-GAMP and `lsqlin` because, for $\boldsymbol{A}$ drawn i.i.d Gaussian, the solution to the non-negative LS problem "$\arg\min_{\boldsymbol{x}\geq\boldsymbol{0}} \|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|_2^2$" is not guaranteed to be unique when $M < N$ [86, Thm. 1], which is the setting considered here. Figures 5.1 and 5.2 also show $\rho_{\mathsf{SE}}(\frac{M}{N})$ from (5.61), i.e., the theoretical large-system-limit PTC for ℓ_1-based recovery of sparse non-negative (SNN) signals.

Looking at Figures 5.1 and 5.2, we see that the empirical PTCs of EM-NNL-GAMP are close to the theoretical ℓ_1 PTC, as expected, and significantly better than those of GSSP. More striking is the far superior PTCs of EM-NNGM-GAMP. We attribute EM-NNGM-GAMP's success to three factors: i) the generality of the

[31]For GSSP, we used code provided by its authors, but found that its performance was greatly enhanced by initializing the algorithm at the Basis Pursuit solution (as computed by SPGL1 [70]) and using the stepsize $100/\|\boldsymbol{A}\|_F^2$.

NNGM prior (5.31), ii) the ability of the proposed EM approach to accurately learn the prior parameters, and iii) the ability of sum-product GAMP to exploit the learned prior. In fact, Fig. 5.2 shows EM-NNGM-GAMP reliably reconstructing K-sparse signals from only $M=K$ measurements in the compressive (i.e., $M<N$) regime.

5.5.3 Sparse Non-negative Compressive Imaging

As a practical example, we experimented with the recovery of a sparse non-negative image. For this, we used the $N=256\times 256$ satellite image shown on the left of Fig. 5.3, which contained $K=6678$ nonzero pixels and $N-K=58858$ zero-valued pixels, and thus was approximately 10% sparse. Measurements $\boldsymbol{y}=\boldsymbol{A}\boldsymbol{x}+\boldsymbol{w}\in\mathbb{R}^M$ were collected under i.i.d Gaussian noise $\boldsymbol{w}$ whose variance was selected to achieve an $\mathsf{SNR}=60$ dB. Here, $\boldsymbol{x}$ represents the (rasterized) image and $\boldsymbol{A}$ a linear measurement operator configured as $\boldsymbol{A}=\boldsymbol{\Phi}\boldsymbol{\Psi}\boldsymbol{S}$, where $\boldsymbol{\Phi}\in\{0,1\}^{M\times N}$ was constructed from rows of the $N\times N$ identity matrix selected uniformly at random, $\boldsymbol{\Psi}\in\{-1,1\}^{N\times N}$ was a Hadamard transform, and $\boldsymbol{S}\in\mathbb{R}^{N\times N}$ was a diagonal matrix with ± 1 diagonal entries chosen uniformly at random. Note that multiplication by $\boldsymbol{A}$ can be executed using a fast binary algorithm, making it attractive for hardware implementation. For this experiment, no linear equality constraints exist and so the observation model was not augmented as in (5.6).

As a function of the sampling ratio $\frac{M}{N}$, Fig. 5.4 shows the NMSE and runtime averaged over $R=100$ realizations of $\boldsymbol{A}$ and $\boldsymbol{w}$ for the proposed EM-NNGM-GAMP and EM-NNL-GAMP in comparison to EM-GM-GAMP from [34], genie-tuned non-negative LASSO via TFOCS [88],[32] and genie-tuned standard LASSO implemented

[32]Using EM-NNL-GAMP's $\hat{\boldsymbol{x}}$, we ran TFOCS over an 11-point grid of hypothesized non-negative ℓ_1 penalty $\lambda\in\{0.5\|\boldsymbol{A}^{\mathsf{T}}(\boldsymbol{y}-\boldsymbol{A}\hat{\boldsymbol{x}})\|_\infty,\ldots,2\|\boldsymbol{A}^{\mathsf{T}}(\boldsymbol{y}-\boldsymbol{A}\hat{\boldsymbol{x}})\|_\infty\}$ and then reported the total runtime and best NMSE.

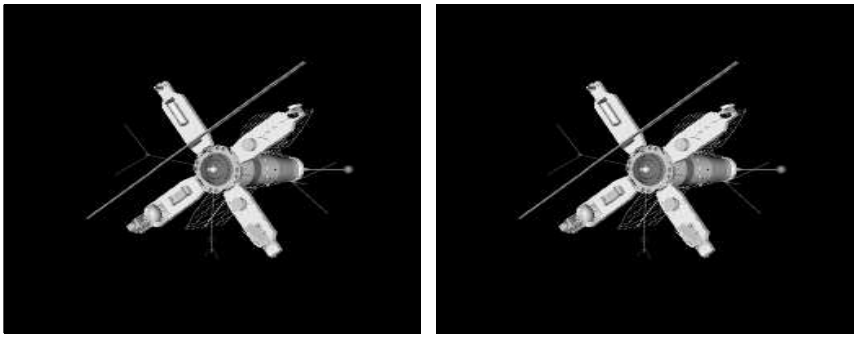


Figure 5.3: Sparse non-negative image of a satellite: original image on left and EM-NNGM-GAMP recovery at $\frac{M}{N} = \frac{1}{4}$ on right.

via SPGL1[33] [70]. NNLS methods were not considered because of the non-uniqueness of their solutions in the $M < N$ regime (recall [86, Thm. 1]).

Figure 5.4 shows that the proposed EM-NNGM-GAMP algorithm provided the most accurate signal recoveries for all undersampling ratios. Remarkably, its phase-transition occurred at $\frac{M}{N} \approx 0.25$, whereas that of the other algorithms occurred at $\frac{M}{N} \approx 0.35$. The gain of EM-NNGM-GAMP over EM-GM-GAMP can be attributed to the former's exploitation of signal non-negativity, whereas the gain of EM-NNGM-GAMP over non-negative LASSO (either via EM-NNL-GAMP or genie-tuned TFOCS) can be attributed to former's learning/exploitation of the true signal distribution. Finally, the gain of non-negative LASSO over standard LASSO can be attributed to the former's exploitation of signal non-negativity.

Figure 5.4 also demonstrates that the LASSO tuning procedure proposed in Chapter 5.4 works very well: the NMSE of EM-NNL-GAMP is nearly identical to that of oracle-tuned TFOCS for all sampling ratios M/N.

[33] We ran SPGL1 in "BPDN mode," i.e., solving $\min_{\boldsymbol{x}} \|\boldsymbol{x}\|_1$ s.t. $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2 < \sigma$ for hypothesized tolerances $\sigma^2 \in \{0.3, 0.6, \ldots, 1.5\} \times M\psi$ and then reported the total runtime and best NMSE.

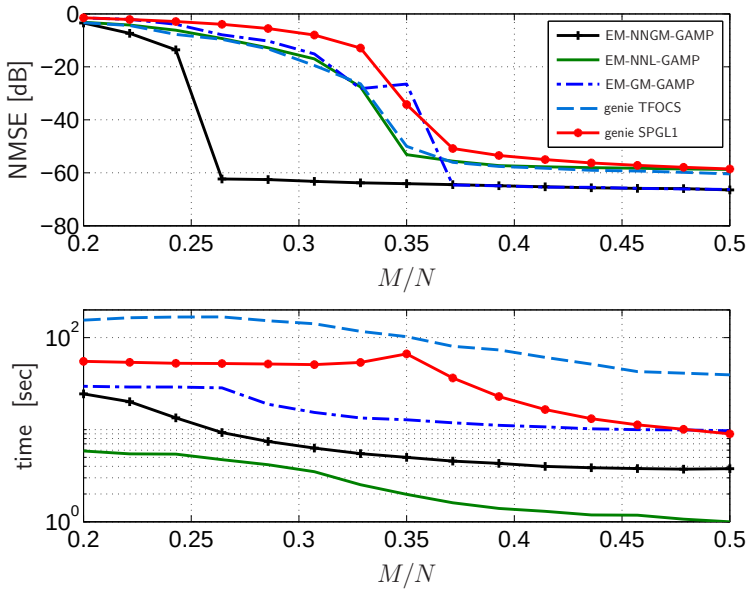


Figure 5.4: Recovery NMSE (top) and runtime (bottom) versus $\frac{M}{N}$ for the sparse NN satellite image for the proposed EM-NNGM-GAMP and EM-NNL-GAMP compared to EM-GM-GAMP, non-negative LASSO via oracle-tuned TFOCS, and standard LASSO via oracle-tuned SPGL1.

Finally, Fig. 5.4 shows that EM-NNGM-GAMP was about 3 times as fast as EM-GM-GAMP, between 3 to 15 times as fast as SPGL1 (implementing standard LASSO), and between 10 to 20 times as fast as TFOCS (implementing non-negative LASSO). The proposed EM-NNL-GAMP was about 2 to 4 faster than EM-NNGM-GAMP, although it did not perform as well in terms of NMSE.

5.5.4 Portfolio Optimization

As another practical example, we consider portfolio optimization under the return-adjusted Markowitz mean-variances (MV) framework [75]: if $\boldsymbol{x} \in \Delta_+^N$ is a portfolio and $\boldsymbol{r}_{M+1} \in \mathbb{R}^N$ is a random vector that models the returns of N commodities at the future time $M+1$, then we desire to design $\boldsymbol{x}$ so that the future sum-return $\boldsymbol{r}_{M+1}^{\mathsf{T}}\boldsymbol{x}$ has relatively high mean and low variance. Although $\boldsymbol{r}_{M+1}$ is unknown at design

time, we assume knowledge of the past M returns $\boldsymbol{A} \triangleq [\boldsymbol{r}_1, \ldots, \boldsymbol{r}_M]^\mathsf{T}$, which can be time-averaged to yield $\boldsymbol{\mu} \triangleq \frac{1}{M}\sum_{m=1}^{M} \boldsymbol{r}_m = \frac{1}{M}\boldsymbol{A}^\mathsf{T}\mathbf{1}$, and then (assuming stationarity) design $\boldsymbol{x}$ that minimizes the variance around a target sum-return of ρ, i.e.,

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}\in\Delta_+^N}{\arg\min} \|\mathbf{1}\rho - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_1 \;\; \text{s.t.} \;\; \boldsymbol{\mu}^\mathsf{T}\boldsymbol{x} = \rho. \tag{5.64}$$

In (5.64), the use of sparsity promoting ℓ_1 regularization [76] aims to help the portfolio designed from past data $\{\boldsymbol{r}_m\}_{m=1}^M$ generalize to the future data $\boldsymbol{r}_{M+1}$. Without ℓ_1 regularization, the solutions to (5.64) are often outperformed by the "naïve" portfolio $\boldsymbol{x}_{\mathsf{naïve}} \triangleq \frac{1}{N}\mathbf{1}$ in practice [90].

Noting that (5.64) is a special case of (5.2), MV portfolio optimization is a natural application for the algorithms developed in this chapter. We thus tested our proposed algorithms against[34] `lsqlin` and cross-validated (CV)[35] TFOCS using the FF49 portfolio database,[36] which consists of monthly returns for $N = 49$ securities from July 1971 (i.e., $\boldsymbol{r}_1$) to July 2011 (i.e., $\boldsymbol{r}_{481}$). In particular, starting from July 1981 and moving forward in yearly increments, we collected the past $M = 120$ months of return data in $\boldsymbol{A}(i) \triangleq [\boldsymbol{r}_{12(i-1)+1}, \ldots, \boldsymbol{r}_{12(i-1)+M}]^\mathsf{T}$ and computed the corresponding time-average return $\boldsymbol{\mu}(i) \triangleq \frac{1}{M}\boldsymbol{A}(i)^\mathsf{T}\mathbf{1}$, where $i \in \{1, \ldots, 30\}$ indexed the years from 1981 to 2010. Then, we chose the target sum-return $\rho(i)$ to be that of the naïve scheme, i.e., $\rho(i) = \frac{1}{N}\boldsymbol{\mu}(i)^\mathsf{T}\mathbf{1}$, and computed the portfolio $\hat{\boldsymbol{x}}(i)$ from $\{\boldsymbol{A}(i), \boldsymbol{\mu}(i), \rho(i)\}$ for each algorithm under test. The resulting $\hat{\boldsymbol{x}}(i)$ was evaluated on the *future* $T = 12$

[34] We were not able to configure GSSP in a way that maintained $\boldsymbol{\mu}^\mathsf{T}\hat{\boldsymbol{x}} = \rho$, even approximately, after the simplex projection step in (5.3).

[35] For CV-TFOCS, we used 4-fold cross-validation to tune λ over a 15-point grid between 0.001 and 0.1.

[36] The publicly available FF49 database and other financial datasets can be obtained from http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html.

months of return data using the Sharpe ratio $\mathsf{SR}(i) \triangleq \hat{\rho}(i)/\hat{\sigma}(i)$, where

$$\hat{\rho}(i) \triangleq \frac{1}{T}\sum_{t=1}^{T} \boldsymbol{r}_{12(i-1)+M+t}^{\mathsf{T}} \hat{\boldsymbol{x}}(i), \tag{5.65}$$

$$\hat{\sigma}^2(i) \triangleq \frac{1}{T}\sum_{t=1}^{T} \left(\boldsymbol{r}_{12(i-1)+M+t}^{\mathsf{T}} \hat{\boldsymbol{x}}(i) - \hat{\rho}(i)\right)^2, \tag{5.66}$$

For `lsqlin`, the constraints were specified directly. For NNLS-GAMP, EM-NNL-GAMP, and EM-NNGM-GAMP, the constraints were enforced using (5.6) with $\boldsymbol{B} = [\boldsymbol{\mu}, \mathbf{1}]^{\mathsf{T}}$ and $\boldsymbol{c} = [\rho, 1]^{\mathsf{T}}$, and for CV-TFOCS, the constraints were enforced using the augmentation

$$\bar{\boldsymbol{y}} \triangleq \begin{bmatrix} \rho(i)\mathbf{1} \\ 500\rho(i) \\ 500 \end{bmatrix} \quad \text{and} \quad \bar{\boldsymbol{A}} = \begin{bmatrix} \boldsymbol{A}(i) \\ 500\boldsymbol{\mu}(i)^{\mathsf{T}} \\ 500\,\mathbf{1}^{\mathsf{T}} \end{bmatrix}, \tag{5.67}$$

where the gain of 500 helped to weight the constraints above the loss. Lastly, we tried our GAMP-based approaches using both the AWGN likelihood (5.10) as well as the AWLN likelihood (5.13).

Table 5.3 reports the average Sharpe ratios $\mathsf{SR} \triangleq \frac{1}{30}\sum_{i=1}^{30} \mathsf{SR}(i)$ and runtimes for each algorithm under test. In addition, it reports the average squared constraint error $\mathcal{E} \triangleq \frac{1}{30}\sum_{i=1}^{30} |\boldsymbol{\mu}(i)^{\mathsf{T}}\hat{\boldsymbol{x}}(i) - \rho(i)|^2$, showing that all algorithms near-perfectly met the target sum-return constraint $\boldsymbol{\mu}(i)^{\mathsf{T}}\hat{\boldsymbol{x}}(i) = \rho(i)$. The table shows that Matlab's `lsqlin` and AWGN NNLS-GAMP (which solve the same NNLS problem) yielded identical Sharpe ratios, which were $\approx$ 19% larger than the naïve value. Meanwhile, CV-TFOCS and AWGN EM-NNL-GAMP (which solve the same NN LASSO problem) yielded very similar Sharpe ratios, also $\approx$ 19% larger than the naïve value. As in previous experiments, AWGN EM-NNGM-GAMP outperformed both NNLS and NN LASSO, in this case improving on the naïve Sharpe ratio by 24%. The table also shows that the use of an AWLN likelihood (robust to outliers [89]) resulted in across-the-board improvements in Sharpe ratio. Among the algorithms under test, AWLN

		SR	time (sec)	$\mathcal{E}$ (dB)
	naïve	0.3135	-	-∞
	`lsqlin`	0.3725	**0.06**	-307.4
	CV-TFOCS	0.3747	31.92	-56.9
AWGN	NNLS-GAMP	0.3724	0.68	-72.0
AWGN	EM-NNL-GAMP	0.3725	1.48	-60.9
AWGN	EM-NNGM-GAMP	0.3900	6.98	-41.5
AWLN	NNLS-GAMP	0.3818	1.80	-56.1
AWLN	EM-NNL-GAMP	0.3829	5.14	-43.2
AWLN	EM-NNGM-GAMP	**0.3995**	2.95	-42.3

Table 5.3: Average Sharpe ratio SR, constraint error $\mathcal{E}$ (in dB), and runtime (in sec) versus algorithm for the FF49 dataset.

EM-NNGM-GAMP yielded the best performance, improving the naïve Sharpe ratio by 27%.

In terms of runtimes, Matlab's `lsqlin` was by far the fastest algorithm, CV-TFOCS was by far the slowest, and the AMP approaches were in-between. NNLS-GAMP and NNL-GAMP were slower here than in Table 5.1 and Table 5.2 because the matrix $\boldsymbol{A}$ in this financial experiment had correlated columns and thus required the use of a stronger damping factor in the GAMPmatlab implementation [48].

5.5.5 Hyperspectral Image Inversion

As a final practical example, we consider hyperspectral image inversion [18]. A hyperspectral image is like a color image, but instead of 3 spectral bands (red, green, and blue) it contains $M \gg 3$ spectral bands. With $T = T_1 \times T_2$ spatial pixels, such an image can be represented by a matrix $\boldsymbol{Y} \in \mathbb{R}^{M \times T}$ and, under the macroscopic model, "unmixed" into

$$\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{X} + \boldsymbol{W} \tag{5.68}$$

where the nth column in $\boldsymbol{A} \in \mathbb{R}^{M \times N}$ is the spectral signature (or "endmember") of the nth material present in the scene, the nth row in $\boldsymbol{X} \in \mathbb{R}_{\geq 0}^{N \times T}$ is the spatial abundance of that material, and $\boldsymbol{W}$ is additive noise. The tth column of $\boldsymbol{X}$, henceforth denoted as $\boldsymbol{x}_t$, describes the distribution of materials within the tth pixel, and so for a valid distribution we need $\boldsymbol{x}_t \in \Delta_+^N$. We will assume that the endmembers $\boldsymbol{A}$ have been extracted from $\boldsymbol{Y}$ (e.g., via the well known VCA algorithm [91]) and therefore focus on image inversion, where the goal is to estimate $\boldsymbol{X}$ in (5.68) given $\boldsymbol{Y}$ and $\boldsymbol{A}$. In particular, the goal is to estimate a (possibly sparse) simplex-constrained $\boldsymbol{x}_t$ from the observation $\boldsymbol{y}_t = \boldsymbol{A}\boldsymbol{x}_t + \boldsymbol{w}_t$ at each pixel t.

We evaluated algorithm performance using the SHARE 2012 Avon dataset[37] [92], which uses $M = 360$ spectral bands, corresponding to wavelengths between 400 and 2450 nm, over a large rural scene. To do this, we first cropped down to the scene shown in Fig. 5.5, known to consist primarily of pure grass, dry sand, black felt, and white TyVek [93]. We then extracted the endmembers $\boldsymbol{A}$ from $\boldsymbol{Y}$ using VCA. Finally, we estimated the simplex-constrained columns of $\boldsymbol{X}$ from $(\boldsymbol{Y}, \boldsymbol{A})$ using NNLS-GAMP, EM-NNL-GAMP, EM-NNGM-GAMP, `lsqlin` (known in the hyperspectral literature as "fully constrained least squares" [94]), and GSSP. For both EM-NNL-GAMP and EM-NNGM-GAMP, we opted to learn the prior parameters separately for each *row* of $\boldsymbol{X}$, since the marginal distributions can be expected to differ across materials. For GSSP, we assumed that each pixel was at most $K = 3$-sparse and used a step size of $3/\|\boldsymbol{A}\|_F^2$, as these choices seemed to yield the best results.

[37] The SHARE 2012 Avon dataset can be obtained from http://www.rit.edu/cos/share2012/.

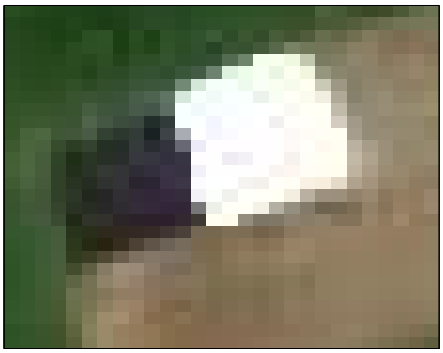
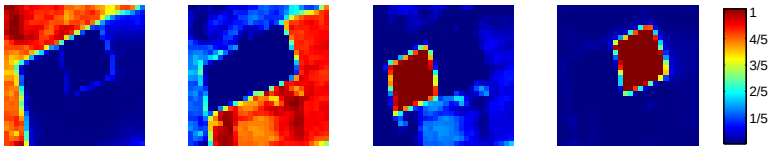


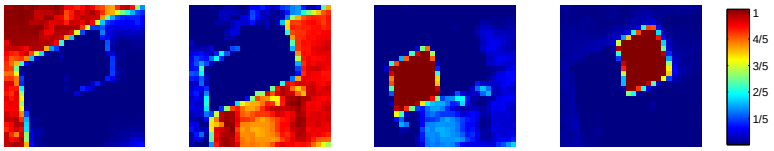
Figure 5.5: RGB image of the cropped scene of the SHARE 2012 dataset [92].

Since we have no knowledge of the true abundances $\boldsymbol{X}$, we are unable to present quantitative results on estimation accuracy. However, a qualitative comparison is made possible using the fact that most pixels in this scene are known to be pure [92] (i.e., contain only one material). In particular, each row of Fig. 5.6 shows the $N=4$ abundance maps recovered by a given algorithm, and we see that all recoveries are nearly pure. However, the recoveries of EM-NNGM-GAMP are the most pure, as evident from the deep blue regions in the first and third columns of Fig. 5.6, as well as the deep red regions in the first and second columns. In terms of runtime, GSSP was by far the slowest algorithm, whereas all the other algorithms were similar (with `lsqlin` beating the others by a small margin).

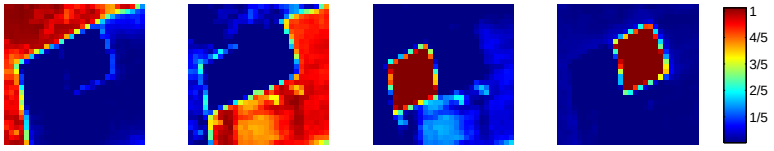
(a) `lsqlin` (runtime = 2.26 sec):



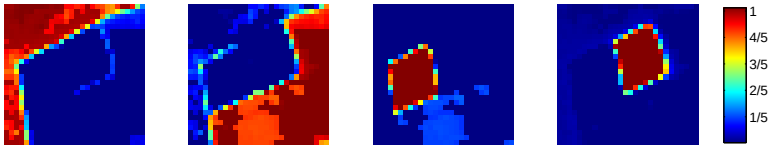
(b) NNLS-GAMP (runtime = 2.84 sec):



(c) EM-NNL-GAMP (runtime = 3.23 sec):



(d) EM-NNGM-GAMP (runtime = 4.37 sec):



(e) GSSP (runtime = 170.71 sec):

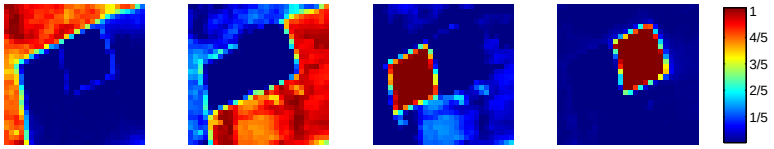


Figure 5.6: Each row shows the $N = 4$ abundance maps estimated by a given algorithm. From left to right, the materials are: grass, dry sand, black felt, and white TyVek. Figure 5.5 shows the RGB image of the same scene.

5.6 Conclusions

The problem of recovering a linearly constrained non-negative sparse signal $\boldsymbol{x}$ from noisy linear measurements $\boldsymbol{y}$ arises in many applications. One approach is to pose a sparsity-inducing convex optimization problem like (5.2) and then apply standard solvers like `lsqlin` (when $\lambda = 0$) or TFOCS (when $\lambda > 0$), although doing so requires also solving the non-trivial problem of optimizing λ [85]. Another approach is to solve for the MMSE estimate of $\boldsymbol{x}$, but doing so is made difficult by the need to estimate the prior distribution of $\boldsymbol{x}$ and then compute the resulting posterior mean.

In this chapter, we proposed new solvers for (5.2) based on the min-sum AMP methodology, yielding NNLS-GAMP (for $\lambda = 0$) and NNL-GAMP (for $\lambda > 0$), and we demonstrated computational advantages relative to standard solvers in the large-N regime. In addition, we proposed a novel EM-based approach to optimizing λ that, in our empirical experiments, worked nearly as well as cross-validation and oracle methods. Moreover, we proposed a new approximate-MMSE estimation scheme that models $\boldsymbol{x}$ using an i.i.d Bernoulli non-negative Gaussian-mixture, learns the distributional parameters via the EM algorithm, and exploits the learned distribution via sum-product AMP. In all of our experiments, the resulting EM-NNGM-GAMP algorithm yielded superior performance while maintaining a reasonable computational efficiency. Finally, for problems where the noise may be non-Gaussian, we developed Laplacian likelihood models for both min-sum and sum-product GAMP, in addition to EM-tuning procedures, and demonstrated performance gains on practical datasets.

Chapter 6: Hyperspectral Unmixing via Turbo Approximate Message Passing

6.1 Introduction

In *hyperspectral unmixing* (HU), the objective is to jointly estimate the radiance spectra and per-pixel abundances of the N materials present in a scene, given measurements across M spectral bands at each of $T = T_1 \times T_2$ pixels.[38] Often, linear mixing [18, 95] is assumed, in which case the measurements $\boldsymbol{Y} \in \mathbb{R}^{M \times T}$ are modeled as

$$\boldsymbol{Y} = \boldsymbol{S}\boldsymbol{A} + \boldsymbol{W}, \tag{6.1}$$

where the nth column of $\boldsymbol{S} \in \mathbb{R}_+^{M \times N}$ represents the spectrum (or "endmember") of the nth material, the nth row of $\boldsymbol{A} \in \mathbb{R}_+^{N \times T}$ represents the spatial abundance of the nth material, and $\boldsymbol{W}$ represents noise. We note that this is analagous to the bilinear obervation model (1.5), noting the change of variables. Both $\boldsymbol{S}$ and $\boldsymbol{A}$ must contain only non-negative (NN) elements, and each column of $\boldsymbol{A}$ must obey the simplex constraint (i.e., NN and sum-to-one).

Traditionally, hyperspectral unmixing is a two-step procedure, consisting of *endmember extraction* (EE) to recover the endmembers followed by *inversion* to recover

[38] This chapter is an extended version of work our work published in [36] and further developed in [37].

the abundances. Many EE algorithms leverage the "*pure pixel*" assumption: for each material, there exists at least one observed pixel containing only that material (i.e., all columns of the $N \times N$ identity matrix can be found among the columns of $\boldsymbol{A}$). Well-known examples of pure-pixel-based EE algorithms include N-FINDR [96] and VCA [91]. The existence of pure pixels in HU is equivalent to "*separability*" in the problem of non-negative matrix factorization (NMF), where the goal is to find $\boldsymbol{S} \in \mathbb{R}_+^{M \times N}$ and $\boldsymbol{A} \in \mathbb{R}_+^{N \times T}$ matching a given $\boldsymbol{Z} = \boldsymbol{S}\boldsymbol{A}$. There, separability has been shown to be sufficient for the existence of unique factorizations [21] and polynomial-time solvers [97], with a recent example being the FSNMF algorithm from [98]. In HU, however, the limited spatial-resolution of hyperspectral cameras implies that the pure-pixel assumption does not always hold in practice. With "mixed pixel" scenarios in mind, algorithms such as Minimum Volume Simplex Analysis (MVSA) [99] and Minimum Volume Enclosing Simplex (MVES) [100] attempt to find the minimum-volume simplex that contains the data $\boldsymbol{Y}$.

In the inversion step, the extracted endmembers in $\hat{\boldsymbol{S}}$ are used to recover the simplex-constrained abundances in $\boldsymbol{A}$. Often this is done by solving [94, 101]

$$\hat{\boldsymbol{A}} = \underset{\boldsymbol{A} \geq 0}{\arg\min} \|\boldsymbol{Y} - \hat{\boldsymbol{S}}\boldsymbol{A}\|_F^2 \text{ s.t. } \mathbf{1}_N^\mathsf{T}\boldsymbol{A} = \mathbf{1}_T^\mathsf{T}, \tag{6.2}$$

where $\mathbf{1}_N$ denotes the $N \times 1$ vector of ones, which is usually referred to as *fully constrained least squares* (FCLS).

Real-world hyperspectral datasets can contain significant structure beyond non-negativity on s_{mn} and simplex constraints on $\{a_{nt}\}_{n=1}^N$. For example, the abundances $\{a_{nt}\}_{n=1}^N$ will be *sparse* if most pixels contain significant contributions from only a small subset of the N materials. Also, the abundances $\{a_{nt}\}_{t=1}^T$ will be *spatially coherent* if the presence of a material in a given pixel makes it more likely for that

same material to exist in neighboring pixels. Likewise, the endmembers $\{s_{mn}\}_{m=1}^{M}$ will be *spectrally coherent* if the radiance values are correlated across frequency.

Various unmixing algorithms have been proposed to leverage these additional structures. For example, given an endmember estimate $\hat{\boldsymbol{S}}$, the SUnSAL algorithm [102] estimates sparse abundances $\boldsymbol{A}$ using ℓ_1-regularized least-squares (LS), and the SUnSAL-TV algorithm [103] adds total-variation (TV) regularization [24] to also penalize changes in abundance across neighboring pixels (i.e., to exploit spatial coherence). SUnSAL and SUnSAL-TV can be categorized as unmixing algorithms, rather than inversion algorithms, since their ℓ_1-regularization supports the use of large (i.e., $N > M$) and scene-independent endmember libraries for $\hat{\boldsymbol{S}}$. However, there are limitations on the size of the library $\hat{\boldsymbol{S}}$, and it can be difficult to determine suitable choices for the ℓ_1 and TV regularization weights.

Bayesian approaches to hyperspectral unmixing have also been proposed. For example, the Bayesian Linear Unmixing (BLU) algorithm [104] employs priors that enforce NN constraints on the endmembers and simplex constraints on the per-pixel abundances, and returns either (approximately) minimum mean-square error (MMSE) or maximum a posteriori (MAP) estimates using Gibbs sampling. The Spatially Constrained Unmixing (SCU) [105] algorithm, an extension of BLU, furthermore exploits spatial coherence using a hierarchical Dirichlet-process prior. Both BLU and SCU have been shown to outperform N-FINDR and VCA-plus-FCLS under certain conditions [105], but at the cost of several orders-of-magnitude increase in runtime.

In this chapter, we propose a novel Bayesian approach to HU that is based on loopy belief propagation. Our approach, referred to as HU turbo-AMP (HUT-AMP),

partitions the factor graph (see Fig. 6.1) into three subgraphs: one that models spectral coherence (using N Gauss-Markov chains), one that models spatial coherence (using N binary Markov Random Fields (MRFs)), and one that models the NN bilinear structure of (6.1). While the first two subgraphs yield inference problems that are handled efficiently by standard methods [106, 107], the third does not. Thus, to perform efficient inference on the bilinear structure subgraph, we apply the recently proposed Bilinear Generalized Approximate Message Passing (BiG-AMP) algorithm [29]. BiG-AMP can be interpreted as an extension of approximate message passing (AMP) techniques [16, 27, 28], originally proposed for the linear observation models that arise in compressive sensing, to bilinear models like (6.1). To merge BiG-AMP-based inference with Markov-chain and MRF-based inference, we leverage the "turbo AMP" approach first proposed in [32] and subsequently applied to joint channel-estimation and decoding [108,109], compressive image retrieval [110,111], and compressive video retrieval [112], all with state-of-the-art results. Furthermore, since the parameters of the various prior distributions are unknown in practice, we use the expectation-maximization (EM) algorithm to automatically *tune* them, building on the NN sparse reconstruction work in [35]. Lastly, when the number of materials N is unknown, we show how it can be accurately estimated using a classical model-order selection (MOS) strategy [63].

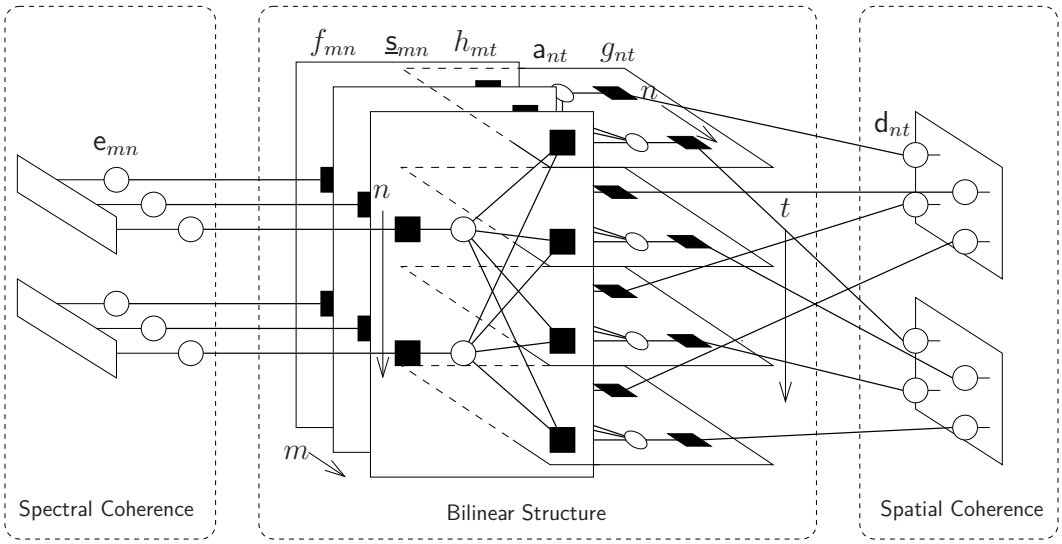


Figure 6.1: The factor graph for HUT-AMP for the toy-problem dimensions $M = 3$, $N = 2$, and $T = 4$. Circles represent random variables and dark squares represent pdf factors. Each elongated bar in the left subgraph conglomerates the factors associated with an M-variable Markov chain (detailed in Fig. 6.2), while each square in the right subgraph conglomerates the factors associated with a $T_1 \times T_2$-pixel Markov random field (detailed in Fig. 6.3).

We evaluate the performance of our proposed technique, in comparison to several recently proposed methods, through a detailed numerical study that includes both synthetic and real-world datasets. The results, presented in Chapter 6.4, suggest that HUT-AMP yields an excellent combination of unmixing performance and computational complexity.

6.2 Signal and Observation Models

6.2.1 Augmented Observation Model

We model the elements of the additive noise matrix $\boldsymbol{W}$ in (6.1) as i.i.d zero-mean Gaussian with variance $\psi > 0$. Thus, the BiG-AMP marginal likelihoods take the

form $p_{\overline{y}_{mt}|\overline{\boldsymbol{z}}_{mt}}(\overline{y}_{mt}|\overline{\boldsymbol{z}}_{mt}) = \mathcal{N}(\overline{y}_{mt}; \overline{\boldsymbol{z}}_{mt}; \psi)$. For now we treat ψ as known, but later (in Chapter 5.4) we describe how it and other model parameters can be learned from $\boldsymbol{Y}$.

Leveraging the zero-mean property of the noise, we first perform mean-removal on the observations $\boldsymbol{Y}$. In particular, we subtract the empirical mean

$$\mu \triangleq \frac{1}{MT} \sum_{t=1}^{T} \sum_{m=1}^{M} y_{mt} = \frac{1}{MT} \mathbf{1}_M^{\mathsf{T}} \boldsymbol{Y} \mathbf{1}_T \tag{6.3}$$

from $\boldsymbol{Y}$ to obtain

$$\underline{\boldsymbol{Y}} \triangleq \boldsymbol{Y} - \mu \mathbf{1}_M \mathbf{1}_T^{\mathsf{T}} \tag{6.4}$$

$$= \underbrace{\left(\boldsymbol{S} - \mu \mathbf{1}_M \mathbf{1}_N^{\mathsf{T}}\right)}_{\triangleq \underline{\boldsymbol{S}}} \boldsymbol{A} + \boldsymbol{W}, \tag{6.5}$$

where (6.5) employed (6.1) and $\mathbf{1}_N^{\mathsf{T}} \boldsymbol{A} = \mathbf{1}_T^{\mathsf{T}}$, the latter of which results from the simplex constraint on the columns of $\boldsymbol{A}$. It can then be shown (see Appendix C.1) that the elements of $\underline{\boldsymbol{S}}$ in (6.5) are approximately zero-mean.

To enforce the linear equality constraint $\mathbf{1}_N^{\mathsf{T}} \boldsymbol{A} = \mathbf{1}_T^{\mathsf{T}}$, we augment the observation model (6.5) into the form

$$\underbrace{\begin{bmatrix} \underline{\boldsymbol{Y}} \\ \mathbf{1}_T^{\mathsf{T}} \end{bmatrix}}_{\triangleq \bar{\boldsymbol{Y}}} = \underbrace{\begin{bmatrix} \underline{\boldsymbol{S}} \\ \mathbf{1}_N^{\mathsf{T}} \end{bmatrix}}_{\triangleq \bar{\boldsymbol{S}}} \boldsymbol{A} + \underbrace{\begin{bmatrix} \boldsymbol{W} \\ \mathbf{0}_T^{\mathsf{T}} \end{bmatrix}}_{\triangleq \bar{\boldsymbol{W}}}. \tag{6.6}$$

For the augmented model (6.6), the likelihood function of $\bar{\boldsymbol{Z}} \triangleq \bar{\boldsymbol{S}} \boldsymbol{A}$ takes the form in (2.6) with

$$p_{\overline{y}_{mt}|\overline{\boldsymbol{z}}_{mt}}(\overline{y}_{mt}|\overline{\boldsymbol{z}}_{mt}) = \underbrace{\begin{cases} \mathcal{N}(\overline{y}_{mt}; \overline{\boldsymbol{z}}_{mt}, \psi) & m = 1, \ldots, M \\ \delta(\overline{y}_{mt} - \overline{\boldsymbol{z}}_{mt}) & m = M+1. \end{cases}}_{\triangleq h_{mt}(\overline{\boldsymbol{z}}_{mt})} \tag{6.7}$$

We note that, ignoring spectral and spatial coherence, the model (6.6) is appropriate for the application of BiG-AMP, since the likelihood function $p_{\overline{\boldsymbol{Y}}|\overline{\boldsymbol{Z}}}(\bar{\boldsymbol{Y}}|\bar{\boldsymbol{Z}})$ is

known (up to ψ) and separable, and since the elements in $\bar{\boldsymbol{S}}$ and $\boldsymbol{A}$ can be treated as independent random variables with priors known up to a set of parameters, with those in $\bar{\boldsymbol{S}}$ being approximately zero-mean. In the sequel, we describe how the model (6.6) can be extended to capture spectral and spatial coherence.

6.2.2 Endmember Model

We desire a model that promotes spectral coherence in the endmembers, i.e., correlation among the (mean removed) spectral amplitudes $\{\underline{\mathsf{s}}_{mn}\}_{m=1}^{M}$ of each material n. However, since BiG-AMP needs $\underline{\mathsf{s}}_{mn}$ to be independent, we cannot impose correlation on these variables directly. Instead, we introduce an auxiliary sequence of correlated amplitudes $\{\mathsf{e}_{mn}\}_{m=1}^{M}$ such that $\underline{\mathsf{s}}_{mn}$ are independent *conditional on* e_{mn}. In particular,

$$p_{\underline{\boldsymbol{S}}|\boldsymbol{E}}(\underline{\boldsymbol{S}}|\boldsymbol{E}) = \prod_{m=1}^{M}\prod_{n=1}^{N} p_{\underline{\mathsf{s}}|\mathsf{e}}(\underline{s}_{mn}|e_{mn}) \tag{6.8}$$

$$p_{\underline{\mathsf{s}}|\mathsf{e}}(\underline{s}_{mn}|e_{mn}) = \underbrace{\delta(\underline{s}_{mn} - e_{mn})}_{\triangleq\, f_{mn}(s_{mn}, e_{mn})}, \tag{6.9}$$

implying that e_{mn} is merely a copy of $\underline{\mathsf{s}}_{mn}$. To impart correlation within the auxiliary sequences $\{\mathsf{e}_{mn}\}_{m=1}^{M}$, we model them as independent Gauss-Markov models

$$p_{\boldsymbol{E}}(\boldsymbol{E}) = \prod_{n=1}^{N} \underbrace{p(e_{1n}) \prod_{m=2}^{M} p(e_{mn}|e_{m-1,n})}_{\triangleq\, p_{\boldsymbol{e}_n}(\boldsymbol{e}_n)}, \tag{6.10}$$

where $\boldsymbol{e}_n \triangleq [\mathsf{e}_{1n}, \ldots, \mathsf{e}_{Mn}]^{\mathsf{T}}$, $\boldsymbol{e}_n \triangleq [e_{1n}, \ldots, e_{Mn}]^{\mathsf{T}}$, and

$$p(e_{1n}) = \mathcal{N}(e_{mn}; \kappa_n, \sigma_n^2) \tag{6.11}$$

$$p(e_{mn}|e_{m-1,n}) = \mathcal{N}\Big(e_{mn}; (1-\eta_n)e_{m-1,n} + \eta_n\kappa_n, \eta_n^2\sigma_n^2\Big). \tag{6.12}$$

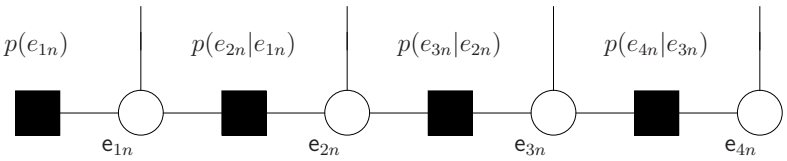


Figure 6.2: Factor graph for the stationary first-order Gauss-Markov chain used to model coherence in the spectrum of the n^{th} endmember, shown here for $M = 4$ spectral bands. Incoming messages from BiG-AMP flow downward into the e_{mn} nodes, and outgoing messages to BiG-AMP flow upward from the e_{mn} nodes.

In (6.11)-(6.12), $\kappa_n \in \mathbb{R}$ controls the mean of the nth process, σ_n^2 controls the variance, and $\eta_n \in [0, 1]$ controls the correlation. The resulting factor graph is illustrated in Fig. 6.2.

We note that the model (6.11)-(6.12) does not explicitly enforce non-negativity in s_{mn} because, for simplicity, we have omitted the constraint $\underline{s}_{mn} \geq -\mu$. Enforcement of $\underline{s}_{mn} \geq -\mu$ could be accomplished by replacing the pdfs in (6.11)-(6.12) with truncated Gaussian versions, but the computations required for inference would become much more tedious. In our experience, this tedium is not warranted: with practical HU datasets,[39] it suffices to enforce non-negativity in $\boldsymbol{A}$ and keep $\boldsymbol{Y} \approx \boldsymbol{S}\boldsymbol{A}$.

6.2.3 Abundance Model

We desire a model that promotes both sparsity and spatial coherence in the abundances a_{nt}. To accomplish the latter, we impose structure on the *support* of $\{\mathsf{a}_{nt}\}_{t=1}^{T}$ for each material n. For this purpose, we introduce the support variables $\mathsf{d}_{nt} \in \{-1, 1\}$, where $\mathsf{d}_{nt} = -1$ indicates that a_{nt} is zero-valued, and $\mathsf{d}_{nt} = 1$ indicates that a_{nt} is non-zero with probability 1, which we will refer to as "active." By modeling the abundances a_{nt} as independent *conditional on* d_{nt}, we comply with the

[39] Throughout our numerical experiments, the proposed inference method never produced a negative estimate of s_{mn}.

independence assumptions of BiG-AMP. In particular, we assume that

$$p_{\boldsymbol{A}|\boldsymbol{D}}(\boldsymbol{A}|\boldsymbol{D}) = \prod_{n=1}^{N}\prod_{t=1}^{T} p_{\mathsf{a}_n|\mathsf{d}_n}(a_{nt}|d_{nt}) \tag{6.13}$$

$$p_{\mathsf{a}_n|\mathsf{d}_n}(a_{nt}|d_{nt}) = \underbrace{\begin{cases}\delta(a_{nt}) & d_{nt}=-1\\ \zeta_n(a_{nt}) & d_{nt}=1\end{cases}}_{\triangleq\, g_{nt}(a_{nt},d_{nt})}, \tag{6.14}$$

where $\zeta_n(\cdot)$ denotes the pdf of a_{nt} when active. Essentially, we employ a Bernoulli-$\zeta_n(\cdot)$ distribution for the nth material.

We then place a Markov random field (MRF) prior on the support of the nth material, $\boldsymbol{d}_n \triangleq [\mathsf{d}_{n1},\ldots,\mathsf{d}_{nT}]^{\mathsf{T}}$:

$$p_{\boldsymbol{D}}(\boldsymbol{D}) = \prod_{n=1}^{N} p_{\boldsymbol{d}_n}(\boldsymbol{d}_n) \tag{6.15}$$

$$p_{\boldsymbol{d}_n}(\boldsymbol{d}_n) \propto \exp\left(\sum_{t=1}^{T}\left(\frac{1}{2}\sum_{i\in\mathcal{D}_t}\beta_n d_{ni} - \alpha_n\right)d_{nt}\right), \tag{6.16}$$

where $\mathcal{D}_t \subset \{1,\ldots,T\}\setminus t$ denotes the neighbors of pixel t. Roughly speaking, larger β_n yields higher spatial coherence and larger α_n yields higher sparsity. For simplicity, we adopt a neighborhood structure corresponding to the classical Ising model [106], as illustrated by the factor graph in Fig. 6.3.

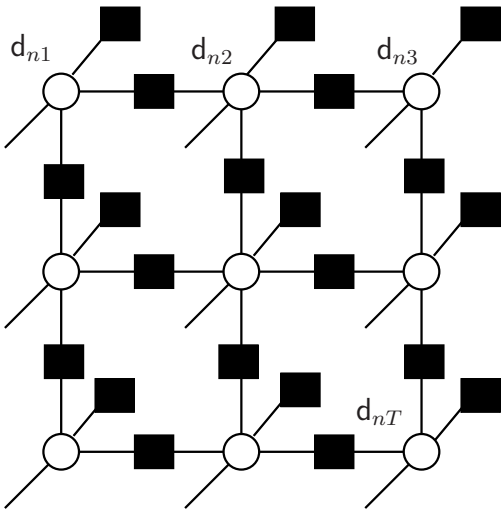


Figure 6.3: Factor graph for the Ising MRF used to model spatial coherence in the support of the nth abundance map, here for $T = 3 \times 3$ pixels. Incoming messages from BiG-AMP flow diagonally upward into the d_{nt} nodes, and outgoing messages to BiG-AMP flow diagonally downward from the d_{nt} nodes.

As for the active abundances, we adopt a non-negative Gaussian mixture (NNGM) distribution for $\zeta_n(\cdot)$:

$$\zeta_n(a) = \sum_{\ell=1}^{L} \omega_{n\ell}^{\mathsf{a}} \mathcal{N}_+(a; \theta_{n\ell}^{\mathsf{a}}, \phi_{n\ell}^{\mathsf{a}}), \tag{6.17}$$

where $\omega_{n\ell}^{\mathsf{a}} \geq 0$ and $\sum_{\ell=1}^{L} \omega_{n\ell}^{\mathsf{a}} = 1$. In (6.17), $\mathcal{N}_+$ refers to the truncated Gaussian pdf

$$\mathcal{N}_+(x; \theta, \phi) \triangleq \begin{cases} 0 & x < 0 \\ \dfrac{\mathcal{N}(x; \theta, \phi)}{\Phi_c(\theta/\sqrt{\phi})} & x \geq 0 \end{cases}, \tag{6.18}$$

where $\theta \in \mathbb{R}$ is a location parameter (but not the mean), $\phi > 0$ is a scale parameter (but not the variance), and $\Phi_c(\cdot)$ is the complimentary cdf of the $\mathcal{N}(0, 1)$ distribution. In practice, we find that $L = 3$ mixture components suffice, and we used this value throughout our numerical experiments in Chapter 6.4.

6.3 The HUT-AMP Algorithm

6.3.1 Message Passing and Turbo Inference

Our overall goal is to jointly estimate the (correlated, non-negative) endmembers $\boldsymbol{S}$ and (structured sparse, simplex-constrained) abundances $\boldsymbol{A}$ from noisy observations $\boldsymbol{Y}$ of the bilinear form $\boldsymbol{Z} = \boldsymbol{S}\boldsymbol{A}$. Using the mean-removed, augmented probabilistic models from Chapter 6.2, the joint pdf of all random variables can be factored as follows:

$$p(\bar{\boldsymbol{Y}}, \bar{\boldsymbol{S}}, \boldsymbol{A}, \boldsymbol{E}, \boldsymbol{D}) = p(\bar{\boldsymbol{Y}}|\bar{\boldsymbol{S}}, \boldsymbol{A})\, p(\bar{\boldsymbol{S}}, \boldsymbol{E})\, p(\boldsymbol{A}, \boldsymbol{D}) \tag{6.19}$$

$$= p_{\overline{\boldsymbol{Y}}|\overline{\boldsymbol{Z}}}(\bar{\boldsymbol{Y}}|\bar{\boldsymbol{S}}\boldsymbol{A})\, p_{\overline{\boldsymbol{S}}|\boldsymbol{E}}(\bar{\boldsymbol{S}}|\boldsymbol{E})\, p_{\boldsymbol{E}}(\boldsymbol{E})\, p_{\boldsymbol{A}|\boldsymbol{D}}(\boldsymbol{A}|\boldsymbol{D})\, p_{\boldsymbol{D}}(\boldsymbol{D}) \tag{6.20}$$

$$= \left(\prod_{m=1}^{M+1} \prod_{t=1}^{T} h_{mt} \left(\sum_{n=1}^{N} \overline{s}_{mn} a_{nt} \right) \right) \times \prod_{n=1}^{N} \left(\delta(\overline{s}_{M+1,n} - 1)\, p_{\boldsymbol{e}_n}(\boldsymbol{e}_n) \prod_{m=1}^{M} f_{mn}(\overline{s}_{mn}, e_{mn}) \times p_{\boldsymbol{d}_n}(\boldsymbol{d}_n) \prod_{t=1}^{T} g_{nt}(a_{nt}, d_{nt}) \right), \tag{6.21}$$

yielding the factor graph in Fig. 6.1. Due to the cycles within the factor graph, exact inference is NP-hard [113], and so we settle for approximate MMSE inference.

To accomplish approximate MMSE inference, we apply a form of loopy belief propagation that is inspired by the "turbo decoding" approach used in modern communications receivers [114]. In particular, after partitioning the overall factor graph into three subgraphs, as in Fig. 6.1, we alternate between message-passing *within* subgraphs and message-passing *between* subgraphs. In our case, BiG-AMP [29] is used for message-passing within the bilinear subgraph and standard methods from [106, 107] are used for message-passing within the other two subgraphs, which involve N Gauss-Markov chains and N binary MRFs, respectively. Overall, our proposed approach can

be interpreted as a bilinear extension of the "turbo AMP" approach first proposed in [32].

6.3.2 Messaging Between Subgraphs

For a detailed description of the message passing *within* the Gauss-Markov, MRF, and BiG-AMP subgraphs, we refer interested readers to [106], [107], and [29], respectively. We now describe the message passing *between* subgraphs, which relies on the sum-product algorithm (SPA) [44]. In our implementation of the SPA, we assume that all messages are scaled to form valid pdfs (in the case of continuous random variables) or pmfs (in the case of discrete random variables), and we use $\Delta_c^b(\cdot)$ to represent the message passed from node b to node c.

As described in [44], the SPA message flowing out of a variable node along a given edge equals the (scaled) product of messages flowing into that node along its other edges. Meanwhile, the SPA message flowing out of a factor node along a given edge equals the (scaled) integral of the product of all incoming messages times the factor associated with that node. Finally, the SPA approximates the posterior of a given random variable as the (scaled) product of messages flowing into that random variable.

As discussed in Chapter 2.1, a key property of BiG-AMP is that certain messages within its sub-graph are approximated as Gaussian. In particular,

$$\Delta_{f_{mn}}^{\mathsf{s}_{mn}}(\underline{s}) = \mathcal{N}(\underline{s}; \hat{q}_{mn}, \nu_{mn}^q) \tag{6.22}$$

$$\Delta_{g_{nt}}^{\mathsf{a}_{nt}}(a) = \mathcal{N}(a; \hat{r}_{nt}, \nu_{nt}^r), \tag{6.23}$$

where the quantities $\hat{q}_{mn}, \nu^q_{mn}, \hat{r}_{nt}, \nu^r_{nt}$ are computed during the final iteration of BiG-AMP. Thus, the SPA approximated posteriors on $\underline{\mathsf{s}}_{mn}$ and a_{nt} take the form

$$p_{\underline{\mathsf{s}}_{mn}|\mathsf{q}_{mn}}(\underline{s} \,|\, \hat{q}_{mn}; \nu^q_{mn}) \propto \Delta^{f_{mn}}_{\underline{\mathsf{s}}_{mn}}(\underline{s}) \mathcal{N}(\underline{s}; \hat{q}_{mn}, \nu^q_{mn}) \tag{6.24}$$

$$p_{\mathsf{a}_{nt}|\mathsf{r}_{nt}}(a \,|\, \hat{r}_{nt}; \nu^r_{nt}) \propto \Delta^{g_{nt}}_{\mathsf{a}_{nt}}(a) \mathcal{N}(a; \hat{r}_{nt}, \nu^r_{nt}). \tag{6.25}$$

We will use these properties in the sequel.

First, we discuss the message-passing between the bilinear sub-graph and spectral-coherence sub-graph in Fig. 6.1. Given (6.9), (6.22), and the construction of the factor graph in Fig. 6.1, the SPA implies that

$$\Delta^{f_{mn}}_{\mathsf{e}_{mn}}(e) \propto \int f_{mn}(\underline{s}, e) \, \Delta^{\underline{\mathsf{s}}_{mn}}_{f_{mn}}(\underline{s}) \, d\underline{s} = \mathcal{N}(e; \hat{q}_{mn}, \nu^q_{mn}). \tag{6.26}$$

The messages in (6.26) are used as inputs to the Gauss-Markov inference procedure. By construction, the outputs of the Gauss-Markov inference procedure will also be Gaussian beliefs. Denoting their means and variances by $\theta^{\underline{\mathsf{s}}}_{mn}$ and $\phi^{\underline{\mathsf{s}}}_{mn}$, respectively, we have that

$$\Delta^{\mathsf{e}_{mn}}_{f_{mn}}(e) \propto \mathcal{N}(e; \theta^{\underline{\mathsf{s}}}_{mn}, \phi^{\underline{\mathsf{s}}}_{mn}) \tag{6.27}$$

$$\Delta^{f_{mn}}_{\underline{\mathsf{s}}_{mn}}(\underline{s}) = \int f_{mn}(\underline{s}, e) \, \Delta^{\mathsf{e}_{mn}}_{f_{mn}}(e) \, de = \mathcal{N}(\underline{s}; \theta^{\underline{\mathsf{s}}}_{mn}, \phi^{\underline{\mathsf{s}}}_{mn}). \tag{6.28}$$

When BiG-AMP is subsequently called for inference on the bilinear sub-graph, it will use $\Delta^{f_{mn}}_{\underline{\mathsf{s}}_{mn}}(\cdot)$ as the prior on $\underline{\mathsf{s}}_{mn}$.

Next we discuss the message-passing between the bilinear sub-graph and the spatial-coherence sub-graph in Fig. 6.1. The SPA, together with the construction of the factor graph in Fig. 6.1, imply

$$\Delta^{g_{nt}}_{\mathsf{d}_{nt}}(d) = \frac{\int g_{nt}(a, d) \, \Delta^{\mathsf{a}_{nt}}_{g_{nt}}(a) \, da}{\sum_{d'=\pm 1} \int g_{nt}(a, d') \, \Delta^{\mathsf{a}_{nt}}_{g_{nt}}(a) \, da}, \; d \in \pm 1. \tag{6.29}$$

Given (6.14) and (6.23), we find that

$$\int g_{nt}(a,d)\,\Delta_{g_{nt}}^{\mathsf{a}_{nt}}(a)\,da = \begin{cases} \mathcal{N}(0;\hat{r}_{nt},\nu_{nt}^r)\,da & d=-1 \\ \int \zeta_n(a)\,\mathcal{N}(a;\hat{r}_{nt},\nu_{nt}^r)\,da & d=1 \end{cases} \tag{6.30}$$

which implies

$$\Delta_{\mathsf{d}_{nt}}^{g_{nt}}(d=+1) = \left(1+\frac{\mathcal{N}(0;\hat{r}_{nt},\nu_{nt}^r)}{\int \zeta_n(a)\,\mathcal{N}(a;\hat{r}_{nt},\nu_{nt}^r)}\right)^{-1} \tag{6.31a}$$

$$\Delta_{\mathsf{d}_{nt}}^{g_{nt}}(d=-1) = 1-\Delta_{\mathsf{d}_{nt}}^{g_{nt}}(d=+1), \tag{6.31b}$$

where the fraction in (6.31a) is BiG-AMP's approximation of the likelihood ratio $p_{\boldsymbol{Y}|\mathsf{d}_{nt}}(\boldsymbol{Y}|-1)/p_{\boldsymbol{Y}|\mathsf{d}_{nt}}(\boldsymbol{Y}|+1)$.

The Bernoulli beliefs from (6.31) are used as inputs to the MRF-based support-inference procedure. The outputs of the MRF inference procedure will also be Bernoulli beliefs of the form

$$\Delta_{g_{nt}}^{\mathsf{d}_{nt}}(d=+1) = \pi_{nt} \tag{6.32a}$$

$$\Delta_{g_{nt}}^{\mathsf{d}_{nt}}(d=-1) = 1-\pi_{nt} \tag{6.32b}$$

for some $\pi_{nt} \in (0,1)$. The SPA and (6.14) then imply that

$$\Delta_{\mathsf{a}_{nt}}^{g_{nt}}(a) \propto \sum_{d=\pm 1} g_{nt}(a,d)\,\Delta_{g_{nt}}^{\mathsf{d}_{nt}}(d) = (1-\pi_{nt})\delta(a)+\pi_{nt}\zeta_n(a) \tag{6.33}$$

for $\zeta_n(\cdot)$ defined in (6.17). When BiG-AMP is subsequently called for inference on the bilinear sub-graph, it will use $\Delta_{\mathsf{a}_{nt}}^{g_{nt}}(\cdot)$ as the prior on a_{nt}.

6.3.3 EM Learning of the Prior Parameters

In practice, we desire that the parameters

$$\boldsymbol{\Omega} = \left\{\psi, \{\omega_{n\ell}^{\mathsf{a}}, \theta_{n\ell}^{\mathsf{a}}, \phi_{n\ell}^{\mathsf{a}}\}_{\forall n\ell}, \{\eta_n, \kappa_n, \sigma_n^2, \alpha_n, \beta_n\}_{\forall n}\right\} \tag{6.34}$$

used for the assumed likelihood $p_{y_{mt}|\mathsf{z}_{mt}}(y_{mt}|\cdot)$, NNGM abundance prior $\zeta_n(\cdot)$, Gauss-Markov chain $p_{\boldsymbol{e}_n}(\cdot)$, and binary MRF $p_{\boldsymbol{d}_n}(\cdot)$ are well tuned. With this in mind, we propose an expectation-maximization (EM) [31] procedure to tune $\boldsymbol{\Omega}$, similar to that used for the GAMP-based sparse-reconstruction algorithms in [34] and [35].

To tune $\boldsymbol{\Omega}$, the EM algorithm [31] iterates

$$\boldsymbol{\Omega}^{i+1} = \arg\max_{\boldsymbol{\Omega}} \mathrm{E}\left\{ \ln p(\boldsymbol{E}, \boldsymbol{A}, \boldsymbol{D}, \overline{\boldsymbol{Y}}; \boldsymbol{\Omega}) \,\middle|\, \bar{\boldsymbol{Y}}; \boldsymbol{\Omega}^i \right\} \tag{6.35}$$

with the goal of increasing a lower bound on the true likelihood $p(\bar{\boldsymbol{Y}}; \boldsymbol{\Omega})$ at each EM-iteration i. In our case, the true posterior distribution used to evaluate the expectation in (6.35) is hard to compute, and so we use the SPA-approximated posteriors $\widehat{p_{\boldsymbol{E}|\overline{\boldsymbol{Y}}}}(\boldsymbol{E}|\bar{\boldsymbol{Y}}) \propto \prod_{m,n} \Delta^{f_{mn}}_{\mathsf{e}_{mn}}(e_{mn}) \Delta^{\mathsf{e}_{mn}}_{f_{mn}}(e_{mn})$ from (6.26)-(6.27), $\widehat{p_{\boldsymbol{D}|\overline{\boldsymbol{Y}}}}(\boldsymbol{D}|\bar{\boldsymbol{Y}}) \propto \prod_{n,t} \Delta^{g_{nt}}_{\mathsf{d}_{nt}}(d_{nt}) \Delta^{\mathsf{d}_{nt}}_{g_{nt}}(d_{nt})$ from (6.31)-(6.32), and $\widehat{p_{\boldsymbol{A}|\overline{\boldsymbol{Y}}}}(\boldsymbol{A}|\bar{\boldsymbol{Y}}) \propto \prod_{n,t} \Delta^{\mathsf{a}_{nt}}_{g_{nt}}(a_{nt}) \Delta^{g_{nt}}_{\mathsf{a}_{nt}}(a_{nt})$ from (6.23) and (6.33). Furthermore, since it is difficult to perform the maximization in (6.35) jointly, we maximize $\boldsymbol{\Omega}$ one component at a time (while holding the others fixed), which is the well known "incremental" variant of EM [61].

The resulting EM-update expressions for the parameters $\psi, \omega^{\mathsf{a}}_{n\ell}, \theta^{\mathsf{a}}_{n\ell}, \phi^{\mathsf{a}}_{n\ell}$ can be found in [35], and those for the Gauss-Markov chain parameters $\eta_n, \kappa_n, \sigma^2_n$ can be found in [112]. They are all computed in closed-form using readily available quantities, and thus do not add significantly to the complexity of HUT-AMP. The update procedure for the binary MRF parameters α_n, β_n is described in [111] and uses gradient descent. Since a small number of gradient-descent iterations suffice, this latter procedure does not significantly increase the complexity of HUT-AMP.

6.3.4 EM Initialization

Since the EM algorithm may converge to a local maximum of the likelihood, care must be taken when initializing the EM-learned parameters. Below, we propose an initialization strategy for HUT-AMP that, based on our empirical experience, seems to work well.

We first initialize the endmembers $\boldsymbol{S}$. For this, we found it effective to use an off-the-shelf EE algorithm like VCA [91] or FSNMF[40] [98] to recover $\hat{\boldsymbol{S}}^0$. Then, as described in (6.5), we subtract the observation mean μ from $\hat{\boldsymbol{S}}^0$ to obtain the initialization $\hat{\underline{\boldsymbol{S}}}^0$.

With the aid of $\hat{\underline{\boldsymbol{S}}}^0$, we next run BiG-AMP under

1. the trivial endmember prior

$$\Delta_{\underline{\mathsf{s}}_{mn}}^{f_{mn}}(\underline{s}) = \delta(\underline{s} - \hat{\underline{s}}_{mn}^0), \tag{6.36}$$

   which essentially fixes the endmembers at $\hat{\underline{\boldsymbol{S}}}^0$,

2. the agnostic NNGM abundance initialization from [35]:

$$\Delta_{\mathsf{a}_{nt}}^{g_{nt}}(a) = (1-\pi_{nt}^0)\delta(a) + \pi_{nt}^0 \sum_{\ell=1}^{L} \omega_{n\ell}^{\mathsf{a}} \mathcal{N}_+(a; \theta_{n\ell}^{\mathsf{a}}, \phi_{n\ell}^{\mathsf{a}}) \tag{6.37}$$

   with $\{\omega_{n\ell}^{\mathsf{a}}, \theta_{n\ell}^{\mathsf{a}}, \phi_{n\ell}^{\mathsf{a}}\}_{\ell=1}^{L}$ set at the best fit to a uniform distribution on the interval $[0,1]$ and $\pi_{nt}^0 = \frac{1}{N}$, and

3. the noise variance initialization from [35]:

$$\psi^0 = \frac{\|\underline{\boldsymbol{Y}}\|_F^2}{(\mathsf{SNR}^0 + 1)MT}, \tag{6.38}$$

[40] With FSNMF (which was used for all of the experiments in Chapter 6.4), we found that it helped to post-process the observations to reduce the effects of noise. For this, we used the standard PCA-based denoising approach described in [18]: the signal subspace was estimated from the left singular vectors of $\boldsymbol{Y}$ after row-wise mean-removal, and the FSNMF-estimated endmembers were projected onto the signal subspace.

where, without any prior knowledge of the true $\mathsf{SNR} \triangleq \mathrm{E}\{|z_{mt}|^2\}/\psi$, we suggest setting $\mathsf{SNR}^0 = 0$ dB.

By running BiG-AMP under these settings, we initialize the messages $\Delta_{f_{mn}}^{\mathsf{s}_{mn}}(\cdot)$ and $\Delta_{g_{nt}}^{\mathsf{a}_{nt}}(\cdot)$ from (6.22)-(6.23) and we also obtain an initial estimate of $\boldsymbol{A}$ from the mean of the approximate posterior (6.25), which we shall refer to as $\hat{\boldsymbol{A}}^0$.

Finally, we initialize the remaining parameters in Ω. Starting with the spectral coherence parameters, we set the mean κ_n^0 and variance $(\sigma_n^2)^0$ at the empirical mean and variance, respectively, of the elements in the nth column of $\hat{\underline{\boldsymbol{S}}}^0$. Then, we initialize the correlation η_n as suggested in [112], i.e.,

$$\varphi^0 = \frac{\|\underline{\boldsymbol{Y}}\|_F^2 - MT\psi^0}{\left\|\hat{\boldsymbol{A}}^0\right\|_F^2} \tag{6.39}$$

$$\eta_n^0 = 1 - \frac{1}{M-1}\sum_{m=1}^{M-1} \frac{|\underline{\boldsymbol{y}}_m^{\mathsf{T}}\underline{\boldsymbol{y}}_{m+1}|}{\varphi^0\left\|\hat{\boldsymbol{A}}^0\right\|_F^2} \text{ for } n = 1,\ldots,N, \tag{6.40}$$

where $\underline{\boldsymbol{y}}_m^{\mathsf{T}}$ denotes the mth row of $\underline{\boldsymbol{Y}}$. Lastly, we initialize the spatial coherence parameters as suggested in [111], i.e., $\beta_n^0 = 0.4$ and $\alpha_n^0 = 0.4$, since [111] shows these values to work well over a wide operating range.

6.3.5 HUT-AMP Summary

We now describe the scheduling of turbo-messaging and EM-tuning steps, which together constitute the HUT-AMP algorithm. Essentially, we elect to perform one EM update per turbo iteration, yielding the steps tabulated in Table 6.1. As previously mentioned, the "**BiGAMP**" operation iterates the BiG-AMP algorithm to convergence as described in [29], the "**GaussMarkov**" operation performs standard Gauss-Markov inference as described in [106], and the "**MRF**" operation performs MRF inference via the belief-propagation method described in [107].

Definitions:

$\boldsymbol{\Delta}_{\boldsymbol{F}}^{\boldsymbol{E}} \triangleq \{\Delta_{f_{mn}}^{\mathsf{e}_{mn}}(\cdot)\}_{\forall mn}$ $\quad \boldsymbol{\Delta}_{\boldsymbol{E}}^{\boldsymbol{F}} \triangleq \{\Delta_{\mathsf{e}_{mn}}^{f_{mn}}(\cdot)\}_{\forall mn}$

$\boldsymbol{\Delta}_{\underline{\boldsymbol{S}}}^{\boldsymbol{F}} \triangleq \{\Delta_{\underline{\mathsf{s}}_{mn}}^{f_{mn}}(\cdot)\}_{\forall mn}$ $\quad \boldsymbol{\Delta}_{\boldsymbol{F}}^{\underline{\boldsymbol{S}}} \triangleq \{\Delta_{f_{mn}}^{\underline{\mathsf{s}}_{mn}}(\cdot)\}_{\forall mn}$

$\boldsymbol{\Delta}_{\boldsymbol{G}}^{\boldsymbol{A}} \triangleq \{\Delta_{g_{nt}}^{\mathsf{a}_{nt}}(\cdot)\}_{\forall nt}$ $\quad \boldsymbol{\Delta}_{\boldsymbol{A}}^{\boldsymbol{G}} \triangleq \{\Delta_{\mathsf{a}_{nt}}^{g_{nt}}(\cdot)\}_{\forall nt}$

$\boldsymbol{\Delta}_{\boldsymbol{D}}^{\boldsymbol{G}} \triangleq \{\Delta_{\mathsf{d}_{nt}}^{g_{nt}}(\cdot)\}_{\forall nt}$ $\quad \boldsymbol{\Delta}_{\boldsymbol{G}}^{\boldsymbol{D}} \triangleq \{\Delta_{g_{nt}}^{\mathsf{d}_{nt}}(\cdot)\}_{\forall nt}$

Initialize $\boldsymbol{\Delta}_{\boldsymbol{F}}^{\underline{\boldsymbol{S}}}$, $\boldsymbol{\Delta}_{\boldsymbol{G}}^{\boldsymbol{A}}$, and $\boldsymbol{\Omega}^0$ as described in Chapter 6.3.4.
for $i = 1, 2, 3, \ldots$ **do**
  convert $\boldsymbol{\Delta}_{\boldsymbol{F}}^{\underline{\boldsymbol{S}}}$ to $\boldsymbol{\Delta}_{\boldsymbol{E}}^{\boldsymbol{F}}$ via (6.22) and (6.26)
  convert $\boldsymbol{\Delta}_{\boldsymbol{G}}^{\boldsymbol{A}}$ to $\boldsymbol{\Delta}_{\boldsymbol{D}}^{\boldsymbol{G}}$ via (6.23) and (6.31)
  $\boldsymbol{\Delta}_{\boldsymbol{F}}^{\boldsymbol{E}} = \mathsf{GaussMarkov}(\boldsymbol{\Delta}_{\boldsymbol{E}}^{\boldsymbol{F}}, \boldsymbol{\Omega}^i)$
  $\boldsymbol{\Delta}_{\boldsymbol{G}}^{\boldsymbol{D}} = \mathsf{MRF}(\boldsymbol{\Delta}_{\boldsymbol{D}}^{\boldsymbol{G}}, \boldsymbol{\Omega}^i)$
  convert $\boldsymbol{\Delta}_{\boldsymbol{F}}^{\boldsymbol{E}}$ to $\boldsymbol{\Delta}_{\underline{\boldsymbol{S}}}^{\boldsymbol{F}}$ via (6.27) and (6.28)
  convert $\boldsymbol{\Delta}_{\boldsymbol{G}}^{\boldsymbol{D}}$ to $\boldsymbol{\Delta}_{\boldsymbol{A}}^{\boldsymbol{G}}$ via (6.32) and (6.33)
  $\boldsymbol{\Omega}^i = \mathsf{EM}(\boldsymbol{\Delta}_{\boldsymbol{E}}^{\boldsymbol{F}}, \boldsymbol{\Delta}_{\boldsymbol{F}}^{\boldsymbol{E}}, \boldsymbol{\Delta}_{\boldsymbol{D}}^{\boldsymbol{G}}, \boldsymbol{\Delta}_{\boldsymbol{G}}^{\boldsymbol{D}}, \boldsymbol{\Delta}_{\boldsymbol{A}}^{\boldsymbol{G}}, \boldsymbol{\Delta}_{\boldsymbol{G}}^{\boldsymbol{A}}, \boldsymbol{\Omega}^{i-1})$
  $[\boldsymbol{\Delta}_{\boldsymbol{F}}^{\underline{\boldsymbol{S}}}, \boldsymbol{\Delta}_{\boldsymbol{G}}^{\boldsymbol{A}}] = \mathsf{BiGAMP}(\boldsymbol{\Delta}_{\underline{\boldsymbol{S}}}^{\boldsymbol{F}}, \boldsymbol{\Delta}_{\boldsymbol{A}}^{\boldsymbol{G}}, \boldsymbol{\Omega}^i)$
end for

Table 6.1: HUT-AMP pseudocode for fixed number of materials N.

6.3.6 Selection of Number of Materials

In practice, the number of materials N present in a scene may be unknown. Thus, we now propose a method to estimate N from the observed data $\boldsymbol{Y}$. For this, we use the standard form of penalized log-likelihood maximization [63]

$$\hat{N} = \arg\max_N 2 \ln p_{\overline{\boldsymbol{Y}}|\boldsymbol{Z}}(\bar{\boldsymbol{Y}}|\hat{\underline{\boldsymbol{S}}}_N \hat{\boldsymbol{A}}_N; \hat{\psi}_{\mathrm{ML}}) - \gamma(N), \tag{6.41}$$

where $\hat{\underline{\boldsymbol{S}}}_N$ and $\hat{\boldsymbol{A}}_N$ are the estimates of the mean-removed endmembers and abundances returned from N-material HUT-AMP, $\hat{\psi}_{\mathrm{ML}}$ is the ML estimate of the noise variance, and $\gamma(\cdot)$ is a penalty term. As recommended in [115], we choose $\gamma(\cdot)$ in accordance with the small-sample-corrected Akaike information criterion (AICc) [63], i.e., $\gamma(N) = 2\frac{MT}{MT-n(N)-1}n(N)$, where MT is the number of scalar observations in $\boldsymbol{Y}$ and $n(N)$ is the number of scalar degrees-of-freedom (DoF) in our model, which

depends on N. In particular, $n(N)$ comprises MN DoF from $\boldsymbol{S}$, $(N-1)T$ DoF from $\boldsymbol{A}$, and $5N + 2NL + N(L-1) + 1$ DoF from $\boldsymbol{\Omega}$. Plugging the standard form of the ML estimate of ψ (see, e.g., [63, eq. (7)]) into (6.41), we obtain

$$\hat{N} = \arg\max_{N} -MT \ln\left(\frac{\|\boldsymbol{Y} - \hat{\boldsymbol{S}}_N \hat{\boldsymbol{A}}_N\|_F^2}{MT}\right) - \frac{2MTn(N)}{MT - n(N) - 1}. \tag{6.42}$$

To solve the maximization in (6.42), we first run $N = 2$ HUT-AMP to completion and compute the penalized log-likelihood. We then increment N by 1, and compute the penalized log-likelihood again. If it increases, we increment N by 1 and repeat the procedure. Once the penalized log-likelihood decreases, we stop the procedure and select the previous model order N, which is the maximizer of the penalized log-likelihood. We refer to the resulting procedure as "HUT-AMP with model-order selection" (HUT-AMP-MOS).

We also note that a similar model-order selection strategy can be implemented to tune the number of NNGM components L used in (6.17), and we refer interested readers to [34] for more details. We note, however, that the fixed choice $L = 3$ was sufficient to yield the excellent results in Chapter 6.4.

6.4 Numerical Results

In this section, we report the results of several experiments that we conducted to characterize the performance of our proposed methods on both synthetic and real-world datasets.

In these experiments, we compared the endmembers $\hat{\boldsymbol{S}}$ recovered from our proposed HUT-AMP and HUT-AMP-MOS[41] unmixing algorithms to those recovered by

[41] Matlab code can be found at http://www.ece.osu.edu/~schniter/HUTAMP.

the Bayesian unmixing algorithm SCU [105] and the endmember extraction (EE) algorithms VCA [91], FSNMF [98], and MVSA [99]. For FSNMF, we used the PCA post-processing described in footnote 40 to reduce the effects of measurement noise, since this greatly improved its mean-squared estimation error.

We also compared the abundances $\hat{\boldsymbol{A}}$ recovered by our proposed HUT-AMP and HUT-AMP-MOS unmixing algorithms to those recovered by the Bayesian unmixing algorithm SCU, as well as those recovered by both FCLS (6.2) (implemented via Matlab's `lsqlin`) and SUnSAL-TV [103] using the endmember estimates produced by VCA, FSNMF, and MVSA.

In all cases, algorithms were run using their authors' implementation and suggested default settings, unless noted otherwise. For SUnSAL-TV, the regularization weights for the ℓ_1 and TV norms were hand-tuned, because cross-validation tuning was too computationally expensive given the sizes of the datasets.

6.4.1 Pixel Purity versus Abundance Sparsity

In the first experiment, we aim to assess EE performance as a function of pixel purity and abundance sparsity. Our motivation stems from the fact that the proposed HUT-AMP algorithm aims to exploit sparsity in the columns of the abundance matrix $\boldsymbol{A}$, while classical EE techniques like VCA and FSNMF aim to exploit the presence of pure pixels, recalling the discussion in Chapter 6.1. Thus, we are interested in seeing how these contrasting approaches fare under varying degrees of pixel purity and abundance sparsity.

For this first experiment, we constructed synthetic data consisting of $M = 100$ spectral bands, $T = 115$ spatial pixels, and $N = 10$ materials. The endmember matrix

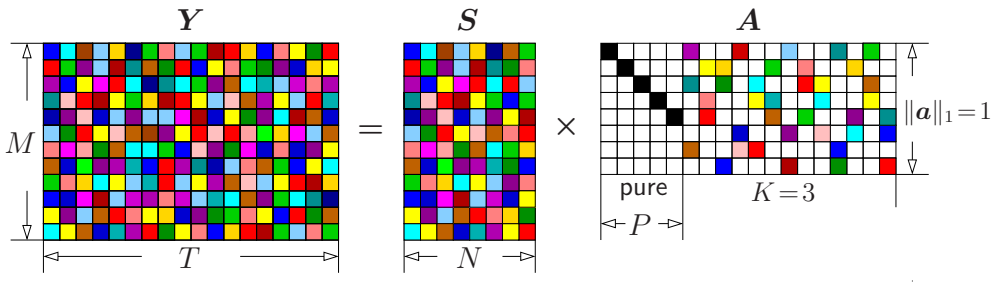


Figure 6.4: Illustration of the non-negative endmember matrix $\boldsymbol{S}$ and the K-sparse P-pure abundance matrix $\boldsymbol{A}$ for the first experiment.

$\boldsymbol{S} \in \mathbb{R}_+^{M\times N}$ was drawn i.i.d such that $\mathsf{s}_{mn} \sim \mathcal{N}_+(0.5, 0.05)$. The abundance matrix $\boldsymbol{A} \in \mathbb{R}_+^{N\times T}$ was generated as shown in Fig. 6.4, where P of the columns of $\boldsymbol{A}$ were assigned (uniformly at random) to be pure pixels, and the remaining columns were drawn K-sparse on the simplex. In particular, for each of these latter columns, the support was drawn uniformly at random, and the non-zero values $\{\underline{a}_k\}_{k=1}^K$ were drawn from a Dirichlet distribution, i.e.,

$$p(\underline{a}_1, \ldots, \underline{a}_{K-1}) = \begin{cases} \frac{\Gamma(\alpha K)}{\Gamma(\alpha)^K} \prod_{k=1}^K \underline{a}_k^{\alpha-1}, & \underline{a}_k \in [0,1] \\ 0 & \text{else} \end{cases} \tag{6.43a}$$

$$p(\underline{a}_K | \underline{a}_1, \ldots, \underline{a}_{K-1}) = \delta(1 - \underline{a}_1 - \cdots - \underline{a}_K), \tag{6.43b}$$

where $\Gamma(\cdot)$ denotes the gamma function, with concentration parameter $\alpha = 1$. Finally, the observation matrix $\boldsymbol{Y}$ was created by adding white Gaussian noise $\boldsymbol{W}$ to $\boldsymbol{Z} = \boldsymbol{S}\boldsymbol{A}$, where the noise variance ψ was adjusted to achieve $\mathsf{SNR} \triangleq \frac{1}{MT}\|\boldsymbol{Z}\|_F^2/\psi = 30$ dB.

We emphasize that neither spectral nor spatial coherence were used in this experiment. Thus, we turn off the spectral and spatial coherence exploitation in HUT-AMP, reducing our approach to EM-tuned BiG-AMP [29]. However, we emphasize that this application of EM-BiG-AMP differs from those in [115] in that it involves non-negativity and simplex constraints, i.e., it targets the NMF problem.

Figure 6.5 shows empirical success probability averaged over $R = 100$ realizations, as a function of pixel purity P and sparsity K, for the HUT-AMP, MVSA, VCA, and FSNMF algorithms. Here, a recovery was considered successful if $\mathsf{NMSE}_S \triangleq \|\boldsymbol{S} - \hat{\boldsymbol{S}}\|_F^2/\|\boldsymbol{S}\|_F^2 < -40$ dB. As seen in Fig. 6.5(c) and Fig. 6.5(d), VCA and FSNMF were only successful for the $K=1$ and $P=10$ cases, i.e., the pure-pixel cases. HUT-AMP, on the other hand, was able to successfully recover the endmembers for $K \leq 6$-sparse abundances, even when there was only $P = 1$ pure-pixels available. We attribute HUT-AMP's improved performance to its exploitation of sparsity rather than pure pixels (as with VCA and FSNMF). We also conjecture that sparsity (i.e., $K>1$ and $P<N$) is more important in practice, since the spatial resolution of the hyperspectral sensors may not guarantee pixel-purity for all materials, while sparse abundances (i.e., $K \ll N$) are more likely to hold. Meanwhile, we note that, although MVSA performs remarkably well for this problem, its performance suffers when real-world endmembers are considered, as demonstrated by the experiments in the sequel.

6.4.2 Pure-Pixel Synthetic Abundances

The second experiment uses synthetic pure-pixel abundances $\boldsymbol{A}$ with endmembers $\boldsymbol{S}$ chosen from the USGS Digital Spectral Library splib06a,[42] which contains laboratory-measured reflectance values for various materials over $M = 224$ spectral bands. To construct the data, we partitioned a scene of $T = 50 \times 50$ pixels into $N = 5$ equally sized vertical strips, each containing a single pure material. We then selected endmembers corresponding to the materials Grossular, Alunite, well crystallized (wxl) Kaolinite, Hydroxyl-Apatite, and Amphibole, noting that similar results were obtained in experiments we conducted with other materials. Finally, white

[42] See http://speclab.cr.usgs.gov/spectral.lib06/ds231/

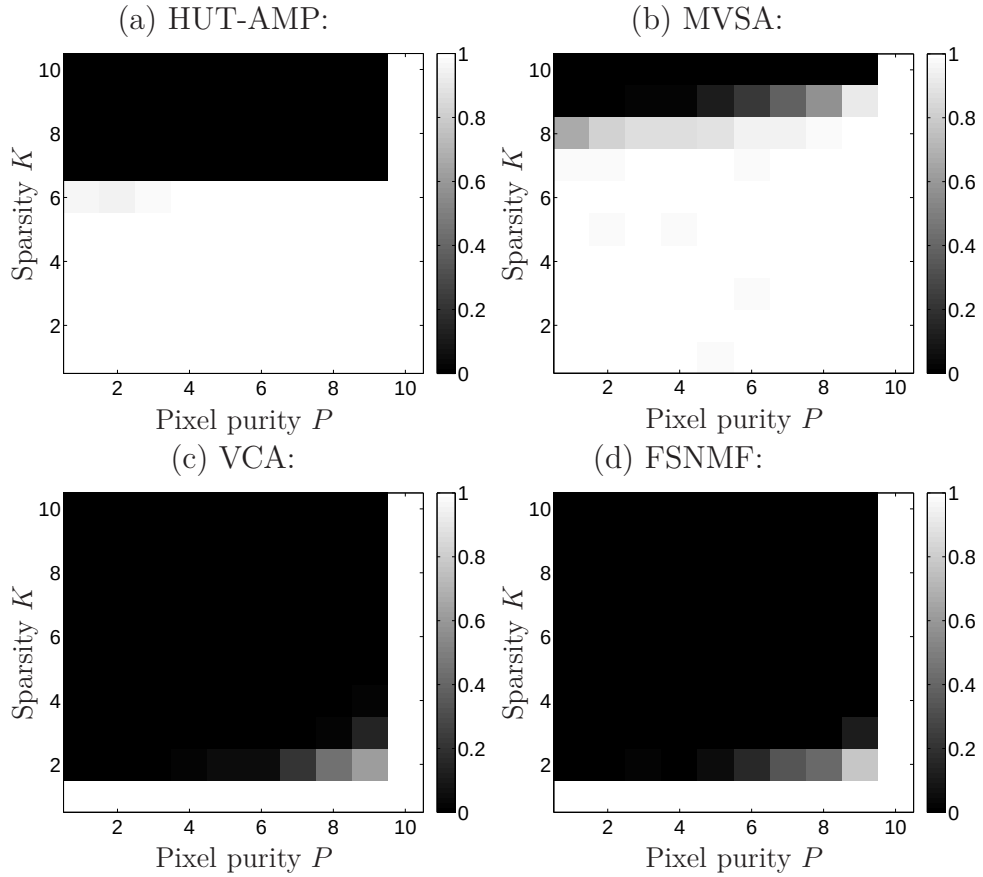


Figure 6.5: First experiment: Average success rate for near-perfect recovery of i.i.d endmembers $\boldsymbol{S}$ and K-sparse and P-pure abundances $\boldsymbol{A}$ using (a) HUT-AMP, (b) MVSA, (c) VCA, and (d) FSNMF.

Gaussian noise was added to achieve $\mathsf{SNR} = 30$ dB. Figure 6.6 shows an RGB image constructed from the noiseless measurements $\boldsymbol{Z}$.

Averaging over $R = 50$ noise realizations, Table 6.2 shows the normalized mean-squared error (i.e., $\mathsf{NMSE}_{\boldsymbol{S}}$ and $\mathsf{NMSE}_{\boldsymbol{A}} \triangleq \|\boldsymbol{A} - \hat{\boldsymbol{A}}\|_F^2 / \|\boldsymbol{A}\|_F^2$) of the estimated endmembers and abundances, the runtimes for the individual estimation of $\boldsymbol{S}$ and $\boldsymbol{A}$, and the total runtime. (For HUT-AMP and SCU, the estimation of $\boldsymbol{S}$ and $\boldsymbol{A}$ is done jointly, and thus only the total runtime is reported).

Figure 6.6: RGB image of the noiseless measurements $\boldsymbol{Z}$ used for the second experiment. Since the pixels are pure, each strip shows the RGB color of one of the $N=5$ materials. They are, in order from left to right: Grossular, Alunite, wxl Kaolinite, Hydroxyl-Apatite, and Amphibole.

For this pure-pixel dataset, Table 6.2 shows HUT-AMP dominating the other algorithms in both endmember and abundance estimation accuracy. In particular, HUT-AMP outperformed the best competing techniques by 9 dB in $\mathsf{NMSE}_{\boldsymbol{S}}$ and 60 dB in $\mathsf{NMSE}_{\boldsymbol{A}}$. We attribute HUT-AMP's excellent NMSE to several factors. First, it has the ability to *jointly* estimate endmembers and abundances, to exploit spectral coherence in the endmembers, and to exploit both spatial coherence and sparsity in the abundances (of which there is plenty in this experiment). Furthermore, due to the presence of pure-pixels throughout the scene, the "active" distribution $\zeta_n(\cdot)$ in (6.17) is simply a Bernoulli distribution, which HUT-AMP is able to learn (via EM) and exploit (via BiG-AMP) for improved performance. Although SCU also performs joint estimation and is able to exploit spectral and spatial coherence, we conjecture that its priors are less well-matched to this highly sparse dataset. Regarding the NMSE advantage of FCLS over SUnSAL-TV, we attribute this to the fact that FCLS enforces the sum-to-one abundance constraint whereas SUnSAL-TV does not.

	$\boldsymbol{S}$ time	$\boldsymbol{A}$ time	Total time	$\mathsf{NMSE}_{\boldsymbol{S}}$	$\mathsf{NMSE}_{\boldsymbol{A}}$
HUT-AMP	-	-	13.75 sec	**-52.0 dB**	**-91.8 dB**
SCU	-	-	3178 sec	-39.9 dB	-30.0 dB
VCA + FCLS	0.15 sec	6.15 sec	6.31 sec	-42.0 dB	-31.5 dB
VCA + SUnSAL-TV	0.15 sec	11.61 sec	11.76 sec	-42.2 dB	-27.6 dB
FSNMF + FCLS	0.06 sec	5.63sec	5.69 sec	-43.0 dB	-30.8 dB
FSNMF + SUnSAL-TV	0.05 sec	11.51 sec	11.56 sec	-43.0 dB	-29.7 dB
MVSA + FCLS	0.93 sec	3.66 sec	**4.59 sec**	-26.2 dB	-18.9 dB
MVSA + SUnSAL-TV	0.42 sec	7.73 sec	8.15 sec	-26.1 dB	-18.9 dB

Table 6.2: Runtime and endmember/abundance recovery NMSE and runtime for the second experiment.

Table 6.2 also shows that the total runtime of HUT-AMP is comparable to that of the "EE-and-inversion" techniques, while being 230 times faster than that of the Bayesian joint unmixing algorithm, SCU. In fact, SCU took advantage of parallel processing over 8 cores, whereas the other algorithms all used a single core. We conjecture that the slower runtime of SCU is due to its use of Gibbs sampling. As for the EE-and-inversion techniques, we note that their total runtime is dominated by the inversion step, which is 2-3 orders-of-magnitude more time-consuming than the EE step.

Although not shown in Table 6.2, we also ran HUT-AMP-MOS on this dataset. The result was that HUT-AMP-MOS correctly estimated the number of materials (i.e., $N = 5$) on every realization and thus gave identical $\mathsf{NMSE}_{\boldsymbol{S}}$ and $\mathsf{NMSE}_{\boldsymbol{A}}$ as HUT-AMP. The total runtime of HUT-AMP-MOS was 94.27 seconds, which was about 7 times slower than HUT-AMP but still 33 times faster than SCU.

6.4.3 SHARE 2012 Avon Dataset

Next, we evaluated algorithm performance on the real-world SHARE 2012 Avon dataset[43] [92], which uses $M = 360$ spectral bands, corresponding to wavelengths between 400 and 2450 nm, over a large rural area. To do this, we first cropped down the full image to the scene shown in Fig. 5.5, which is known to consist of $N=4$ materials: grass, dry sand, black felt, and white TyVek [93]. This scene was explicitly constructed for use in hyperspectral unmixing experiments, as efforts were made to ensure that the vast majority of the pixels were pure. Also, the data was collected on a nearly cloudless day, implying that shadowing effects were minimal.

To construct "ground truth" endmembers,[44] we averaged a 4×4 pixel grid of the received spectra in a "pure" region for each material. We then computed the spectral angle distance

$$\mathsf{SAD}_n = \arccos\left(\frac{\langle \boldsymbol{s}_n, \hat{\boldsymbol{s}}_n \rangle}{\|\boldsymbol{s}_n\|_2 \|\hat{\boldsymbol{s}}_n\|_2}\right) \tag{6.44}$$

between each ground-truth endmember $\boldsymbol{s}_n$ and the estimate $\hat{\boldsymbol{s}}_n$ produced by each algorithm. Table 6.3 shows median SAD over 50 trials, noting that VCA and SCU are random algorithms and thus exhibit variability across trials, even for this deterministic dataset. The table shows that HUT-AMP most accurately extracted all four endmembers. It also shows that SCU, VCA, and FSNMF all had a difficult time estimating the black felt endmember, most likely due to its low reflectivity. HUT-AMP, on the other hand, was able to leverage the spectral coherence of the black

[43] The SHARE 2012 Avon dataset can be obtained from http://www.rit.edu/cos/share2012/.

[44] In real-world HU data, ground truth endmembers are hard to come by, since lab-measured reflectivity can differ dramatically from received radiance at the sensor. In this experiment, we exploit the known purity of the pixels and we minimize noise effects through averaging.

	grass	dry sand	black felt	white TyVek
HUT-AMP	**1.46**	**0.79**	**3.14**	**0.45**
VCA	1.58	2.20	11.01	2.09
FSNMF	1.65	1.68	7.36	1.46
MVSA	4.57	10.42	45.47	1.60
SCU	2.69	2.48	32.10	1.19

Table 6.3: Median spectral angle distance (in degrees) between recovered and ground-truth endmembers in the SHARE 2012 experiment.

felt endmember, resulting in a 4 degree SAD improvement over the next best algorithm. Figure 6.7 shows an example of the extracted and ground-truth endmembers for visual comparison. It can be seen that HUT-AMP's estimates are a much better match to the ground-truth for all but the dry-sand material, where VCA performed very well *on this trial*; on most other trials, VCA performed worse. Figure 6.7 reveals that MVSA does not always yield non-negative endmembers, which may account for its relatively poor performance in all but our first experiment.

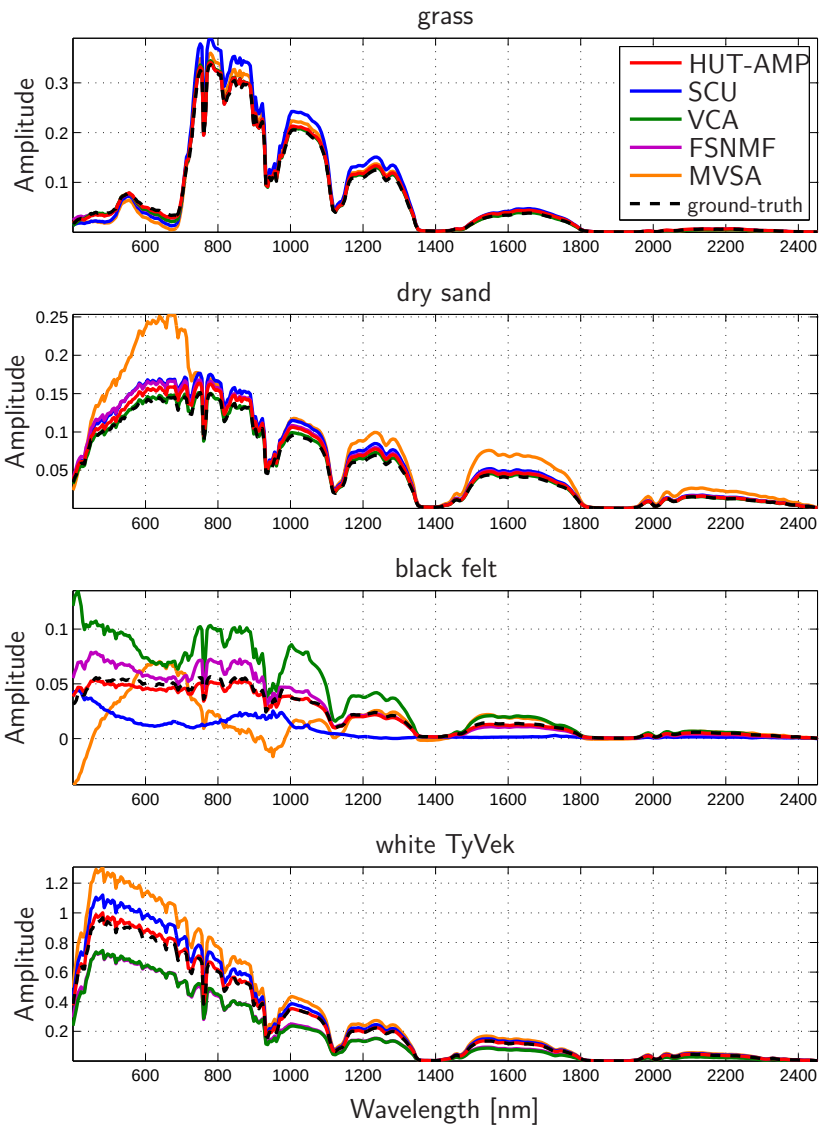


Figure 6.7: Examples of recovered and ground-truth endmembers for the SHARE 2012 experiment.

Since ground-truth abundance maps are not available for the SHARE 2012 dataset, we do not present quantitative results on the accuracy of abundance estimation. We do, however, plot the recovered abundance maps in Fig. 6.8. In interpreting Fig. 6.8, we reason that the "best" recoveries are the ones that are the most pure within the green, tan, black, and white regions of Fig. 5.5, given that great care was taken during data collection to keep each region occupied by a single material. Figure 6.8 shows that, in the case of dry sand and black felt, the abundances recovered by HUT-AMP were the most pure and, in the case of grass and Tyvek, the abundances recovered by HUT-AMP were among the most pure. The other Bayesian approach, SCU, yielded abundance estimates with much less purity, and we conjecture that was due to its priors being less well-matched to this highly sparse scene. Meanwhile, SUnSAL-TV (using both EE techniques) failed to recover the black felt material, which we attribute to its lack of a sum-to-one constraint.

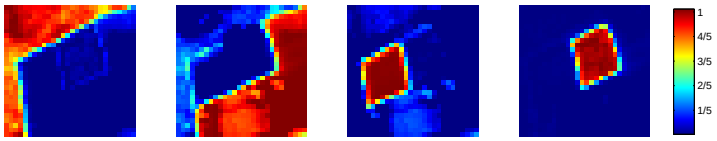
Figure 6.8 also reports the total runtime of each algorithm. There we see that HUT-AMP's runtime was 3.6 times slower than the average EE-and-inversion technique but 230 times faster than SCU, the other Bayesian technique. We also ran HUT-AMP-MOS on the SHARE 2012 data and found that it correctly estimated the presence of $N = 4$ materials, thus yielding identical recovery performance to HUT-AMP. HUT-AMP-MOS's runtime was 36.54 seconds, which was 3.5 times slower than HUT-AMP but still 67 times faster than SCU.

6.4.4 AVIRIS Cuprite Dataset

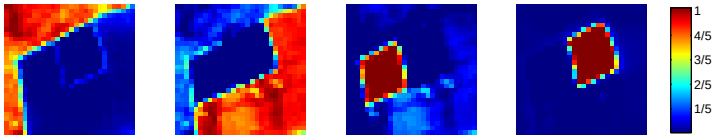
Our final numerical experiment was performed on the well known AVIRIS Cuprite dataset. Although the original dataset consisted of $M = 224$ spectral bands, ranging

from 0.4 to 2.5 μm, we aimed to replicate the setup in [105], which removed bands 1-10, 108-113, 153-168, and 223-224 to avoid water-absorption effects, resulting in $M = 189$ spectral measurements per pixel. And, like [105], we considered only the 80×80 pixel scene identified by the black square in Fig. 6.9. According to the tricorder classification map in Fig. 6.9, this scene contains 4 materials: Montmorillonite, Alunite, well crystallized (wxl) Kaolinite, and partially crystallized (pxl) Kaolinite. But, as in [18], we allow for the additional presence of Desert Varnish, bringing the total number of materials to $N = 5$. Also, like in [105], we consider both noiseless and white-Gaussian-noise corrupted measurements (at $\mathsf{SNR} = 30$ dB).

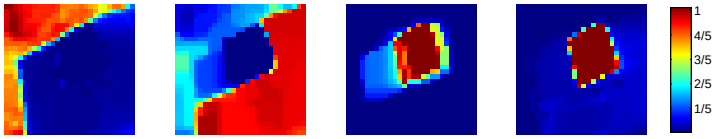
(a) HUT-AMP (average runtime = 10.52 sec):



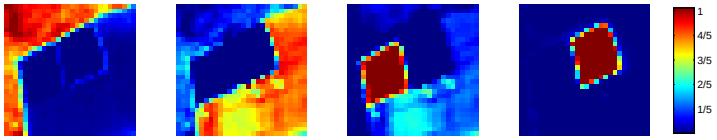
(b) VCA+FCLS (average runtime = 2.50 sec):



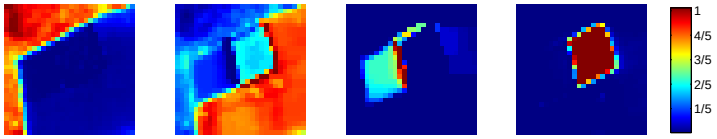
(c) VCA+SUnSAL-TV (average runtime = 4.13 sec):



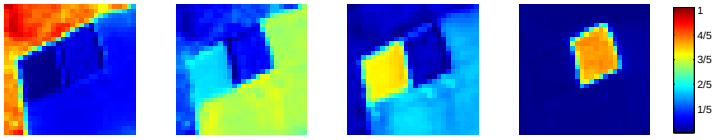
(d) FSNMF+FCLS (average runtime = 1.67 sec):



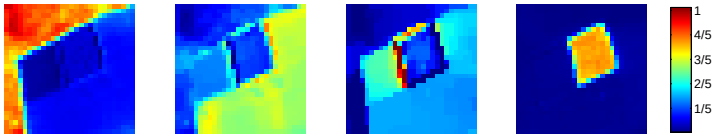
(e) FSNMF+SUnSAL-TV (average runtime = 3.36 sec):



(f) MVSA+FCLS (average runtime = 1.82 sec):



(g) MVSA+SUnSAL-TV (average runtime = 4.08 sec):



(h) SCU (average runtime = 2438 sec):

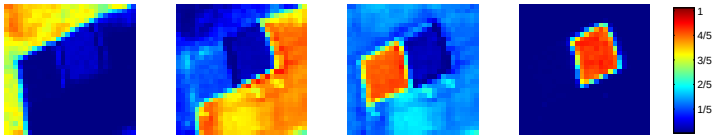


Figure 6.8: Examples of recovered abundance maps and average runtimes for the SHARE 2012 experiment. From left to right, the materials are: grass, dry sand, black felt, and white TyVek.

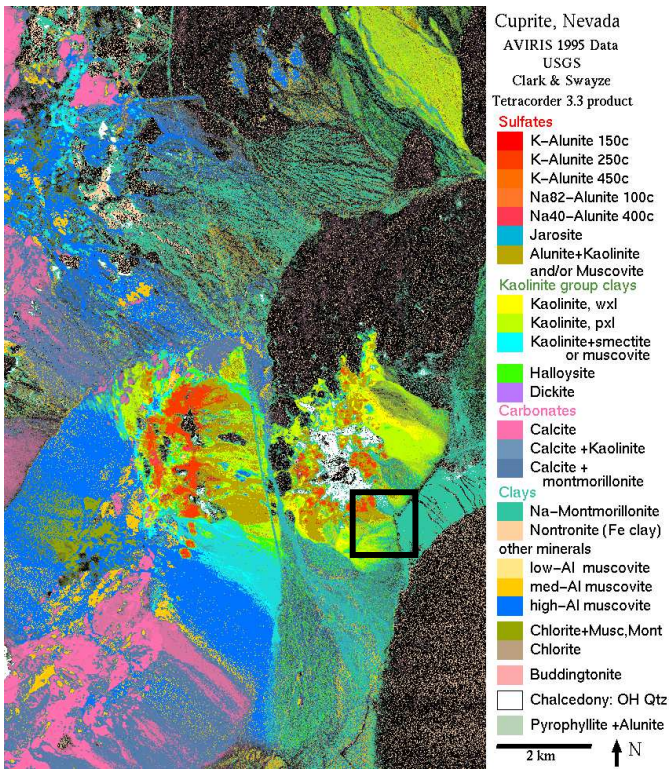


Figure 6.9: Mineral classification mapping of the Cuprite Dataset using the Tricorder 3.3 product [116]. We used the scene cropped by the black rectangle.

Table 6.4 shows the median SAD achieved during endmember extraction over 50 trials. (We used median for consistency with [105], and note that the performance of both VCA and SCU varied over trials due to the random nature of each algorithm.) Table 6.4 shows that, in the noiseless case, HUT-AMP achieved the best median SAD for three materials, while FSNMF achieved the best for two and SCU achieved the best for one. Meanwhile, in the noisy case, HUT-AMP achieved the best median SAD for two materials, while SCU, VCA, and FSNMF each achieved the best for one material. However, the SAD values in the table should be interpreted with caution, since i) they are based on the use of laboratory-measured reflectance spectra from the 2006 USGS library as ground-truth, whereas the Cuprite dataset itself uses reflectance units obtained via atmospheric correction of radiance data,[45] and ii) they are premised on the assumption that these particular materials are actually present in the scene.

[45]The reflectance and radiance versions of the Cuprite dataset can be found at http://aviris.jpl.nasa.gov/html/aviris.freedata.html

		SAD [degrees]				
		HUT-AMP	SCU	VCA	FSNMF	MVSA
SNR = ∞	Montmor.	**3.54**	3.95	3.91	**3.54**	6.03
	wxl Kaolinite	**9.68**	13.14	10.45	10.86	15.42
	pxl Kaolinite	**9.00**	11.45	9.22	9.38	10.51
	Desert Varnish	12.73	**10.83**	12.99	12.79	11.15
	Alunite	7.19	6.62	6.55	**6.40**	7.11
SNR = 30 dB	Montmor.	**3.52**	3.80	3.79	3.57	5.64
	wxl Kaolinite	12.73	12.46	**10.62**	12.93	15.59
	pxl Kaolinite	**9.08**	11.47	9.32	10.49	11.55
	Desert Varnish	12.90	**11.28**	12.97	12.77	11.70
	Alunite	7.19	7.94	6.60	**6.39**	7.16

Table 6.4: Median spectral angle distance (in degrees) for the Cuprite experiment.

Although a lack of ground truth prevents us from assessing abundance-map recovery performance for the Cuprite data, we plot the recovered (noiseless) abundance maps in Fig. 6.10 for visual comparison to the classification map in Fig. 6.9. Figure 6.10 shows that the abundance maps returned by HUT-AMP, FSNMF+FCLS, VCA+FCLS, FSNMF+SUnSAL-TV, and VCA+SUnSAL-TV have the highest contrast, suggesting that if certain pixels are truly pure then these algorithms are accurately estimating those pixels. The maps produced by SUnSAL-TV appear more "blurred," probably as an artifact of TV regularization. The abundances returned by SCU, MVSA+FCLS, and MVSA+SUnSAL-TV were of much lower contrast (i.e., much less sparse) and suggest different material placements than the maps generated by the other algorithms. For example, SCU suggests a significant wxl-Kaolin presence throughout the lower half of the scene, in contrast to other algorithms. However, Table 6.4 shows that SCU gave the worst SAD for wxl-Kaolin.

Figure 6.10 also shows the total runtimes of the various approaches. There we see that HUT-AMP is 5.6 times slower than the average VCA-or-FCLS-based approach, while SCU is 460 times slower. We also ran HUT-AMP-MOS on the Cuprite data and found that, in both the noiseless and noisy cases, it estimated the presence of $N = 5$ materials, and thus returned identical estimates to HUT-AMP. Meanwhile, HUT-AMP-MOS had an average runtime of 191.49 seconds, which is 30 times faster than SCU.

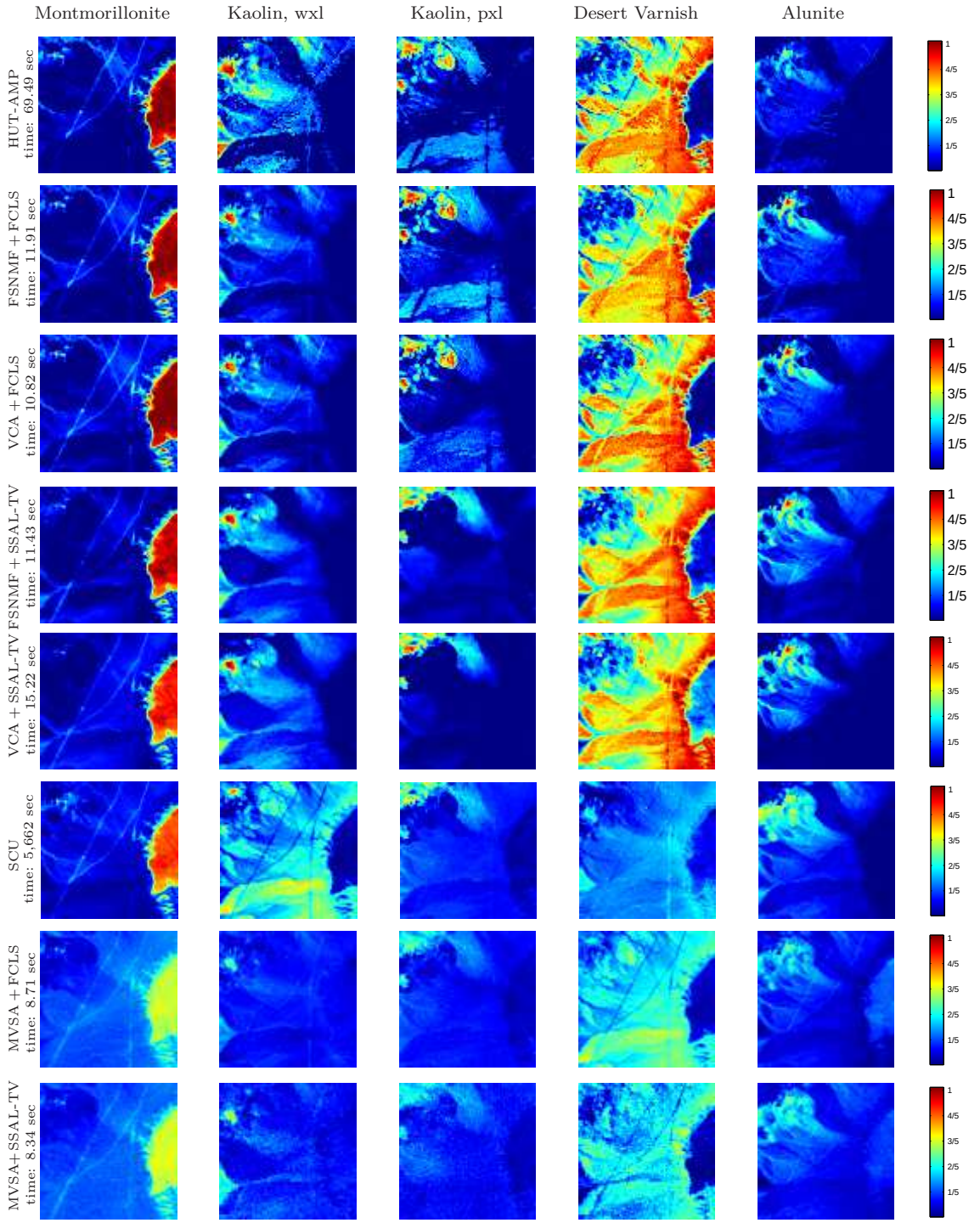


Figure 6.10: Examples of recovered abundance maps in the noiseless Cuprite experiment. Each row corresponds to an algorithm and each column corresponds to a material. Average runtimes are also listed on the left.

6.5 Conclusions

In this chapter, we proposed a novel Bayesian hyperspectral-unmixing algorithm that jointly estimates endmembers and abundance maps while exploiting the spectral and spatial coherence, as well as the abundance sparsity, that is often present in practice. To perform the overall inference task, we used the "turbo" approach suggested in [32], which breaks up the factor graph into three subgraphs, performs (approximate) BP individually on each subgraph, and then exchanges beliefs between subgraphs. For the spectral and spatial coherence subgraphs, we used standard Gauss-Markov and discrete-Markov methods [106, 107], respectively, while for the non-negative bilinear-mixing subgraph, we use the recently proposed BiG-AMP algorithm from [29], which exploits the approximate message passing framework from [16, 27]. Furthermore, we tune our prior and likelihood parameters using expectation-maximization, and we estimate the number of materials in the scene using penalized log-likelihood maximization.

Through a detailed numerical study, we demonstrated that our proposed HUT-AMP algorithm yields excellent recoveries on both synthetic and real-world datasets, and in many cases outperforms the other state-of-the-art algorithms under test. For example, our results suggest that HUT-AMP can reliably recover endmembers in the absence of pixel purity, unlike those endmember extraction algorithms designed around the pure-pixel assumption (e.g., VCA, FSNMF). Moreover, the runtime of HUT-AMP is on par with those of other EE-and-inversion techniques (e.g., VCA+FCLS), in contrast to other Bayesian spectral/spatial-coherence exploitation

techniques like SCU, whose runtime is 2-3 orders-of-magnitude larger. Our experiments also demonstrated that our model-order selection technique was able to correctly estimate the number of materials in several synthetic and real-world datasets, without requiring a very large increase in runtime.

Chapter 7: Conclusions

As the dimensionality of data continues to grow, it becomes increasingly important to develop inference algorithms that exploit known low-dimensional structures without inducing unnecessary computational burden.

In this dissertation, we proposed an empirical-Bayes methodology for the inference of structurally sparse signals, whose complexity scales linearly with problem dimensions M, N, and T. To do so, we iterate between using the AMP family of algorithms for signal inference and EM for automatic distributional parameter tuning, both of which thrive in the high-dimensional problem setting. When additional structure such as non-negativity, linear equality constraints, amplitude correlation, or structured sparsity persists, we can readily incorporate them into our Bayesian models and supplement them into AMP utilizing conditional independence. Furthermore, the developed framework can be easily adapted to additional types of structure, lending itself to a variety of additional applications.

We demonstrate numerous advantages of our approach using a variety of synthetic and real-world experiments on the compressive sensing and hyperspectral unmixing problems. In many experiments, the EM-AMP tuning methods yield equivalent recovery performance to "genie"-AMP methods, where the underlying probability distributions are perfectly known. This has the added benefit of relieving the user of the

task of "hand-tuning" our approach, in stark contrast of numerous other algorithms. Moreover, the developed algorithms exhibit favorable complexity scaling when compared to other commonly used approaches. Lastly, our empirical-Bayes approaches readily yield more accurate recoveries compared to other convex, greedy, and Bayesian methods, which we attribute to our ability to automatically tune a more flexible model while still promoting known low-dimensional structure.

Bibliography

[1] M. Wall, "Big data: Are you ready for blast-off?" http://www.bbc.com/news/business-26383058.htm, Mar. 2014.

[2] D. L. Donoho, "High-dimensional data analysis: The curses and blessings of dimensionality. aide-memoire of a lecture at," in *AMS Conference on Math Challenges of the 21st Century*, 2000.

[3] B. Efron, *Large-Scale Inference: Empirical Bayes Methods for Estimation, Testing, and Prediction*. New York: Cambridge University Press, 2010.

[4] V. Cevher, "Learning with compressible priors," in *Proc. Neural Inform. Process. Syst. Conf.*, Vancouver, B.C., Dec. 2009, pp. 261–269.

[5] S. K. Mitra, *Digital Signal Processing*, 2nd ed. New York: McGraw-Hill, 2001.

[6] E. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Trans. Inform. Theory*, vol. 52, no. 2, pp. 489–509, Feb. 2006.

[7] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inform. Theory*, vol. 52, no. 9, pp. 1289–1306, Sep. 2006.

[8] W. Lu and N. Vaswani, "Modified compressive sensing for real-time dynamic mr imaging," in *Image Processing (ICIP), 2009 16th IEEE International Conference on*, Nov 2009, pp. 3045–3048.

[9] L. C. Potter, E. Ertin, J. Parker, and M. Cetin, "Sparsity and compressed sensing in radar imaging," *Proc. IEEE*, vol. 98, no. 6, pp. 1006–1020, June 2010.

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

[11] M. Bayati, M. Lelarge, and A. Montanari, "Universality in polytope phase transitions and iterative algorithms," in *Proc. IEEE Int. Symp. Inform. Thy.*, Boston, MA, Jun. 2012, pp. 1643–1647, (full paper at *arXiv:1207.7321*).

[12] B. K. Natarajan, "Sparse approximate solutions to linear systems," *SIAM J. Comput.*, vol. 24, no. 2, pp. 227–234, Apr. 1995.

[13] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Roy. Statist. Soc. B*, vol. 58, no. 1, pp. 267–288, 1996.

[14] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM J. Scientific Comput.*, vol. 20, no. 1, pp. 33–61, 1998.

[15] D. L. Donoho, A. Maleki, and A. Montanari, "Message passing algorithms for compressed sensing: II. Analysis and validation," in *Proc. Inform. Theory Workshop*, Cairo, Egypt, Jan. 2010, pp. 1–5.

[16] ——, "Message passing algorithms for compressed sensing," *Proc. Nat. Acad. Sci.*, vol. 106, no. 45, pp. 18 914–18 919, Nov. 2009.

[17] Y. Wu and S. Verdú, "Optimal phase transitions in compressed sensing," *IEEE Trans. Inform. Theory*, vol. 58, no. 10, pp. 6241–6263, Oct. 2012.

[18] J. Bioucas-Dias, A. Plaza, N. Dobigeon, M. Parente, Q. Du, P. Gader, and J. Chanussot, "Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches," *IEEE J. Sel. Topics Appl. Earth Observ.*, vol. 5, no. 2, pp. 354–379, 2012.

[19] R. Rubinstein, A. Bruckstein, and M. Elad, "Dictionaries for sparse representation modeling," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 1045–1057, June 2010.

[20] E. J. Candès and Y. Plan, "Matrix completion with noise," *Proc. IEEE*, vol. 98, no. 6, pp. 925–936, Jun. 2010.

[21] D. L. Donoho and V. Stodden, "When does non-negative matrix factorization give a correct decomposition into parts?" in *Proc. Neural Inform. Process. Syst. Conf.*, 2003.

[22] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York: Springer, 2007.

[23] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear Phenomena*, vol. 60, no. 1-4, pp. 259–268, Nov. 1992.

[24] A. Chambolle, "An algorithm for total variation minimization and applications," *J. Math. Imaging Vis.*, vol. 20, pp. 80–97, 2004.

[25] S. Mallat, *A Wavelet Tour of Signal Processing: The Sparse Way*, 3rd ed. San Diego, CA: Academic Press, 2008.

[26] B. J. Frey and D. J. C. MacKay, "A revolution: Belief propagation in graphs with cycles," in *Proc. Neural Inform. Process. Syst. Conf.*, Denver, CO, 1997, pp. 479–485.

[27] D. L. Donoho, A. Maleki, and A. Montanari, "Message passing algorithms for compressed sensing: I. Motivation and construction," in *Proc. Inform. Theory Workshop*, Cairo, Egypt, Jan. 2010, pp. 1–5.

[28] S. Rangan, "Generalized approximate message passing for estimation with random linear mixing," in *Proc. IEEE Int. Symp. Inform. Thy.*, Aug. 2011, pp. 2168–2172, (full version at *arXiv:1010.5141*).

[29] J. T. Parker, P. Schniter, and V. Cevher, "Bilinear generalized approximate message passing—Part I: Derivation," *IEEE Trans. Signal Process.*, vol. 62, no. 22, pp. 5839–5853, Nov. 2014, (See also arXiv:1310:2632).

[30] M. Bayati and A. Montanari, "The dynamics of message passing on dense graphs, with applications to compressed sensing," *IEEE Trans. Inform. Theory*, vol. 57, no. 2, pp. 764–785, Feb. 2011.

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

[32] P. Schniter, "Turbo reconstruction of structured sparse signals," in *Proc. Conf. Inform. Science & Syst.*, Princeton, NJ, Mar. 2010, pp. 1–6.

[33] J. Vila and P. Schniter, "An empirical-Bayes approach to recovering linearly constrained non-negative sparse signals," in *Proc. IEEE Workshop Comp. Adv. Multi-Sensor Adaptive Process.*, Saint Martin, Dec. 2013, pp. 5–8, (full version at *arXiv:1310.2806*).

[34] J. P. Vila and P. Schniter, "Expectation-maximization Gaussian-mixture approximate message passing," *IEEE Trans. Signal Process.*, vol. 61, no. 19, pp. 4658–4672, Oct. 2013.

[35] ——, "An empirical-Bayes approach to recovering linearly constrained non-negative sparse signals," *IEEE Trans. Signal Process.*, vol. 62, no. 18, pp. 4689–4703, Sep. 2014, (see also *arXiv:1310.2806*).

[36] J. Vila, P. Schniter, and J. Meola, "Hyperspectral image unmixing via bilinear generalized approximate message passing," *Proc. SPIE*, vol. 8743, no. 87430Y, p. 9, 2013.

[37] J. Vila and P. Schniter, "Hyperspectral unmixing via turbo bilinear approximate message passing," *arXiv:1502.06435*, 2015.

[38] S. Rangan, P. Schniter, E. Riegler, A. Fletcher, and V. Cevher, "Fixed points of generalized approximate message passing with arbitrary matrices," in *Proc. IEEE Int. Symp. Inform. Thy.*, Jul. 2013, pp. 664–668, (full version at *arXiv:1301.6295*).

[39] S. Rangan, P. Schniter, and A. Fletcher, "On the convergence of generalized approximate message passing with arbitrary matrices," in *Proc. IEEE Int. Symp. Inform. Thy.*, Jul. 2014, pp. 236–240, (full version at *arXiv:1402.3210*).

[40] A. Javanmard and A. Montanari, "State evolution for general approximate message passing algorithms, with applications to spatial coupling," *Inform. Inference*, vol. 2, no. 2, pp. 115–144, 2013.

[41] A. Chambolle, R. A. DeVore, N. Lee, and B. J. Lucier, "Nonlinear wavelet image processing: Variational problems, compression, and noise removal through wavelet shrinkage," *IEEE Trans. Image Process.*, vol. 7, no. 3, pp. 319–335, Mar. 1998.

[42] I. Daubechies, M. Defrise, and C. D. Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint„" *Commun. Pure & Appl. Math.*, vol. 57, no. 11, pp. 1413–1457, Nov. 2004.

[43] A. Maleki and D. L. Donoho, "Optimally tuned iterative reconstruction algorithms for compressed sensing," *IEEE J. Sel. Topics Signal Process.*, vol. 4, no. 2, pp. 330–341, Apr. 2010.

[44] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. Inform. Theory*, vol. 47, pp. 498–519, Feb. 2001.

[45] F. Krzakala, A. Manoel, E. W. Tramel, and L. Zdeborová, "Variational free energies for compressed sensing," in *Proc. IEEE Int. Symp. Inform. Thy.*, Jul. 2014, pp. 1499–1503, (see also *arXiv:1402.1384*).

[46] F. Caltagirone, F. Krzakala, and L. Zdeborová, "On convergence of approximate message passing," in *Proc. IEEE Int. Symp. Inform. Thy.*, Jul. 2014, pp. 1812–1816, (see also *arXiv:1401.6384*).

[47] A. Manoel, F. Krzakala, E. W. Tramel, and L. Zdeborová, "Sparse estimation with the swept approximated message-passing algorithm," *arXiv:1406.4311*, Jun. 2014.

[48] S. Rangan, P. Schniter, J. T. Parker, J. Ziniel, J. Vila, M. Borgerding *et al.*, "GAMPmatlab," `https://sourceforge.net/projects/gampmatlab/`.

[49] T. Heskes, "Stable fixed points of loopy belief propagation are minima of the Bethe free energy," in *Proc. Neural Inform. Process. Syst. Conf.*, Vancouver, B.C., Dec. 2002, pp. 343–350.

[50] P. Schniter and S. Rangan, "Compressive phase retrieval via generalized approximate message passing," in *Proc. Allerton Conf. Commun. Control Comput.*, Monticello, IL, Oct. 2012, pp. 815–822.

[51] A. Manoel, F. Krzakala, E. W. Tramel, and L. Zdeborová, "SwAMP demo user's manual," `https://github.com/eric-tramel/SwAMP-Demo`.

[52] U. S. Kamilov, V. K. Goyal, and S. Rangan, "Message-passing de-quantization with applications to compressed sensing," *IEEE Trans. Signal Process.*, vol. 60, no. 12, pp. 6270–6281, Dec. 2012.

[53] D. L. Donoho and J. Tanner, "Observed universality of phase transitions in high-dimensional geometry, with implications for modern data analysis and signal processing," *Phil. Trans. Royal Soc. A*, vol. 367, no. 1906, pp. 4273–4293, 2009.

[54] D. L. Donoho, A. Maleki, and A. Montanari, "The noise-sensitivity phase transition in compressed sensing," *arXiv:1004.1218*, Apr. 2010.

[55] M. E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *J. Mach. Learn. Res.*, vol. 1, pp. 211–244, 2001.

[56] D. P. Wipf and B. D. Rao, "Sparse Bayesian learning for basis selection," *IEEE Trans. Signal Process.*, vol. 52, pp. 2153–2164, Aug. 2004.

[57] S. Ji, Y. Xue, and L. Carin, "Bayesian compressive sensing," *IEEE Trans. Signal Process.*, vol. 56, no. 6, pp. 2346–2356, Jun. 2008.

[58] U. S. Kamilov, S. Rangan, A. K. Fletcher, and M. Unser, "Approximate message passing with consistent parameter estimation and applications to sparse learning," in *Proc. Neural Inform. Process. Syst. Conf.*, Lake Tahoe, NV, Dec. 2012, pp. 2447–2455, (full version at *arXiv:1207.3859*).

[59] U. S. Kamilov, A. Bourquard, A. Amini, and M. Unser, "One-bit measurements with adaptive thresholds," *IEEE Signal Process. Lett.*, vol. 19, no. 10, pp. 607–610, Oct. 2012.

[60] C. F. J. Wu, "On the convergence properties of the EM algorithm," *Ann. Statist.*, vol. 11, no. 1, pp. 95–103, 1983.

[61] R. Neal and G. Hinton, "A view of the EM algorithm that justifies incremental, sparse, and other variants," in *Learning in Graphical Models*, M. I. Jordan, Ed. MIT Press, 1998, pp. 355–368.

[62] J. P. Vila and P. Schniter, "Expectation-maximization Bernoulli-Gaussian approximate message passing," in *Proc. Asilomar Conf. Signals Syst. Comput.*, Pacific Grove, CA, Nov. 2011, pp. 799–803.

[63] P. Stoica and Y. Selén, "Model-order selection: A review of information criterion rules," *IEEE Signal Process. Mag.*, vol. 21, no. 4, pp. 36–47, Jul. 2004.

[64] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM J. Imag. Sci.*, vol. 2, no. 1, pp. 183–202, 2009.

[65] H. Monajemi, S. Jafarpour, M. Gavish, Stat 330/CME 362 Collaboration, and D. L. Donoho, "Deterministic matrices matching the compressed sensing phase transitions of Gaussian random matrices," *Proc. Nat. Acad. Sci.*, vol. 110, no. 4, pp. 1181–1186, Jan. 2013.

[66] V. V. Buldygin and Y. V. Kozachenko, *Metric Characterization of Random Variables and Random Processes*. Providence, RI: Americal Mathematical Society, 2000.

[67] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *Proc. Asilomar Conf. Signals Syst. Comput.*, Pacific Grove, CA, 1993, pp. 40–44.

[68] W. Dai and O. Milenkovic, "Subspace pursuit for compressive sensing reconstruction," *IEEE Trans. Inform. Theory*, vol. 55, no. 5, pp. 2230–2249, Mar. 2009.

[69] Z. Zhang and B. D. Rao, "Sparse signal recovery with temporally correlated source vectors using sparse Bayesian learning," *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 5, pp. 912–926, Sep. 2011.

[70] E. van den Berg and M. P. Friedlander, "Probing the Pareto frontier for basis pursuit solutions," *SIAM J. Scientific Comput.*, vol. 31, no. 2, pp. 890–912, 2008.

[71] H. Mohimani, M. Babaie-Zadeh, and C. Jutten, "A fast approach for overcomplete sparse decomposition based on smoothed norm," *IEEE Trans. Signal Process.*, vol. 57, no. 1, pp. 289–301, Jan. 2009.

[72] Z. Zhang, "Master the usage of T-MSBL in 3 minutes," Univ. of California, San Diego, Tech. Rep., Nov. 2011.

[73] J. Ziniel, S. Rangan, and P. Schniter, "A generalized framework for learning and recovery of structured sparse signals," in *Proc. IEEE Workshop Statist. Signal Process.*, Ann Arbor, MI, Aug. 2012, pp. 325–328.

[74] "Compressive sensing resources: References and software," `http://dsp.rice.edu/cs`.

[75] H. M. Markowitz, *Portfolio Selection: Efficient Diversification of Investments*. New York: Wiley, 1991.

[76] J. Brodie, I. Daubechies, C. De Mol, D. Giannone, and I. Loris, "Sparse and stable Markowitz portfolios," *Proc. Nat. Acad. Sci.*, vol. 106, no. 30, pp. 12 267–12 272, 2009.

[77] B. M. Jedynak and S. Khudanpur, "Maximum likelihood set for estimating a probability mass function," *Neural Comput.*, vol. 17, no. 7, pp. 1508–1530, Jul. 2005.

[78] A. Kyrillidis, S. Becker, V. Cevher, and C. Koch, "Sparse projections onto the simplex," in *Proc. Int. Conf. Mach. Learning*, Jun. 2013, pp. 235–243, (full version at *arXiv:1206.1529*).

[79] D. L. Donoho and J. Tanner, "Sparse nonnegative solution of underdetermined linear equations by linear programming," *Proc. Nat. Acad. Sci.*, vol. 102, no. 27, pp. 9446–9451, 2005.

[80] A. M. Bruckstein, M. Elad, and M. Zibulevsky, "On the uniqueness of nonnegative sparse solutions to underdetermined systems of equations," *IEEE Trans. Inform. Theory*, vol. 54, no. 11, pp. 4813–4820, Nov. 2008.

[81] M. A. Khajehnejad, A. Dimakis, W. Xu, and B. Hassibi, "Sparse recovery of nonnegative signals with minimal expansion," *IEEE Trans. Signal Process.*, vol. 59, no. 1, pp. 198–208, Jan. 2011.

[82] J. Romberg, "Imaging via compressive sampling," *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 14–20, Mar. 2008.

[83] D. Chen and R. J. Plemmons, "Nonnegativity constraints in numerical analysis," in *The Birth of Numerical Analysis*, A. Bultheel and R. Cools, Eds. World Scientific Publishing Co., 2009, pp. 109–140.

[84] D. Bertsekas, *Nonlinear Programming*, 2nd ed. Athena Scientific, 1999.

[85] R. Giryes, M. Elad, and Y. C. Eldar, "The projectured GSURE for automatic parameter tuning in iterative shrinkage methods," *Appl. Computational Harmonic Anal.*, vol. 30, no. 2, pp. 407–422, 2011.

[86] M. Slawski and M. Hein, "Non-negative least squares for high-dimensional linear models: Consistency and sparse recovery without regularization," *Electron. J. Statist.*, vol. 7, pp. 3004–3056, Dec. 2013.

[87] R. Garg and R. Khandekar, "Gradient descent with sparsification: An iterative algorithm for sparse recovery with restricted isometry property," in *Proc. Int. Conf. Mach. Learning*, New York, 2009, pp. 337–344.

[88] S. Becker, E. Candès, and M. M. Grant, "Templates for convex cone problems with applications to sparse signal recovery," *Math. Program. Comput.*, vol. 3, no. 3, pp. 165–218, 2011.

[89] P. Bloomfield and W. L. Steiger, *Least Absolute Deviations: Theory, Applications, and Algorithms.* Boston: Birkhäuser, 1984.

[90] V. DeMiguel, L. Garlappi, and R. Uppal, "Optimal versus naive diversification: How inefficient is the 1-n portfolio strategy?" *Rev. of Financ. Stud.*, vol. 22, no. 5, pp. 1915–1953, May 2009.

[91] J. Nascimento and J. Bioucas Dias, "Vertex component analysis: a fast algorithm to unmix hyperspectral data," *IEEE Trans. Geosci. Remote Sens.*, vol. 43, no. 4, pp. 898–910, 2005.

[92] A. Giannandrea *et al.*, "The SHARE 2012 data collection campaign," *Proc. SPIE*, vol. 8743, no. 87430F, p. 15, 2013.

[93] K. Canham, D. Goldberg, J. Kerekes, N. Raqueno, and D. Messinger, "SHARE 2012: Large edge targets for hyperspectral imaging applications," *Proc. SPIE*, vol. 8743, no. 87430G, p. 9, 2013.

[94] D. Heinz and C.-I. Chang, "Fully constrained least squares linear spectral mixture analysis method for material quantification in hyperspectral imagery," *IEEE Trans. Geosci. Remote Sens.*, vol. 39, no. 3, pp. 529–545, 2001.

[95] P. E. Johnson, M. O. Smith, S. Taylor-George, and J. B. Adams, "A semiempirical method for analysis of the reflectance spectra of binary mineral mixtures," *J. Geophysical Research*, vol. 88, pp. 3557–3561, 1983.

[96] M. E. Winter, "N-FINDR: an algorithm for fast autonomous spectral endmember determination in hyperspectral data," *Proc. SPIE*, pp. 266–275, 1999.

[97] S. Arora, R. Ge, R. Kannan, and A. Moitra, "Computing a nonnegative matrix factorization – provably," in *Proc. Sym. Thy. Computing*, 2012, pp. 145–162.

[98] N. Gillis and S. A. Vavasis, "Fast and robust recursive algorithms for separable nonnegative matrix factorization," *arXiv:1208.1237v2*, 2012.

[99] J. Li and J. Bioucas-Dias, "Minimum volume simplex analysis: A fast algorithm to unmix hyperspectral data," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, vol. 3, Jul. 2008, pp. III – 250–III – 253.

[100] T.-H. Chan, C.-Y. Chi, Y.-M. Huang, and W.-K. Ma, "A convex analysis-based minimum-volume enclosing simplex algorithm for hyperspectral unmixing," *IEEE Trans. Signal Process.*, vol. 57, no. 11, pp. 4418–4432, Nov. 2009.

[101] R. Heylen, D. Burazerovic, and P. Scheunders, "Fully constrained least squares spectral unmixing by simplex projection," *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 11, 2011.

[102] M.-D. Iordache, J. Bioucas-Dias, and A. Plaza, "Sparse unmixing of hyperspectral data," *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 6, pp. 2014–2039, Jun. 2011.

[103] ——, "Total variation spatial regularization for sparse hyperspectral unmixing," *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 11, pp. 4484–4502, Nov. 2012.

[104] N. Dobigeon, S. Moussaoui, M. Coulon, J. Y. Tourneret, and A. Hero, "Joint bayesian endmember extraction and linear unmixing for hyperspectral imagery," *IEEE Trans. Signal Process.*, vol. 57, no. 11, pp. 4355–4368, 2009.

[105] R. Mittelman, N. Dobigeon, and A. Hero, "Hyperspectral image unmixing using a multiresolution sticky hdp," *IEEE Trans. Signal Process.*, vol. 60, no. 4, pp. 1656–1671, 2012.

[106] C. A. Bouman, "Markov random fields and stochastic image models," in *IEEE Int. Conf. Image Processing Tutorial*, Oct. 1995.

[107] S. Z. Li, *Markov Random Field Modeling in Image Analysis*, 3rd ed. London: Springer, 2009.

[108] P. Schniter, "A message-passing receiver for BICM-OFDM over unknown clustered-sparse channels," *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 8, pp. 1462–1474, Dec. 2011.

[109] M. Nassar, P. Schniter, and B. Evans, "A factor-graph approach to joint OFDM channel estimation and decoding in impulsive noise environments," *IEEE Trans. Signal Process.*, vol. 62, no. 6, pp. 1576–1589, Mar. 2014.

[110] S. Som and P. Schniter, "Compressive imaging using approximate message passing and a Markov-tree prior," *IEEE Trans. Signal Process.*, vol. 60, no. 7, pp. 3439–3448, Jul. 2012, (see also *arXiv:1108.2632*).

[111] ——, "Approximate message passing for recovery of sparse signals with Markov-random-field support structure," Jul. 2011, *Internat. Conf. Mach. Learning—Workshop on Structured Sparsity: Learning and Inference*, (Bellevue, WA).

[112] J. Ziniel and P. Schniter, "Dynamic compressive sensing of time-varying signals via approximate message passing," *IEEE Trans. Signal Process.*, vol. 61, no. 21, pp. 5270–5284, Nov. 2013, (see also *arXiv:1205.4080*).

[113] G. F. Cooper, "The computational complexity of probabilistic inference using Bayesian belief networks," *Artificial Intelligence*, vol. 42, pp. 393–405, 1990.

[114] R. J. McEliece, D. J. C. MacKay, and J.-F. Cheng, "Turbo decoding as an instance of Pearl's 'belief propagation' algorithm," *IEEE J. Sel. Areas Commun.*, vol. 16, no. 2, pp. 140–152, Feb. 1998.

[115] J. T. Parker, P. Schniter, and V. Cevher, "Bilinear generalized approximate message passing—Part II: Applications," *IEEE Trans. Signal Process.*, vol. 62, no. 22, pp. 5854–5867, Nov. 2014, (See also arXiv:1310:2632).

[116] R. N. Clark, G. A. Swayze, K. E. Livo, R. F. Kokaly, S. J. Sutley, J. B. Dalton, R. R. McDougal, and C. A. Gent, "Imaging spectroscopy: Earth and planetary remote sensing with the USGS tetracorder and expert systems," *J. Geophys. Res.*, vol. 108, no. E12, p. 5131, 2003.

[117] N. L. Johnson, S. Kotz, and N. Balakrishnan, *Continuous Univariate Distributions*. New York: Wiley & Sons, 1995, vol. 2.

Appendix A: EM-GM-AMP Derivations

A.1 EM Update of the Gaussian Noise Variance

From (4.20), the EM update for the noise variance solves

$$\psi^{i+1} = \arg\max_{\psi>0} \sum_{m=1}^{M} \int_{z_m} p_{\mathsf{z}|\mathsf{y}}(z_m|\boldsymbol{y};\boldsymbol{q}^i) \ln p_{\mathsf{y}|\mathsf{z}}(y_m|z_m;\psi) \tag{A.1}$$

since $\mathsf{z}_m = \boldsymbol{a}_m^{\mathsf{T}}\boldsymbol{x}$. The maximizing value of ψ in (A.1) is necessarily a value of ψ that zeroes the derivative of the sum, i.e., that satisfies[46]

$$\sum_{m=1}^{M} \int_{z_m} p_{\mathsf{z}|\mathsf{y}}(z_m|\boldsymbol{y};\boldsymbol{q}^i) \frac{d}{d\psi} \ln p_{\mathsf{y}|\mathsf{z}}(y_m|z_m;\psi) = 0. \tag{A.2}$$

Because $p_{\mathsf{y}|\mathsf{z}}(y_m|z_m;\psi) = \mathcal{N}(y_m; z_m, \psi)$, we can obtain

$$\frac{d}{d\psi} \ln p_{\mathsf{y}|\mathsf{z}}(y_m|z_m;\psi) = \frac{1}{2}\left(\frac{|y_m - z_m|^2}{\psi^2} - \frac{1}{\psi}\right), \tag{A.3}$$

which, when plugged into (A.2), yields the unique solution

$$\psi^{i+1} = \frac{1}{M} \sum_{m=1}^{M} \int_{z_m} p_{\mathsf{z}|\mathsf{y}}(z_m|\boldsymbol{y};\boldsymbol{q}^i) \, |y_m - z_m|^2. \tag{A.4}$$

We then use the moments $\hat{z}_m$ and μ_m^z from (R3)-(R4) in Table 2.1 to get the final update (4.21).

[46] The continuity of both the integrand and its partial derivative with respect to ψ allow the use of Leibniz's integral rule to exchange differentiation and integration.

A.2 EM Updates of the Signal Parameters: BG Case

A.2.1 EM Update of Sparsity Rate

The maximizing value of λ in (4.23) is necessarily a value of λ that zeroes the derivative of the sum, i.e., that satisfies[47]

$$\sum_{n=1}^{N} \int_{x_n} p_{\mathsf{x}|\boldsymbol{y}}(x_n|\boldsymbol{y};\boldsymbol{q}^i) \frac{d}{d\lambda} \ln p_{\mathsf{x}}(x_n;\lambda,\boldsymbol{q}^i_{\setminus\lambda}) = 0. \tag{A.5}$$

For the BG $p_X(x_n;\lambda,\theta,\phi)$ in (4.22), it is readily seen that

$$\frac{d}{d\lambda} \ln p_{\mathsf{x}}(x_n;\lambda,\boldsymbol{q}^i_{\setminus\lambda}) = \frac{\mathcal{N}(x_n;\theta^i,\phi^i) - \delta(x_n)}{p_{\mathsf{x}}(x_n;\lambda,\boldsymbol{q}^i_{\setminus\lambda})} = \begin{cases} \frac{1}{\lambda} & x_n \neq 0 \\ \frac{-1}{1-\lambda} & x_n = 0. \end{cases} \tag{A.6}$$

Plugging (A.6) and (4.7) into (A.5), it becomes evident that the neighborhood around the point $x_n = 0$ should be treated differently than the remainder of $\mathbb{R}$. Thus, we define the closed ball $\mathcal{B}_\epsilon \triangleq [-\epsilon,\epsilon]$ and its complement $\overline{\mathcal{B}_\epsilon} \triangleq \mathbb{R} \setminus \mathcal{B}_\epsilon$, and note that, in the limit $\epsilon \to 0$, the following is equivalent to (A.5):

$$\sum_{n=1}^{N} \underbrace{\int_{x_n \in \overline{\mathcal{B}_\epsilon}} p_{\mathsf{x}|\boldsymbol{y}}(x_n|\boldsymbol{y};\boldsymbol{q}^i)}_{\overset{\epsilon\to 0}{=} \pi_n} = \frac{\lambda}{1-\lambda} \sum_{n=1}^{N} \underbrace{\int_{x_n \in \mathcal{B}_\epsilon} p_{\mathsf{x}|\boldsymbol{y}}(x_n|\boldsymbol{y};\boldsymbol{q}^i)}_{\overset{\epsilon\to 0}{=} 1-\pi_n} \tag{A.7}$$

where the values taken by the integrals are evident from (4.8). Taking $\epsilon \to 0$ in (A.7) and applying straightforward algebra, we get the update

$$\lambda^{i+1} = \frac{1}{N} \sum_{n=1}^{N} \pi_n. \tag{A.8}$$

[47] To justify the exchange of differentiation and integration via Leibniz's integral rule here, one could employ the Dirac approximation $\delta(x) = \mathcal{N}(x;0,\varepsilon)$ for fixed arbitrarily small $\varepsilon > 0$, after which the integrand and its derivative w.r.t λ become continuous. The same comment applies in to all exchanges of differentiation and integration in the sequel.

A.2.2 EM Update of Active Mean

The maximizing value of θ in (4.25) is again a necessarily a value of θ that zeroes the derivative, i.e., that satisfies

$$\sum_{n=1}^{N} \int_{x_n} p_{\mathsf{x}|\boldsymbol{y}}(x_n|\boldsymbol{y};\boldsymbol{q}^i) \frac{d}{d\theta} \ln p_{\mathsf{x}}(x_n;\theta,\boldsymbol{q}^i_{\setminus\theta}) = 0. \tag{A.9}$$

For the BG $p_{\mathsf{x}}(x_n;\lambda,\theta,\phi)$ given in (4.22),

$$\frac{d}{d\theta} \ln p_{\mathsf{x}}(x_n;\lambda^i,\theta,\phi^i) = \frac{(x_n-\theta)}{\phi^i} \frac{\lambda^i \mathcal{N}(x_n;\theta,\phi^i)}{p_{\mathsf{x}}(x_n;\theta,\boldsymbol{q}^i_{\setminus\theta})} = \begin{cases} \frac{x_n-\theta}{\phi^i} & x_n \neq 0 \\ 0 & x_n = 0. \end{cases} \tag{A.10}$$

Splitting the domain of integration in (A.9) into $\mathcal{B}_\epsilon$ and $\overline{\mathcal{B}_\epsilon}$ as before, and then plugging in (A.10), we find that the following is equivalent to (A.9) in the limit of $\epsilon \to 0$:

$$\sum_{n=1}^{N} \int_{x_n \in \overline{\mathcal{B}_\epsilon}} (x_n - \theta)\, p_{\mathsf{x}|\boldsymbol{y}}(x_n|\boldsymbol{y};\boldsymbol{q}^i) = 0. \tag{A.11}$$

The unique value of θ satisfying (A.11) as $\epsilon \to 0$ is then

$$\theta^{i+1} = \frac{\sum_{n=1}^{N} \lim_{\epsilon\to 0} \int_{x_n \in \overline{\mathcal{B}_\epsilon}} x_n\, p_{\mathsf{x}|\boldsymbol{y}}(x_n|\boldsymbol{y};\boldsymbol{q}^i)}{\sum_{n=1}^{N} \lim_{\epsilon\to 0} \int_{x_n \in \overline{\mathcal{B}_\epsilon}} p_{\mathsf{x}|\boldsymbol{y}}(x_n|\boldsymbol{y};\boldsymbol{q}^i)} = \frac{1}{\lambda^{i+1} N} \sum_{n=1}^{N} \pi_n \gamma_{n,1} \tag{A.12}$$

where $\{\gamma_{n,1}\}_{n=1}^N$ are defined in (4.14). The last equality in (A.12) can be verified by plugging the GAMP posterior expression (4.8) into the second term in (A.12).

A.2.3 EM Update of Active Variance

The maximizing value of ϕ in (4.27) is again necessarily a value of ϕ that zeroes the derivative, i.e., that satisfies

$$\sum_{n=1}^{N} \int_{x_n} p_{\mathsf{x}|\boldsymbol{y}}(x_n|\boldsymbol{y};\boldsymbol{q}^i) \frac{d}{d\phi} \ln p_{\mathsf{x}}(x_n;\phi,\boldsymbol{q}^i_{\setminus\phi}) = 0. \tag{A.13}$$

For the $p_{\mathsf{x}}(x_n;\lambda,\theta,\phi)$ given in (4.22), it is readily seen that

$$\begin{aligned} \frac{d}{d\phi} \ln p_{\mathsf{x}}(x_n;\lambda^i,\theta^i,\phi) &= \frac{1}{2}\left(\frac{|x_n-\theta^i|^2}{(\phi)^2} - \frac{1}{\phi}\right) \frac{\lambda^i\, \mathcal{N}(x_n;\theta^i,\phi)}{p_{\mathsf{x}}(x_n;,\phi,\boldsymbol{q}^i_{\setminus\phi})} \\ &= \begin{cases} \frac{1}{2}\left(\frac{|x_n-\theta^i|^2}{(\phi)^2} - \frac{1}{\phi}\right) & x_n \neq 0 \\ 0 & x_n = 0 \end{cases}. \end{aligned} \tag{A.14}$$

Splitting the domain of integration in (A.13) into $\mathcal{B}_\epsilon$ and $\overline{\mathcal{B}_\epsilon}$ as before, and then plugging in (A.14), we find that the following is equivalent to (A.13) in the limit of $\epsilon \to 0$:

$$\sum_{n=1}^{N} \int_{x_n \in \overline{\mathcal{B}_\epsilon}} \left(|x_n - \theta^i|^2 - \phi \right) p_{\mathsf{x}|\boldsymbol{y}}(x_n|\boldsymbol{y}; \boldsymbol{q}^i) = 0. \tag{A.15}$$

The unique value of ϕ satisfying (A.15) as $\epsilon \to 0$ is then

$$\phi^{i+1} \quad = \quad \frac{\sum_{n=1}^{N} \lim_{\epsilon \to 0} \int_{x_n \in \overline{\mathcal{B}_\epsilon}} |x_n - \theta^i|^2 p_{\mathsf{x}|\boldsymbol{y}}(x_n|\boldsymbol{y}; \boldsymbol{q}^i)}{\sum_{n=1}^{N} \lim_{\epsilon \to 0} \int_{x_n \in \overline{\mathcal{B}_\epsilon}} p_{\mathsf{x}|\boldsymbol{y}}(x_n|\boldsymbol{y}; \boldsymbol{q}^i)}. \tag{A.16}$$

Finally, we expand $|x_n - \theta^i|^2 = |x_n|^2 - 2\,\mathrm{Re}(x_n^* \theta^i) + |\theta^i|^2$ which gives

$$\phi^{i+1} = \frac{1}{\lambda^{i+1} N} \sum_{n=1}^{N} \pi_n \left(\left| \theta^i - \gamma_{n,1} \right|^2 + \nu_{n,1} \right) \tag{A.17}$$

where $\{\nu_{n,1}\}_{n=1}^N$ from (4.15) are readily available BG-GAMP quantities. The equality in (A.17) can be readily verified by plugging (4.8) into (A.16).

A.3 EM Updates of the Signal Parameters: GM Case

A.3.1 EM Update of Active Means

Following (A.9), the maximizing value of θ_k in (4.30) is again necessarily a value of θ_k that zeros the derivative, i.e.,

$$\sum_{n=1}^{N} \int_{x_n} p_{\mathsf{x}|\boldsymbol{y}}(x_n|\boldsymbol{y}; \boldsymbol{q}^i) \frac{d}{d\theta_k} \ln p_{\mathsf{x}}(x_n; \theta_k, \boldsymbol{q}^i_{\setminus \theta_k}) = 0, \tag{A.18}$$

Plugging in the derivative

$$\frac{d}{d\theta_k} \ln p_{\mathsf{x}}(x_n; \theta_k, \boldsymbol{q}^i_{\setminus \theta_k}) = \left(\frac{x_n - \theta_k}{\phi_k^i} \right) \tag{A.19}$$
$$\times \frac{\lambda^i \omega_k^i \mathcal{N}(x_n; \theta_k, \phi_k^i)}{(1-\lambda^i)\delta(x_n) + \lambda^i (\omega_k^i \mathcal{N}(x_n; \theta_k, \phi_k^i) + \sum_{\ell \neq k} \omega_\ell^i \mathcal{N}(x_n; \theta_\ell^i, \phi_\ell^i))}$$

and the version of $p_{\mathsf{x}|\boldsymbol{y}}(x_n|\boldsymbol{y}; \boldsymbol{q}^i)$ from (4.7), integrating (A.18) separately over $\mathcal{B}_\epsilon$ and $\overline{\mathcal{B}_\epsilon}$ as in (A.7), and taking $\epsilon \to 0$, we find that the $\mathcal{B}_\epsilon$ portion vanishes, giving the

necessary condition

$$\sum_{n=1}^{N} \int_{x_n} \frac{p(x_n|\mathsf{x}_n \neq 0, \boldsymbol{y}; \boldsymbol{q}^i)\lambda^i \omega_k^i \mathcal{N}(x_n; \theta_k, \phi_k^i)(x_n - \theta_k)}{\zeta_n \Big(\omega_k^i \mathcal{N}(x_n; \theta_k, \phi_k^i) + \sum_{\ell \neq k} \omega_\ell^i \mathcal{N}(x_n; \theta_\ell^i, \phi_\ell^i)\Big)} = 0. \tag{A.20}$$

Since this integral cannot be evaluated in closed form, we apply the approximation $\mathcal{N}(x_n; \theta_k, \phi_k^i) \approx \mathcal{N}(x_n; \theta_k^i, \phi_k^i)$ in both the numerator and denominator, and subsequently exploit the fact that $p(x_n|\mathsf{x}_n \neq 0, \boldsymbol{y}; \boldsymbol{q}^i) = \mathcal{N}(x_n; \hat{r}_n, \mu_n^r) \sum_\ell \omega_\ell^i \mathcal{N}(x_n; \theta_\ell^i, \phi_\ell^i)$ from (4.7) to cancel terms, and so obtain the (approximated) necessary condition

$$\sum_{n=1}^{N} \int_{x_n} \frac{\lambda^i \omega_k^i \mathcal{N}(x_n; \hat{r}_n, \mu_n^r) \mathcal{N}(x_n; \theta_k^i, \phi_k^i)}{\zeta_n} (x_n - \theta_k) = 0. \tag{A.21}$$

We then simplify (A.21) using the Gaussian-pdf multiplication rule, and set θ_k^{i+1} equal to the value of θ_k that satisfies (A.21), which can be found to be

$$\theta_k^{i+1} = \frac{\sum_{n=1}^{N} \pi_n \overline{\beta}_{n,k} \gamma_{n,k}}{\sum_{n=1}^{N} \pi_n \overline{\beta}_{n,k}} \tag{A.22}$$

Note from (4.8) that $\pi_n \overline{\beta}_{n,k}$ can be interpreted as the probability that x_n originated from the k^{th} mixture component.

A.3.2 EM Update of Active Variances

Following (A.18), the maximizing value of ϕ_k in (4.31) is necessarily a value of ϕ_k that zeroes the derivative, i.e.,

$$\sum_{n=1}^{N} \int_{x_n} p_{\mathsf{x}|\boldsymbol{y}}(x_n|\boldsymbol{y}; \boldsymbol{q}^i) \frac{d}{d\phi_k} \ln p_{\mathsf{x}}(x_n; \phi_k, \boldsymbol{q}^i_{\setminus \phi_k}) = 0. \tag{A.23}$$

As for the derivative in the previous expression, we find

$$\frac{d}{d\phi_k} \ln p_{\mathsf{x}}(x_n; \phi_k, \boldsymbol{q}^i_{\setminus \phi_k}) = \frac{1}{2} \left(\frac{|x_n - \theta_k^i|^2}{\phi_k^2} - \frac{1}{\phi_k} \right) \tag{A.24}$$
$$\times \frac{\lambda^i \omega_k^i \mathcal{N}(x_n; \theta_k^i, \phi_k)}{(1 - \lambda^i)\delta(x_n) + \lambda^i (\omega_k^i \mathcal{N}(x_n; \theta_k^i, \phi_k) + \sum_{\ell \neq k} \omega_\ell^i \mathcal{N}(x_n; \theta_\ell^i, \phi_\ell^i))}.$$

Integrating (A.23) separately over $\mathcal{B}_\epsilon$ and $\overline{\mathcal{B}_\epsilon}$, as in (A.7), and taking $\epsilon \to 0$, we find that the $\mathcal{B}_\epsilon$ portion vanishes, giving

$$\sum_{n=1}^{N} \int_{x_n} \frac{p(x_n|\mathsf{x}_n \neq 0, \boldsymbol{y}; \boldsymbol{q}^i) \lambda^i \omega_k^i \mathcal{N}(x_n; \theta_k^i, \phi_k)/\zeta_n}{\omega_k^i \mathcal{N}(x_n; \theta_k^i, \phi_k) + \sum_{\ell \neq k} \omega_\ell^i \mathcal{N}(x_n; \theta_\ell^i, \phi_\ell^i)} \left(\frac{|x_n - \theta_k^i|^2}{\phi_k} - 1 \right) \tag{A.25}$$

Similar to (A.20), this integral is difficult to evaluate, and so we again apply the approximation $\mathcal{N}(x_n; \theta_k^i, \phi_k) \approx \mathcal{N}(x_n; \theta_k^i, \phi_k^i)$ in the numerator and denominator, after which several terms cancel, yielding the necessary condition

$$\sum_{n=1}^{N} \int_{x_n} \frac{\mathcal{N}(x_n; \hat{r}_n, \mu_n^r) \lambda^i \omega_k^i \mathcal{N}(x_n; \theta_k^i, \phi_k^i)}{\zeta_n} \left(\frac{|x_n - \theta_k^i|^2}{\phi_k} - 1 \right) = 0. \tag{A.26}$$

To find the value of ϕ_k satisfying (A.26), we expand $|x_n - \theta_k^i|^2 = |x_n|^2 - 2\,\mathrm{Re}(x_n^* \theta_k^i) + |\theta_k^i|^2$ and apply the Gaussian-pdf multiplication rule, which gives

$$\phi_k^{i+1} = \frac{\sum_{n=1}^{N} \pi_n \overline{\beta}_{n,k} \left(|\theta_k^i - \gamma_{n,k}|^2 + \nu_{n,k} \right)}{\sum_{n=1}^{N} \pi_n \overline{\beta}_{n,k}}. \tag{A.27}$$

A.3.3 EM Update of Active Weights

Finally, the value of the positive $\boldsymbol{\omega}$ maximizing (4.32) under the pmf constraint $\sum_{k=1}^{L} \omega_k = 1$ can be found by solving the augmented Lagrangian $\max_{\boldsymbol{\omega}, \xi} J(\boldsymbol{\omega}, \xi)$, where ξ is a Lagrange multiplier and

$$\begin{aligned} J(\boldsymbol{\omega}, \xi) &\triangleq \sum_{n=1}^{N} \hat{\mathrm{E}} \left\{ \ln p_{\mathsf{x}}(x_n; \boldsymbol{\omega}, \boldsymbol{q}^i_{\setminus \boldsymbol{\omega}}) \,\middle|\, \boldsymbol{y}; \boldsymbol{q}^i \right\} - \xi \left(\sum_{\ell=1}^{L} \omega_\ell - 1 \right) \\ &= \sum_{n=1}^{N} \int_{x_n} p_{\mathsf{x}|\boldsymbol{y}}(x_n|\boldsymbol{y}; \boldsymbol{q}^i) \ln p_{\mathsf{x}}(x_n; \boldsymbol{\omega}, \boldsymbol{q}^i_{\setminus \boldsymbol{\omega}}) - \xi \left(\sum_{\ell=1}^{L} \omega_\ell - 1 \right). \end{aligned} \tag{A.28}$$

We start by setting $\frac{d}{d\omega_k} J(\boldsymbol{\omega}, \xi) = 0$, which yields

$$\sum_{n=1}^{N} \int_{x_n} \frac{p_{\mathsf{x}}(x_n; \boldsymbol{q}^i) \mathcal{N}(x_n; \hat{r}_n, \mu_n^r)}{\zeta_n} \frac{d}{d\omega_k} \ln p_{\mathsf{x}}(x_n; \boldsymbol{\omega}, \boldsymbol{q}^i_{\setminus \boldsymbol{\omega}}) = \xi. \tag{A.29}$$

$$\Leftrightarrow \quad \sum_{n=1}^{N} \int_{x_n} \frac{p_{\mathsf{x}}(x_n; \boldsymbol{q}^i) \mathcal{N}(x_n; \hat{r}_n, \mu_n^r)}{\zeta_n} \frac{\lambda^i \mathcal{N}(x_n; \theta_k^i, \phi_k^i)}{p_{\mathsf{x}}(x_n; \boldsymbol{\omega}, \boldsymbol{q}^i_{\setminus \boldsymbol{\omega}})} = \xi. \tag{A.30}$$

Like in (A.20) and (A.25), the above integral is difficult to evaluate, and so we approximate $\boldsymbol{\omega} \approx \boldsymbol{\omega}^i$, which reduces the previous equation to

$$\xi = \sum_{n=1}^{N} \int_{x_n} \frac{\lambda^i \mathcal{N}(x_n; \theta_k^i, \phi_k^i) \mathcal{N}(x_n; \hat{r}_n, \mu_n^r)}{\zeta_n}. \tag{A.31}$$

Multiplying both sides by ω_k^i for $k = 1, \ldots, L$, summing over k, employing the fact $1 = \sum_k \omega_k^i$, and simplifying, we obtain the equivalent condition

$$\xi = \sum_{n=1}^{N} \int_{x_n} \frac{\lambda^i \sum_{k=1}^{L} \omega_k^i \mathcal{N}(x_n; \theta_k^i, \phi_k^i) \mathcal{N}(x_n; \hat{r}_n, \mu_n^r)}{\zeta_n} = \sum_{n=1}^{N} \pi_n. \tag{A.32}$$

Plugging (A.32) into (A.31) and multiplying both sides by ω_k, the derivative-zeroing value of ω_k is seen to be

$$\omega_k = \frac{\sum_{n=1}^{N} \int_{x_n} \lambda^i \omega_k \mathcal{N}(x_n; \theta_k^i, \phi_k^i) \mathcal{N}(x_n; \hat{r}_n, \mu_n^r) / \zeta_n}{\sum_{n=1}^{N} \pi_n}, \tag{A.33}$$

where, if we use $\omega_k \approx \omega_k^i$ on the right of (A.33), then we obtain

$$\omega_k^{i+1} = \frac{\sum_{n=1}^{N} \pi_n \overline{\beta}_{n,k}}{\sum_{n=1}^{N} \pi_n}. \tag{A.34}$$

Appendix B: EM-NN-AMP Derivations

B.1 EM Update for AWGN Variance

Inserting the Gaussian likelihood (5.10) into (5.44), we see that the EM update for the noise variance ψ becomes

$$\psi^{i+1} = \arg\max_{\psi} \frac{M}{2} \ln \frac{1}{\psi} - \frac{1}{2\psi} \hat{\mathrm{E}}\{\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \,\big|\, \boldsymbol{y}; \psi^i\}, \tag{B.1}$$

where, for the joint posterior $p_{\boldsymbol{x}|\boldsymbol{y}}(\boldsymbol{x}|\boldsymbol{y}; \psi^i)$, we use the product of the approximate marginal GAMP posteriors from (2.5). By zeroing the derivative of the objective in (B.1) w.r.t. ψ, we find that

$$\psi^{i+1} = \frac{1}{M} \hat{\mathrm{E}}\{\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \,\big|\, \boldsymbol{y}; \psi^i\}, \tag{B.2}$$

where the expectation simplifies to

$$\begin{aligned}
&\hat{\mathrm{E}}\{\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \,\big|\, \boldsymbol{y}; \psi^i\} \\
&\quad = \boldsymbol{y}^{\mathsf{T}}\boldsymbol{y} - \boldsymbol{y}^{\mathsf{T}}\boldsymbol{A}\hat{\boldsymbol{x}} + \hat{\mathrm{E}}\{\boldsymbol{x}^{\mathsf{T}}\boldsymbol{A}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{x} \,\big|\, \boldsymbol{y}; \psi^i\} && \text{(B.3)} \\
&\quad = \boldsymbol{y}^{\mathsf{T}}\boldsymbol{y} - \boldsymbol{y}^{\mathsf{T}}\boldsymbol{A}\hat{\boldsymbol{x}} + \mathrm{tr}(\boldsymbol{A}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{\Sigma}) + \hat{\boldsymbol{x}}^{\mathsf{T}}\boldsymbol{A}^{\mathsf{T}}\boldsymbol{A}\hat{\boldsymbol{x}} && \text{(B.4)} \\
&\quad = \|\boldsymbol{y} - \boldsymbol{A}\hat{\boldsymbol{x}}\|_2^2 + \mathrm{tr}(\boldsymbol{A}^{\mathsf{T}}\boldsymbol{A}\boldsymbol{\Sigma}). && \text{(B.5)}
\end{aligned}$$

Here, $\boldsymbol{\Sigma}$ is the posterior covariance matrix of $\boldsymbol{x}$, which—based on our assumptions—is diagonal with $[\boldsymbol{\Sigma}]_{nn} = \mu_n^x$. Plugging in (B.5) into (B.2), we obtain the EM update (5.45).

B.2 Derivation of Laplacian Likelihood Quantities

B.2.1 Laplacian Likelihood Steps for Sum-product GAMP

Inserting the Laplacian likelihood (5.13) into the GAMP-approximated posterior (2.4), the posterior mean in line (R5) of Table 2.1 becomes (removing the m subscript for brevity)

$$\hat{z} \triangleq \mathrm{E}\{\mathsf{z} \,|\, \mathsf{p}=\hat{p}; \mu^p\} = \frac{1}{C}\int_z z\, \mathcal{L}(z; y; \psi)\mathcal{N}(z; \hat{p}, \mu^p) \tag{B.6}$$

where the scaling constant C is calculated as

$$\begin{aligned} C &= \int_z \mathcal{L}(z; y, \psi)\mathcal{N}(z; \hat{p}, \mu^p) && \text{(B.7)} \\ &= \int_{z'} \mathcal{L}(z'; 0, \psi)\mathcal{N}(z'; \hat{p}-y, \mu^p) && \text{(B.8)} \\ &= \underbrace{\frac{\psi}{2}\int_{-\infty}^{0} \mathcal{N}(z; \tilde{p}, \mu^p)e^{\psi z}dz}_{\triangleq \underline{C}} + \underbrace{\frac{\psi}{2}\int_{0}^{\infty} \mathcal{N}(z; \tilde{p}, \mu^p)e^{-\psi z}dz}_{\triangleq \overline{C}} && \text{(B.9)} \end{aligned}$$

where $\tilde{p} \triangleq \hat{p} - y$. The expressions for $\underline{C}$ and $\overline{C}$ reported in (5.19)-(5.20) result after completing the square inside the exponential terms in the integrands in (B.9) and simplifying.

Following similar techniques (i.e., shifting z by y and splitting the integral), it can be shown that (B.6) becomes

$$\hat{z} = y + \frac{\underline{C}}{C}\int_z z\mathcal{N}_-(z; \tilde{p}, \mu^p) + \frac{\overline{C}}{C}\int_z z\mathcal{N}_+(z; \tilde{p}, \mu^p), \tag{B.10}$$

where $\mathcal{N}_+(\cdot)$ is defined in (5.32) and where $\mathcal{N}_-(x; a, b^2)$ is the pdf that results from taking a Gaussian with mean a and variance b^2, truncating its support to $x \in (-\infty, 0]$, and normalizing. Supposing that $\mathsf{u} \sim \mathcal{N}(a, b^2)$, [117] shows that

$$\mathrm{E}\{\mathsf{u} \,|\, \mathsf{u} > 0\} = \int_u u\mathcal{N}_+(u; a, b^2) = a + bh(-\tfrac{a}{b}), \tag{B.11}$$

$$\mathrm{E}\{\mathsf{u} \,|\, \mathsf{u} < 0\} = \int_u u\mathcal{N}_-(u; a, b^2) = a - bh(\tfrac{a}{b}), \tag{B.12}$$

where $h(\cdot)$ is defined in (5.21). Inserting (B.11) and (B.12) into (B.10) yields the posterior mean expression in (5.17).

To calculate the posterior variance μ^z used in line (R6) of Table 2.1, we begin with

$$\mathrm{E}\{\mathsf{z}^2 \,|\, \mathsf{p}{=}\hat{p}; \mu^p\} = \frac{1}{C}\int_z z^2 \mathcal{L}(z; y; \psi)\mathcal{N}(z; \hat{p}, \mu^p) \tag{B.13}$$

$$= \frac{1}{C}\int_{z'} (z'+y)^2 \mathcal{L}(z'; 0; \psi)\mathcal{N}(z'; \tilde{p}, \mu^p) \tag{B.14}$$

$$= 2y(\hat{z}-y) + y^2 + \frac{1}{C}\int_z z^2 \mathcal{L}(z; 0; \psi)\mathcal{N}(z; \tilde{p}, \mu^p) \tag{B.15}$$

$$= 2y\hat{z} - y^2 + \frac{C}{C}\int_z z^2 \mathcal{N}_-(z; \tilde{p}, \mu^p) + \frac{\overline{C}}{C}\int_z z^2 \mathcal{N}_+(z; \tilde{p}, \mu^p). \tag{B.16}$$

Given that $\mathsf{u} \sim \mathcal{N}(a, b^2)$, [117] shows that

$$\begin{aligned}\mathrm{E}\{\mathsf{u}^2 \,|\, \mathsf{u}{>}0\} &= \mathrm{var}\{\mathsf{u} \,|\, \mathsf{u}{>}0\} + \mathrm{E}\{\mathsf{u} \,|\, \mathsf{u}{>}0\}^2 \\ &= b^2 g(-\tfrac{a}{b}) + \big(a + bh(-\tfrac{a}{b})\big)^2, \end{aligned} \tag{B.17}$$

$$\begin{aligned}\mathrm{E}\{\mathsf{u}^2 \,|\, \mathsf{u}{<}0\} &= \mathrm{var}\{\mathsf{u} \,|\, \mathsf{u}{<}0\} + \mathrm{E}\{\mathsf{u} \,|\, \mathsf{u}{<}0\}^2 \\ &= b^2 g(\tfrac{a}{b}) + \big(a - bh(\tfrac{a}{b})\big)^2, \end{aligned} \tag{B.18}$$

where $g(\cdot)$ is defined in (5.22). Inserting (B.17) and (B.18) into (B.16) and noting that $\mathrm{var}\{\mathsf{z} \,|\, \mathsf{p}{=}\hat{p}; \mu^p\} = \mathrm{E}\{\mathsf{z}^2 \,|\, \mathsf{p}{=}\hat{p}; \mu^p\} - \mathrm{E}\{\mathsf{z}|\mathsf{p}{=}\hat{p}; \mu^p\}^2$, we obtain (5.18).

B.2.2 EM Update for Laplacian Rate

Inserting the Laplacian likelihood (5.13) into (5.46), we see that the EM update for the Laplacian rate parameter ψ becomes.

$$\psi^{i+1} = \arg\max_{\psi} \sum_{m=1}^{M} \hat{\mathrm{E}}\{\ln \mathcal{L}(y_m; \boldsymbol{a}_m^{\mathsf{T}}\boldsymbol{x}, \psi) \,\big|\, \boldsymbol{y}; \psi^i\} \tag{B.19}$$

$$= \arg\max_{\psi} M \ln\psi - \psi \sum_{m=1}^{M} \hat{\mathrm{E}}\{|\boldsymbol{a}_m^{\mathsf{T}}\boldsymbol{x} - y_m| \,\big|\, \boldsymbol{y}; \psi^i\}. \tag{B.20}$$

Zeroing the derivative of the objective in (B.20) w.r.t. ψ yields the update (5.47). The expectation in (B.20) can be written as

$$\hat{\mathrm{E}}\{|\boldsymbol{a}_m^{\mathsf{T}}\boldsymbol{x}-y_m|\,\big|\,\boldsymbol{y};\psi^i\}=\int_{\boldsymbol{x}}|\boldsymbol{a}_m^{\mathsf{T}}\boldsymbol{x}-y_m|\,p_{\boldsymbol{x}|\boldsymbol{y}}(\boldsymbol{x}|\boldsymbol{y};\psi^i), \tag{B.21}$$

where $p_{\boldsymbol{x}|\boldsymbol{y}}(\boldsymbol{x}|\boldsymbol{y};\psi^i)$ is taken to be the product of the approximated GAMP marginal posteriors in (2.5).

In the large system limit, the central limit theorem implies that $\mathsf{z}_m \triangleq \boldsymbol{a}_m^{\mathsf{T}}\boldsymbol{x}$, when conditioned on $\boldsymbol{y}=\boldsymbol{y}$, is $\mathcal{N}(\boldsymbol{a}_m^{\mathsf{T}}\hat{\boldsymbol{x}},\sum_{n=1}^N \boldsymbol{a}_{mn}^2\mu_n^x)$, yielding the approximation

$$\hat{\mathrm{E}}\{|\boldsymbol{a}_m^{\mathsf{T}}\boldsymbol{x}-y_m|\,\big|\,\boldsymbol{y};\psi^i\}\approx\int_{z_m}|z_m-y_m|\,\mathcal{N}(z_m;\boldsymbol{a}_m^{\mathsf{T}}\hat{\boldsymbol{x}},\sum_{n=1}^N\boldsymbol{a}_{mn}^2\mu_n^x) \tag{B.22}$$

$$=\int_{z'_m}|z'_m|\,\mathcal{N}(z'_m;\boldsymbol{a}_m^{\mathsf{T}}\hat{\boldsymbol{x}}-y_m,\sum_{n=1}^N\boldsymbol{a}_{mn}^2\mu_n^x). \tag{B.23}$$

Defining $\tilde{z}_m \triangleq \boldsymbol{a}_m^{\mathsf{T}}\hat{\boldsymbol{x}}-y_m$, and using a derivation similar to that used for (B.10), leads to (5.48).

B.3 Derivation of NNGM-GAMP Quantities

B.3.1 BNNGM Prior Steps for Sum-product GAMP

Inserting the Bernoulli NNGM prior (5.31) into the GAMP approximated posterior (2.5), the posterior mean in line (R13) of Table 2.1 becomes (removing the n subscript for brevity)

$$\hat{x}\triangleq\mathrm{E}\{\mathsf{x}\,|\,\mathsf{r}=\hat{r};\mu^r\}=\int_x x\,p_{\mathsf{x}|\mathsf{r}}(x|\hat{r};\mu^r) \tag{B.24}$$

$$=\frac{1}{\zeta}\int_+ x\,\mathcal{N}(x;\hat{r},\mu^r)\Big((1-\tau)\delta(x)+\tau\sum_{\ell=1}^L\omega_\ell\mathcal{N}_+(x;\theta_\ell,\phi_\ell)\Big)$$

$$=\frac{\tau}{\zeta}\sum_{\ell=1}^L\omega_\ell\int_+ x\mathcal{N}(x;\hat{r},\mu^r)\mathcal{N}_+(x;\theta_\ell,\phi_\ell), \tag{B.25}$$

where $\zeta \triangleq \int_x p_{\mathsf{x}}(x)\mathcal{N}(x;\hat{r},\mu^r)$ is a scaling factor. Using the Gaussian-pdf multiplication rule,[48] we get

$$\hat{x} = \frac{\tau}{\zeta}\sum_{\ell=1}^{L}\frac{\omega_\ell \mathcal{N}(\hat{r};\theta_\ell,\mu^r+\phi_\ell)}{\Phi_c(-\theta_\ell/\sqrt{\phi_\ell})}\int_+ x\,\mathcal{N}(x;\gamma_\ell,\nu_\ell), \tag{B.26}$$

with γ_ℓ and ν_ℓ defined in (5.37) and (5.38), respectively.

Using similar techniques, the scaling factor

$$\zeta = \int_+ \mathcal{N}(x;\hat{r},\mu^r)\Big((1-\tau)\delta(x)+\tau\sum_{\ell=1}^{L}\omega_\ell\mathcal{N}_+(x;\theta_\ell,\phi_\ell)\Big) \tag{B.27}$$

can be shown to be equivalent to (5.35). Finally, using the mean of a truncated Gaussian (B.11), and inserting (5.35) into (B.26), we get the NNGM-GAMP estimate (5.33).

To calculate the variance of the GAMP approximated posterior (2.5), we note that

$$\mu^x \triangleq \mathrm{var}\{\mathsf{x}\,|\,\mathsf{r}=\hat{r};\mu^r\} = \int_+ x^2 p_{x|\mathsf{r}}(\mathsf{x}|\hat{r};\mu^r) - \mathrm{E}\{\mathsf{x}\,|\,\mathsf{r}=\hat{r};\mu^r\}^2. \tag{B.28}$$

Following (B.24)-(B.26) and using the Gaussian-pdf multiplication rule, we find the second moment to be

$$\int_+ x^2 p_{\mathsf{x}|\mathsf{r}}(x|\hat{r};\mu^r) = \frac{\tau}{\zeta}\sum_{\ell=1}^{L}\frac{\beta_\ell}{\Phi_c(\alpha_\ell)}\int_+ x^2\mathcal{N}\left(x;\gamma_\ell,\nu_\ell\right), \tag{B.29}$$

where β_ℓ and α_ℓ are given in (5.39) and (5.36), respectively.

Leveraging the second moment of a truncated Gaussian (B.17) in (B.29), and then inserting (5.33) and (B.29) into (B.28), we obtain the NNGM-GAMP variance estimate (5.34).

[48] $\mathcal{N}(x;a,A)\mathcal{N}(x;b,B) = \mathcal{N}\left(x;\frac{a/A+b/B}{1/A+1/B},\frac{1}{1/A+1/B}\right)\mathcal{N}(a;b,A+B)$.

B.3.2 EM Updates of NNGM Parameters

We first derive the EM update for θ_k, the k^{th} component location, given the previous parameter estimate $\boldsymbol{q}^i$. The maximizing value of θ_k in (5.52) is necessarily a value of θ_k that zeros the derivative of the sum, i.e., that satisfies

$$\frac{d}{d\theta_k}\sum_{n=1}^{N}\int_{x_n} p_{\mathsf{x}|\mathsf{r}}\Big(x_n|\hat{r}_n;\mu_n^r,\boldsymbol{q}^i\Big)\ln p_{\mathsf{x}}\left(x_n;\theta_k,\boldsymbol{q}^i_{\backslash\theta_k}\right)=0 \tag{B.30}$$

$$\Leftrightarrow \sum_{n=1}^{N}\int_{x_n} p_{\mathsf{x}|\mathsf{r}}\Big(x_n|\hat{r}_n;\mu_n^r,\boldsymbol{q}^i\Big)\frac{d}{d\theta_k}\ln p_{\mathsf{x}}\left(x_n;\theta_k,\boldsymbol{q}^i_{\backslash\theta_k}\right)=0. \tag{B.31}$$

For all $x_n \geq 0$, the derivative in (B.31) can be written as

$$\frac{d}{d\theta_k}\ln p_{\mathsf{x}}(x_n;\theta_k,\boldsymbol{q}^i_{\backslash\theta_k}) = \frac{\frac{d}{d\theta_k}\tau^i\omega_k^i\frac{\mathcal{N}(x_n;\theta_k;\phi_k^i)}{\Phi_c(-\theta_k/\sqrt{\phi_k^i})}}{p_{\mathsf{x}}(x_n;\theta_k,\boldsymbol{q}^i_{\backslash\theta_k})}. \tag{B.32}$$

Because plugging (B.32) into (B.31) yields an intractable integral, we use the approximation[49] $\Phi_c(-\theta_k/\sqrt{\phi_k^i}) \approx \Phi_c(-\theta_k^i/\sqrt{\phi_k^i})$, yielding

$$\frac{d}{d\theta_k}\ln p_{\mathsf{x}}(x_n;\theta_k,\boldsymbol{q}^i_{\backslash\theta_k}) = \left(\frac{x_n-\theta_k}{\phi_k^i}\right) \tag{B.33}$$
$$\times\frac{\tau^i\omega_k^i\mathcal{N}(x_n;\theta_k,\phi_k^i)/\Phi_c\Big(-\theta_k^i/\sqrt{\phi_k^i}\Big)}{(1-\tau^i)\delta(x_n)+\tau^i(\omega_k^i\mathcal{N}_+(x_n;\theta_k,\phi_k^i)+\sum_{\ell\neq k}\omega_\ell^i\mathcal{N}_+(x_n;\theta_\ell^i,\phi_\ell^i))}.$$

We also note that (B.33) is zero at $x_n = 0$ due to the Dirac delta function in the denominator.

Now, plugging in (B.33) and the approximated GAMP posterior $p_{\mathsf{x}|\mathsf{r}}(x_n|\hat{r}_n;\mu_n^r,\boldsymbol{q}^i)$ from (2.5), integrating (B.31) separately over $[\epsilon,\infty)$ and its complement, and taking $\epsilon\to 0$, we find that the $(-\infty,\epsilon)$ portion vanishes, giving the necessary condition

$$\sum_{n=1}^{N}\int_{+\zeta_n}\frac{\hat{p}(x_n|x_n\neq 0,\boldsymbol{y};\boldsymbol{q}^i)\omega_k^i\frac{\mathcal{N}(x_n;\theta_k,\phi_k^i)}{\Phi_c(-\theta_k^i/\sqrt{\phi_k^i})}(x_n-\theta_k)}{\Big(\omega_k^i\mathcal{N}_+(x_n;\theta_k,\phi_k^i)+\sum_{\ell\neq k}\omega_\ell^i\mathcal{N}_+(x_n;\theta_\ell^i,\phi_\ell^i)\Big)}=0, \tag{B.34}$$

[49] This approximation becomes more accurate as $\frac{d}{d\theta_k}\Phi_c(-\theta_k/\sqrt{\phi_k})$ tends to zero, i.e., when $\theta_k/\sqrt{\phi_k}$ gets large, which was observed for the real-world experiments considered in Chapter 3.2.

where $\hat{p}(x_n|x_n \neq 0, \boldsymbol{y}; \boldsymbol{q}^i) \triangleq p_{\mathsf{x}|\mathsf{r}}(x_n|\hat{r}_n, x_n \neq 0; \mu_n^r, \boldsymbol{q}^i)$. Since this integral cannot be evaluated in closed form, we apply the approximation $\mathcal{N}(x_n; \theta_k, \phi_k^i) \approx \mathcal{N}(x_n; \theta_k^i, \phi_k^i)$ in both the numerator and denominator, and subsequently exploit the fact that, for $x_n \geq 0$, $\hat{p}(x_n|x_n \neq 0, \boldsymbol{y}; \boldsymbol{q}^i) = \mathcal{N}(x_n; \hat{r}_n, \mu_n^r) \sum_\ell \omega_\ell^i \mathcal{N}_+(x_n; \theta_\ell^i, \phi_\ell^i)$ from (2.5) to cancel terms, where we obtain the necessary condition

$$\sum_{n=1}^{N} \int_+ \frac{\omega_k^i \mathcal{N}(x_n; \hat{r}_n, \mu_n^r) \mathcal{N}_+(x_n; \theta_k^i, \phi_k^i)}{\zeta_n} (x_n - \theta_k) = 0. \tag{B.35}$$

Now using the Gaussian-pdf multiplication rule, we get

$$\sum_{n=1}^{N} \frac{\beta_{n,k}}{\Phi_c(\alpha_{n,k})} \int_+ \mathcal{N}(x_n; \gamma_{n,k}, \nu_{n,k})(x_n - \theta_k) = 0. \tag{B.36}$$

Following similar techniques as in Appendix B.3.1 and noting that $\beta_{n,k} = \pi_n \overline{\beta}_{n,k}$, we see that the update θ_k^{i+1} in (5.55) is the value of θ_k that satisfies (B.36).

Similarly, the maximizing value of ϕ_k in (5.53) is necessarily a value of ϕ_k that zeroes the derivative, i.e.,

$$\sum_{n=1}^{N} \int_{x_n} p_{\mathsf{x}|\mathsf{r}}(x_n|\hat{r}_n; \mu_n^r, \boldsymbol{q}^i) \frac{d}{d\phi_k} \ln p_{\mathsf{x}}(x_n; \phi_k, \boldsymbol{q}^i_{\setminus \phi_k}) = 0. \tag{B.37}$$

Using the prior given in (5.31), and simultaneously applying the approximation $\Phi_c(-\theta_k^i/\sqrt{\phi_k}) \approx \Phi_c(-\theta_k^i/\sqrt{\phi_k^i})$, we see that the derivative in (B.37) can be written as

$$\begin{aligned} \frac{d}{d\phi_k} \ln p_{\mathsf{x}}(x_n; \phi_k, \boldsymbol{q}^i_{\setminus \phi_k}) = & \frac{1}{2} \left(\frac{(x_n - \theta_k^i)^2}{\phi_k^2} - \frac{1}{\phi_k} \right) \\ & \times \frac{\tau^i \omega_k^i \mathcal{N}(x_n; \theta_k, \phi_k^i) / \Phi_c(-\theta_k^i/\sqrt{\phi_k^i})}{(1-\tau^i)\delta(x_n) + \tau^i(\omega_k^i \mathcal{N}_+(x_n; \theta_k^i, \phi_k) + \sum_{\ell \neq k} \omega_\ell^i \mathcal{N}_+(x_n; \theta_\ell^i, \phi_\ell^i))}. \end{aligned} \tag{B.38}$$

Integrating (B.37) separately over $(-\infty, \epsilon)$ and $[\epsilon, \infty)$, and taking $\epsilon \to 0$, we see that the $(-\infty, \epsilon)$ portion vanishes, giving

$$\sum_{n=1}^{N} \int_{+} \frac{\hat{p}(x_n | x_n \neq 0, \boldsymbol{y}; \boldsymbol{q}^i) \omega_k^i \mathcal{N}(x_n; \theta_k^i, \phi_k) / \Phi_c(-\theta_k^i / \sqrt{\phi_k^i})}{\zeta_n (\omega_k^i \mathcal{N}(x_n; \theta_k^i, \phi_k) + \sum_{\ell \neq k} \omega_\ell^i \mathcal{N}(x_n; \theta_\ell^i, \phi_\ell^i))} \times \left(\frac{(x_n - \theta_k^i)^2}{\phi_k} - 1 \right) = 0. \tag{B.39}$$

Again, this integral is difficult, so we approximate $\mathcal{N}(x_n; \theta_k, \phi_k^i) \approx \mathcal{N}(x_n; \theta_k^i, \phi_k^i)$ in both the numerator and denominator. After some cancellation (as in (B.34)), we get the necessary condition

$$\sum_{n=1}^{N} \int_{+} \frac{\mathcal{N}(x_n; \hat{r}_n, \mu_n^r) \omega_k^i \mathcal{N}_+(x_n; \theta_k^i, \phi_k^i)}{\zeta_n} \left(\frac{(x_n - \theta_k^i)^2}{\phi_k} - 1 \right) = 0. \tag{B.40}$$

To find the value of ϕ_k that solves (B.40), we expand $(x_n - \theta_k^i)^2 = x_n^2 - 2x_n\theta_k^i + (\theta_k^i)^2$ and apply the Gaussian-pdf multiplication rule, yielding

$$\sum_{n=1}^{N} \frac{\beta_{n,k}}{\Phi_c(\alpha_{n,k})} \int_{+} \mathcal{N}(x_n; \gamma_{n,k}, \nu_{n,k}) \left(\frac{x_n^2 - 2x_n\theta_k^i + (\theta_k^i)^2}{\phi_k} - 1 \right) = 0. \tag{B.41}$$

Using similar techniques as in Appendix B.3.1 and simplifying, we see that ϕ_k^{i+1} in (5.56) is the value of ϕ_k that solves (B.41).

Finally, we calculate the EM update in (5.54) for positive $\boldsymbol{\omega}$ under the pmf constraint $\sum_{k=1}^{L} \omega_k = 1$ by solving the augmented Lagrangian $\max_{\boldsymbol{\omega}, \xi} J(\boldsymbol{\omega}, \xi)$, where ξ is a Lagrange multiplier and

$$J(\boldsymbol{\omega}, \xi) \triangleq \sum_{n=1}^{N} \hat{\mathrm{E}} \left\{ \ln p_{\mathsf{x}}(\mathsf{x}_n; \boldsymbol{\omega}, \boldsymbol{q}_{\setminus \boldsymbol{\omega}}^i) \,\middle|\, \boldsymbol{y}; \boldsymbol{q}^i \right\} - \xi \left(\sum_{\ell=1}^{L} \omega_\ell - 1 \right). \tag{B.42}$$

First, we set $\frac{d}{d\omega_k} J(\boldsymbol{\omega}, \xi) = 0$, which yields

$$\sum_{n=1}^{N} \int_{x_n} \frac{p_{\mathsf{x}}(x_n; \boldsymbol{q}^i) \mathcal{N}(x_n; \hat{r}_n, \mu_n^r)}{\zeta_n} \frac{d}{d\omega_k} \ln p_{\mathsf{x}}(x_n; \boldsymbol{\omega}, \boldsymbol{q}_{\setminus \boldsymbol{\omega}}^i) = \xi \tag{B.43}$$

where, for non-negative x_n,

$$\frac{d}{d\omega_k} \ln p_{\mathsf{x}}\Big(x_n; \boldsymbol{\omega}, \boldsymbol{q}^i_{\setminus \boldsymbol{\omega}}\Big) = \frac{\tau^i \mathcal{N}_+(x_n; \theta^i_k, \phi^i_k)}{p_{\mathsf{x}}\Big(x_n; \boldsymbol{\omega}, \boldsymbol{q}^i_{\setminus \boldsymbol{\omega}}\Big)}. \tag{B.44}$$

Inserting (B.44) into (B.43), we get

$$\sum_{n=1}^{N} \int_+ \frac{p_{\mathsf{x}}(x_n; \boldsymbol{q}^i) \mathcal{N}(x_n; \hat{r}_n, \mu^r_n)}{\zeta_n} \frac{\tau^i \mathcal{N}_+(x_n; \theta^i_k, \phi^i_k)}{p_{\mathsf{x}}(x_n; \boldsymbol{\omega}, \boldsymbol{q}^i_{\setminus \boldsymbol{\omega}})} \;\; = \;\; \xi. \tag{B.45}$$

As in (B.34) and (B.39), the above integral is difficult to evaluate, and so we apply the additional approximation $\boldsymbol{\omega} \approx \boldsymbol{\omega}^i$, which reduces the previous equation to

$$\xi = \sum_{n=1}^{N} \int_+ \frac{\tau^i \mathcal{N}_+(x_n; \theta^i_k, \phi^i_k) \mathcal{N}(x_n; \hat{r}_n, \mu^r_n)}{\zeta_n}. \tag{B.46}$$

We then multiply both sides by ω^i_k for $k = 1, \ldots, L$, and sum over k. Leveraging the fact $1 = \sum_k \omega^i_k$, and simplifying, we obtain the equivalent condition

$$\xi \;\; = \;\; \sum_{n=1}^{N} \int_+ \frac{\tau^i \sum_{k=1}^{L} \omega^i_k \mathcal{N}_+(x_n; \theta^i_k, \phi^i_k) \mathcal{N}(x_n; \hat{r}_n, \mu^r_n)}{\zeta_n} \tag{B.47}$$

$$= \;\; \sum_{n=1}^{N} \frac{\tau^i}{\zeta_n} \sum_{k=1}^{L} \beta_{n,k} \int_+ \frac{\mathcal{N}(x_n; \gamma_{n,k}, \phi_{n,k})}{\Phi_c(\alpha_{n,k})} = \sum_{n=1}^{N} \pi_n. \tag{B.48}$$

Plugging (B.48) into (B.46) and multiplying both sides by ω_k, the derivative-zeroing value of ω_k is seen to be

$$\omega_k = \frac{\sum_{n=1}^{N} \int_+ \tau^i \omega_k \mathcal{N}_+(x_n; \theta^i_k, \phi^i_k) \mathcal{N}(x_n; \hat{r}_n, \mu^r_n)/\zeta_n}{\sum_{n=1}^{N} \pi_n}, \tag{B.49}$$

where, if we use $\omega_k \approx \omega^i_k$ on the right of (B.43), then we obtain the approximate EM update ω^{i+1}_k in (5.57).

Appendix C: HUT-AMP Derivations

C.1 Mean removal

We can see that $\underline{\boldsymbol{S}}$ from (6.5) is approximately zero-mean via

$$0 = \frac{1}{MT}\sum_{m=1}^{M}\sum_{t=1}^{T}\underline{y}_{mt} \tag{C.1}$$

$$= \underbrace{\frac{1}{MT}\sum_{m=1}^{M}\sum_{t=1}^{T}\sum_{n=1}^{N}\underline{s}_{mn}a_{nt}}_{O(1)} + \underbrace{\frac{1}{MT}\sum_{m=1}^{M}\sum_{t=1}^{T}w_{mt}}_{O(1/N)} \tag{C.2}$$

$$\approx \sum_{n=1}^{N}\underbrace{\frac{1}{T}\sum_{t=1}^{T}a_{nt}}_{\triangleq\,\mu_n^a}\frac{1}{M}\sum_{m=1}^{M}\underline{s}_{mn}, \tag{C.3}$$

where (C.1) follows from the definitions (3.14)-(6.4). The underbraces in (C.2) show the scaling on each term in the large-system limit (i.e., as $N \to \infty$). These particular scalings follow from our assumption that the noise is both zero-mean and white and the convention [29] that both y_{mt} and the noise variance ψ scale as $O(1)$. Recalling that $\sum_{n=1}^{N}\mu_n^a = 1$ due to the simplex constraint, expression (C.3) shows that a weighted average of elements in $\underline{\boldsymbol{S}}$ is approximately zero, where the approximation becomes exact in the large-system limit.