UNIVERSITY OF CALIFORNIA SAN DIEGO

Message Passing Algorithms and Extensions of Sparse Bayesian Learning

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Electrical Engineering (Signal and Image Processing)

by

Maher Al-Shoukairi

Committee in charge:

Professor Bhaskar D. Rao, Chair
Professor Sanjoy Dasgupta
Professor Truong Nguyen
Professor Rayan Saab
Professor Paul Siegel

2021

The dissertation of Maher Al-Shoukairi is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2021

DEDICATION

To my family.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

First and foremost I would like to express my deepest gratitude for prof. Bhaskar Rao for giving me the opportunity to pursue a PhD under his supervision. I am very lucky to have had the chance to work under prof. Rao's guidance and to learn from his vast knowledge, kindness and patience. Prof. Rao will always be a role model that has set the highest academic and ethical standards, that I will always strive to achieve.

I would like to thank my father Shafiq Khalawi Alshuqairi, who always encouraged me to achieve my full potential. A big thank you to my best friend, my biggest fan and the best teacher anyone could hope for, my dear mom Salam Arikat, whose love and support have always paved our way. I am also thankful for my brother Musa, my first mentor and the person I seek advice from on a daily basis. I would like to thank Ashleigh for her encouragement and all the inspiration she has provided during my PhD.

I am thankful for my committee members, prof. Truong Nguyen, prof. Paul Siegel, prof. Sanjoy DasGupta and prof. Rayan Saab, for their valuable suggestions during my course of work. I would like to thank all the inspiring professors of UCSD whose courses gave me the skills that enabled me to pursue different research problems. I especially thank prof. Kenneth Kreutz-Delgado for his early on encouragement and his uplifting conversations. I would like to thank prof. Philip Schniter (of The Ohio State University) for giving me the chance to collaborate with him on my first paper, I really appreciate his valuable contributions and feedback.

I am thankful for my colleagues at UCSD Digital Signal Processing Lab, Dr. Bang Nguyen, Dr. Igor Fedorov, Dr. Elina Nayebi, Dr. Alican Nalci, Dr. Ritwik Giri, Dr. Yonghee Han, Dr. Yacong Ding, Dr. Soon-En Chiu, Dr. Jing Liu, Govind Gopal, Tharun Srikrishnan, Rohan Pote, Aditya Sant, Hitesh Khunti and David Ho. I am thankful for their moral support, valuable discussions, help with research and research presentations from which I learned a lot.

Finally, I would like to thank some of my former colleagues and managers at Qualcomm for providing me with the inspiration and motivation to pursue my academic goals and eventually

transition into the next chapter of my career.

Chapter 2, in full, is a reprint of material published in the article Maher Al-Shoukairi, Philip Schniter, and Bhaskar D. Rao, "A GAMP-based low complexity sparse Bayesian learning algorithm", IEEE Transactions on Signal Processing, 2017. I was the primary author and B. D. Rao supervised the research.

Chapter 3, in full, is a reprint of material published in the article Maher Al-Shoukairi, and Bhaskar D. Rao, "A GAMP based algorithm with hierarchical priors for recovering non-negative sparse signals", Asilomar Conference on Signals, Systems, and Computers, 2017. I was the primary author and B. D. Rao supervised the research.

Chapter 4, in full, is a reprint of material published in the article Maher Al-Shoukairi, and Bhaskar D. Rao, "An MPDR Perspective and Extension of Sparse Bayesian Learning", under review at IEEE Transactions on Signal Processing. I was the primary author and B. D. Rao supervised the research.

Chapter 5, in full, is a reprint of material in Maher Al-Shoukairi, and Bhaskar D. Rao, "An MPDR Perspective and Extension of Sparse Bayesian Learning", under review at IEEE Transactions on Signal Processing. I was the primary author and B. D. Rao supervised the research.

Chapter 6, in full, is a reprint of material in Maher Al-Shoukairi, and Bhaskar D. Rao, "Semi-Blind Channel Estimation In MIMO Systems with Discrete Priors On Data Symbols", under review at IEEE Transactions on Signal Processing. I was the primary author and B. D. Rao supervised the research.

VITA

2005	B. S. in Electrical Engineering , University of Jordan, Amman
2008	M. S. in Electrical Engineering , Texas A&M University, College Station
2021	Ph. D. in Electrical Engineering, University of California San Diego

PUBLICATIONS

M Al-Shoukairi, BD Rao, "Semi-Blind Channel Estimation In MIMO Systems with Discrete Priors On Data Symbols", *under review at IEEE Signal Processing Letters*.

M Al-Shoukairi, BD Rao, "An MPDR Perspective and Extension of Sparse Bayesian Learning", *under review at IEEE Transactions on Signal Processing*.

M Al-Shoukairi, BD Rao, "Sparse Signal Recovery using MPDR Estimation", *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

A Nalci, I Fedorov, **M Al-Shoukairi**, TT Liu, BD Rao, "Rectified Gaussian scale mixtures and the sparse non-negative least squares problem", *IEEE Transactions on Signal Processing*, 2018.

M Al-Shoukairi, P Schniter, BD Rao, "A GAMP-based low complexity sparse Bayesian learning algorithm", *IEEE Transactions on Signal Processing*, 2017.

M Al-Shoukairi, BD Rao, "A GAMP based algorithm with hierarchical priors for recovering non-negative sparse signals", *Asilomar Conference on Signals, Systems, and Computers*, 2017.

M Al-Shoukairi, BD Rao, "Sparse Bayesian learning using approximate message passing", *Asilomar Conference on Signals, Systems, and Computers*, 2014.

ABSTRACT OF THE DISSERTATION

Message Passing Algorithms and Extensions of Sparse Bayesian Learning

by

Maher Al-Shoukairi

Doctor of Philosophy in Electrical Engineering (Signal and Image Processing)

University of California San Diego, 2021

Professor Bhaskar D. Rao, Chair

Sparse Signal Recovery (SSR) has an essential role in a number of modern engineering applications. This thesis focuses on Bayesian algorithms for sparse signal recovery, where it addresses some of the shortcomings associated with such algorithms. The high complexity of the sparse Bayesian learning algorithm is addressed first, where we present an algorithm that incorporates damped Gaussian generalized approximate message passing (GGAMP) into Expectation-Maximization (EM)-based sparse Bayesian learning (SBL). In particular, GGAMP is used to implement the E-step in SBL in place of matrix inversion. We propose an algorithm that is much more robust to arbitrary measurement matrices than the standard damped GAMP algorithms while being much lower complexity than the standard SBL algorithm. We then

extend the approach from the single measurement vector (SMV) case to the temporally correlated multiple measurement vector (MMV) case.

The approach developed for the standard SBL is extended to address the sparse non-negative least squares (NNLS) problem using a rectified Gaussian scale mixture approach combined with the generalized approximate message passing algorithm. This approach enhances convergence compared to existing GAMP based sparse NNLS algorithms. Other advantages include significant reduction in derivation complexity of the algorithm, and the ability to impose different priors on the signal, simply by changing the less computationally demanding M-step. Moreover, extending the algorithm to the multiple measurement vector case is straightforward, and is achieved by a simple modification to the M-step as well.

Next, we provide a new perspective of the SBL algorithm. A novel interpretation of the SBL algorithm's iterations is developed, where the iterations are divided into a minimum power distortionless receiver (MPDR) step and a denoising step. This interpretation provides significant intuitive insights into the SBL, which can potentially enable enhancing the algorithm and extending it beyond some of its current limitations. To demonstrate this potential, we propose a low complexity algorithm that can handle a wide range of sparsity promoting priors. We also show how the new perspective on SBL extends to other variants of the algorithm, such as the sequential fast-SBL algorithm and the multiple measurement vector (MMV) SBL variants. We demonstrate potential benefits of such interpretation by extending the fast-SBL to incorporate more general priors, and by developing a low complexity MMV algorithm.

Finally, we address the MIMO semi-blind channel estimation problem, benefiting from the insights gained from previous results. We propose an eigenvalue decomposition based technique to significantly reduce the dimensionality of the Gaussian EM based estimation algorithm, greatly lowering the computational complexity. In addition to that, we apply the MPDR based decoupling principle to derive a tractable EM algorithm that uses the actual discrete prior of the data symbols.

Chapter 1

Introduction

1.1 Sparse Signal Recovery

The problem of sparse signal recovery (SSR) and the related problem of compressed sensing have received much attention in recent years [1–6]. Sparse signal recovery continues to be deployed in an increasing number of engineering applications, the applications include EEG/MEG [7, 8], array signal processing [9–11], pattern recognition [12], speech and audio processing [13], wireless sensor networks [14, 15], wireless channel estimation [16, 17] and many more.

The SSR problem, in the single measurement vector (SMV) case, consists of recovering a sparse signal $\boldsymbol{x} \in \mathbb{R}^N$ from $M \leq N$ noisy linear measurements $\boldsymbol{y} \in \mathbb{R}^M$:

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{e}, \tag{1.1}$$

where $\boldsymbol{A} \in \mathbb{R}^{M \times N}$ is a known measurement matrix and $\boldsymbol{e} \in \mathbb{R}^M$ is additive noise modeled by $\boldsymbol{e} \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I})$. Although this is an underdetermined system, a unique solution can be obtained when sufficient sparsity and appropriate conditions on the measurement matrix are met. The

sparse solution can be obtained by solving the following optimization problem:

$$\underset{\boldsymbol{x}}{\operatorname{argmin}} \|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda\|\boldsymbol{x}\|_0, \tag{1.2}$$

where $\|.\|_0$ is the zero pseudo-norm which is a measure of the support and λ is a regularization parameter related to the noise variance. The optimization in (1.2) is NP hard [5, 18], it requires an exhaustive search over all subsets of columns of $\boldsymbol{A}$. Despite the difficulty in solving this problem, an important finding in recent years is that for a sufficiently sparse $\boldsymbol{x}$ and a well designed $\boldsymbol{A}$, accurate recovery of $\mathbf{x}$ is possible [18]. A popular technique to address (1.1) is based on approximating the zero norm penalty factor in (1.2) by a suitable surrogate penalty factor $g(\boldsymbol{x})$:

$$\underset{\boldsymbol{x}}{\operatorname{argmin}} \|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|_2^2 + \tilde{\lambda}g(\boldsymbol{x}). \tag{1.3}$$

Different SSR algorithms can be obtained using the proper choice of the penalty factor $g(\boldsymbol{x})$. It was shown that under some conditions the choice of strictly increasing concave penalty factor results in an objective function with sparse local minima with the sparsest solution at the global minimum [19, 20]. A popular choice of $g(\boldsymbol{x})$ is the ℓ_1 norm, leading to the Basis Pursuit and LASSO algorithms [21, 22]. The popularity of the ℓ_1 approach is due to the theoretical guarantees of exact recovery when certain conditions are met. More specifically the ℓ_1 minimization solution recovers the ℓ_0 solution if $\boldsymbol{A}$ satisfies the restricted isometry property (RIP) [1, 23]. In addition to the convex relaxation of (1.2), a range of different techniques were proposed to address (1.1). The techniques include greedy algorithms, [5, 24–26], iteratively re-weighted algorithms [27–29] and Bayesian techniques [30–37]. Greedy algorithms can achieve lower complexity compared to other techniques, however they have limited success in noisy scenarios and are highly affected by coherence in the columns of $\boldsymbol{A}$.

A number of approaches have been empirically shown to outperform the ℓ_1 approach, the approaches include reweighted ℓ_1, ℓ_2 algorithms [38], in addition to Bayesian algorithms

including those based on approximate message passing [31–33]. The Bayesian approach is based on incorporating the sparsity constraint by choosing a sparsity promoting prior on the vector $\boldsymbol{x}$. Bayesian approaches can be mainly divided into two types: Type I MAP based approaches, and a Type II Evidence Maximization approaches. The surrogate function in (1.3) can be imposed using a type I MAP Bayesian approach through the proper prior:

$$\hat{\boldsymbol{x}}_{MAP} = \operatorname*{argmax}_{\boldsymbol{x}} p(\boldsymbol{x}|\boldsymbol{y}) = \operatorname*{argmax}_{\boldsymbol{x}} p(\boldsymbol{y}|\boldsymbol{x})p(\boldsymbol{x}) \tag{1.4}$$

$$= \operatorname*{argmax}_{\boldsymbol{x}} \log\left[p(\boldsymbol{y}|\boldsymbol{x})\right] + \log\left[p(\boldsymbol{x})\right] \tag{1.5}$$

$$= \operatorname*{argmin}_{\boldsymbol{x}} \|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|_2^2 + \sum_i \log[p(x_i)], \tag{1.6}$$

therefore the surrogate function where $g(\boldsymbol{x})$ can be determined by $\log p(\boldsymbol{x})$. For example, using an i.i.d. Laplace prior $p(x_i) = \frac{\tilde{\lambda}}{2}\exp(-\tilde{\lambda}\|x_i\|_1)$ will lead to the popular ℓ_1 minimization approach. In the type II a prior is typically imposed using a hierarchical model, where the effective prior is controlled by an unknown parameter that can be learned from the measurements using evidence maximization. The implementation and performance differences between the two Bayesian approaches were discussed in [31, 39–43]. Type II based algorithms were shown to experience more robustness to the inaccurate choice of prior compared to Type I. In addition to that, certain hierarchical models like the Gaussian scale mixture can allow for closed form solutions and other desirable properties as we will see in parts of this dissertation. This dissertation will mainly focus on Bayesian algorithms, more specifically the algorithm known as the sparse Bayesian learning algorithm (SBL) and its variants, and on approximate message passing (AMP) algorithms.

1.1.1 Sparse Bayesian Learning Algorithm

SBL was first proposed in the context of machine learning in 2001 [30], and it was adapted to be used for SSR in 2004 [31, 41]. We give a very brief description of SBL [30, 31] here, saving a more detailed description for subsequent chapters. Essentially, SBL is a type II Bayesian

approach that is based on a Gaussian scale mixture (GSM) [44–46] prior on $\boldsymbol{x}$. That is, the prior is Gaussian conditioned on a variance vector $\boldsymbol{\gamma}$, which is then controlled by a suitable choice of hyperprior $p(\boldsymbol{\gamma})$. A large of number of sparsity-promoting priors, like the Student-t and Laplacian priors, can be modeled using a GSM, making the approach widely applicable [44–48]. In the SBL algorithm, the expectation-maximization (EM) algorithm is used to alternate between estimating $\boldsymbol{\gamma}$ and estimating the signal $\boldsymbol{x}$ under fixed $\boldsymbol{\gamma}$. Since the latter step uses a Gaussian likelihood and Gaussian prior, the exact solution can be computed in closed form via matrix inversion. This matrix inversion is computationally expensive, limiting the algorithms applicability to large scale problems. In addition to the high complexity of the algorithm, SBL is limited to imposing priors through a GSM which is quite flexible but not all encompassing. In this dissertation we will be exploring techniques to lower the complexity of the SBL and extend it beyond the GSM prior to allow it to incorporate more general sparsity promoting priors.

1.1.2 Generalized Approximate Message Passing Algorithm

Another important class of algorithms that we address in this dissertation is the Approximate message passing (AMP) algorithms. AMP algorithms apply quadratic and Taylor series approximations to loopy belief propagation to produce low complexity algorithms. Based on the original AMP work in [49], a generalized AMP (GAMP) algorithm was proposed in [50]. The GAMP algorithm provides an iterative Bayesian framework under which the knowledge of the matrix $\boldsymbol{A}$ and the densities $p(\boldsymbol{x})$ and $p(\boldsymbol{y}|\boldsymbol{x})$ can be used to compute the maximum a posteriori (MAP) estimate $\hat{\boldsymbol{x}}_{MAP} = \arg\min_{\boldsymbol{x}\in\mathbb{R}^N} p(\boldsymbol{x}|\boldsymbol{y})$ when it is used in its max-sum mode, or approximate the minimum mean-squared error (MMSE) estimate $\hat{\boldsymbol{x}}_{MMSE} = \int_{\mathbb{R}^N} \boldsymbol{x} p(\boldsymbol{x}|\boldsymbol{y}) d\boldsymbol{x} = \mathbb{E}(\boldsymbol{x}|\boldsymbol{y})$ when it is used in its sum-product mode.

The performance of AMP/GAMP algorithms in the large system limit ($M, N \to \infty$) under an i.i.d zero-mean sub-Gaussian matrix $\boldsymbol{A}$ is characterized by state evolution [51], whose fixed points, when unique, coincide with the MAP or the MMSE estimate. However, when $\boldsymbol{A}$ is generic,

GAMP's fixed points can be strongly suboptimal and/or the algorithm may never reach its fixed points. Previous work has shown that even mild ill-conditioning or small mean perturbations in $\boldsymbol{A}$ can cause GAMP to diverge [52–54]. To overcome the convergence problem in AMP algorithms, a number of AMP modifications have been proposed. A "swept" GAMP (SwAMP) algorithm was proposed in [55], which replaces parallel variable updates in the GAMP algorithm with serial ones to enhance convergence. But SwAMP is relatively slow and still diverges for certain $\boldsymbol{A}$. An adaptive damping and mean-removal procedure for GAMP was proposed in [54] but it too is somewhat slow and still diverges for certain $\boldsymbol{A}$. An alternating direction method of multipliers (ADMM) version of AMP was proposed in [56] with improved robustness but even slower convergence. In [32] another variant of the approximate message passing algorithms was proposed. The algorithm referred to by Vector AMP (VAMP) uses a single time singular value decomposition operation to enhance the convergence of the AMP algorithm.

The class of AMP based algorithms represent an important class of low complexity Bayesian algorithms that can impose flexible priors on the signal of interest. Throughout the dissertation we will exploit some of the results obtained by different AMP algorithms. We will also use some of the previous AMP results as a benchmark to compare our contributions against.

1.1.3 Multiple Measurement Vector (MMV) Model

The MMV problem extends the SMV problem from a single measurement and signal vector to a sequence of measurement and signal vectors. Applications of MMV include direction of arrival (DOA) estimation and EEG/MEG source localization, among others. In our treatment of the MMV problem, all signal vectors are assumed to share the same support also referred to by having a common sparsity profile. The MMV model can be stated as:

$$\boldsymbol{y}^{(t)} = \boldsymbol{A}\boldsymbol{x}^{(t)} + \boldsymbol{e}^{(t)}, \;\; t = 1, 2, ..., T,$$

where we have T measurement vectors $[\boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)}..., \boldsymbol{y}^{(T)}]$ with $\boldsymbol{y}^{(t)} \in \mathbb{R}^M$. The objective is to recover $\boldsymbol{X} = [\boldsymbol{x}^{(t)}, \boldsymbol{x}^{(2)}..., \boldsymbol{x}^{(T)}]$ with $\boldsymbol{x}^{(t)} \in \mathbb{R}^N$, where in addition to the vectors $\boldsymbol{x}^{(t)}$ being sparse, they share the same sparsity profile. Similar to the SMV case, $\boldsymbol{A} \in \mathbb{R}^{M \times N}$ is known, and $[\boldsymbol{e}^{(1)}, \boldsymbol{e}^{(2)}..., \boldsymbol{e}^{(T)}]$ is a sequence of i.i.d. noise vectors modeled as $\boldsymbol{e}^{(t)} \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I})$.

It was shown in [57–60] that having multiple vectors can improve the recovery performance compared to the single measurement case. In practice it is often the case that the non-zero signal elements will experience temporal correlation, i.e., each non-zero row of the signal matrix can be treated as a correlated time series. If this correlation is not taken into consideration, the performance of MMV algorithms can degrade quickly [61].

A number of Bayesian algorithms were proposed to address the MMV problem. Among the most successful is a Bayesian algorithm known as the TMSBL algorithm [61–64], which models the sparsity and correlation of the signals using a modified Gaussian scale mixture prior. Other Bayesian algorithms addressed the high complexity issues associated with the TMSBL algorithm by using an AMP based model with an AR(1) process to model the temporal correlation [65, 66]. Although AMP algorithms did offer significant complexity improvements, TMSBL still achieved superior successful recovery rates over them. This dissertation in part explores how combining SBL assumptions with AMP based implementations can address the shortcomings of both approaches.

1.2 Dissertation Outline and Contributions

• In Chapter 2, we develop low-complexity algorithms for sparse Bayesian learning (SBL) [30, 31]. Since the traditional implementation of SBL uses matrix inversions at each iteration, its complexity is too high for large-scale problems. We circumvent the matrix inverse using the generalized approximate message passing (GAMP) algorithm [49,50,52]. Using GAMP to implement the E step of EM-based SBL provides a significant reduction in complexity over

the classical SBL algorithm. This work is a beneficiary of the algorithmic improvements and theoretical insights that have taken place in recent work in AMP [49, 50, 52], where we exploit the fact that using a Gaussian prior on $p(\boldsymbol{x})$ can provide guarantees for the GAMP E-step to not diverge when sufficient damping is used [52], even for a non-i.i.d.-Gaussian $\boldsymbol{A}$. In other words, the enhanced robustness of the proposed algorithm is due to the GSM prior used on $\boldsymbol{x}$, as opposed to other sparsity promoting priors for which there are no GAMP convergence guarantees when $\boldsymbol{A}$ is non-i.i.d.-Gaussian. The resulting algorithm is the Gaussian GAMP SBL (GGAMP-SBL) algorithm, which combines the robustness of SBL with the speed of GAMP.

• In chapter 2, to further illustrate and expose the synergy between the AMP and SBL frameworks, we also propose a new approach to the multiple measurement vector (MMV) problem with temporal correlation. Extensions of SBL to the MMV problem have been developed in [60, 61, 64], such as the TSBL and TMSBL algorithms [61]. Although TMSBL has lower complexity than TSBL, it still requires an order of $O(NM^2)$ operations per iteration, making it unsuitable for large-scale problems. To overcome the complexity problem, [65] and [67] proposed AMP-based Bayesian approaches to the MMV problem. However, similar to the SMV case, these algorithms are only expected to work for i.i.d zero-mean sub-Gaussian $\boldsymbol{A}$. We therefore extend the proposed GGAMP-SBL to the MMV case, to produce a GGAMP-TSBL algorithm that is more robust to generic $\boldsymbol{A}$, while achieving linear complexity in all problem dimensions.

• In chapter 3, we show that the previously shown advantages of derivation simplification and convergence improvement still apply in the non-negative constrained SSR problem when an rectified GSM (RGSM) prior is used. We show how the RGSM prior allows to change the effective prior on $\boldsymbol{x}$ by changing the mixing density, which translates into a simple change in the overall algorithm. We additionally show how extending the algorithm to address the multiple measurement vector (MMV) problem with common sparsity profile is also achieved by a simple algorithm change.

• The main contribution of chapter 4 is offering a novel interpretation of the SBL algorithm

based on the minimum power distortionless response (MPDR) beamforming method in array processing. The SBL's iterations are interpreted as the MPDR beamformer (BF) iteratively applied to the measurements and the beamformer parameters updated based on the estimates at each iteration, along with the ability to add a denoising step at the output of the MPDR BF when a prior is imposed on $\boldsymbol{x}$. Unlike the Bayesian formulation of the algorithm which was derived using an expectation maximization (EM) approach and is harder to extend, this new interpretation provides much needed intuitive support and significant clarity into how the SBL algorithm works and it allows for more flexibility in the algorithm's framework.

• In chapter 4, the benefits of the new insight is demonstrated by extending the algorithm to incorporate priors other than GSM priors, where in the past obtaining a solution with such priors was complex and could not be achieved in some cases, like the case of using an non-identical Laplace prior [39]. We also present another important example on potential benefits of the new interpretation by replacing the MPDR BF step with an approximate message passing (AMP) based MPDR implementation, which results in a low complexity Bayesian algorithm that can be used with a wide range of sparsity promoting priors. We refer to this methodology as the MPDR-SSR framework, and present the Laplace SBL (LSBL) algorithm that uses a non-identical Laplace prior as an example of the MPDR-SSR algorithm's capabilities.

• The first contribution of chapter 5 is to show how the new insights into the SBL algorithm extend into other variants of SBL. We show how the concept extends to the sequential fast-SBL algorithm, where we provide an interpretation of its iterations based on the minimum variance distortionless receiver (MVDR) BF [1]. We exploit the significance of this interpretation by showing how the algorithm can be extended to handle more general priors to incorporate any extra information about $\boldsymbol{x}$. We demonstrate how incorporating this information can provide performance enhancement to the fast-SBL by presenting a number of algorithms.

• The second contribution of chapter 5 is to show that the MMV problem can be addressed

[1]The terminology MPDR and MVDR are adopted from [68]. Though both are equivalent in the decorrelated source case, the problem formulation is different and this difference is relevant to our interpretation.

using the MPDR framework when the common support assumption is met. We show how the MPDR BF decouples the measurements in one dimension (space for arrays), making it much easier to deal with correlation across measurements one row at a time. Using the assumptions of TMSBL we show how the TMSBL can be derived using the MPDR framework, and based on this derivation we propose the low complexity MPDR-TSBL algorithm that uses the AMP version of the MPDR. We compare MPDR-TSBL to the previously proposed AMP based MMV algorithms. MPDR-TSBL shows superior performance compared to the two AMP based algorithms. MPDR-TSBL also provides an advantage in complexity over the other two algorithms when the number of measurements is large. Moreover, based on the convergence properties of the AMP based MPDR, the algorithm is expected to have better convergence compared the AMP-MMV when $\boldsymbol{A}$ is not i.i.d Gaussian, since the AMP based MPDR was shown to converge for general $\boldsymbol{A}$ matrices [52, 66].

• In chapter 6, we show how the Minimum Power Distortionless Response (MPDR) based decoupling principle, proposed in [69] to address the sparse signal recovery problem, can be applied to the E-step of the MIMO channel estimation problem. Based on this decoupling, we can impose the actual discrete prior on the data symbols and obtain a tractable approximate posterior that can be used to implement the E-step of the EM algorithm. This MPDR based approximation was shown to perform well in the SSR problem [69]. Similarly our results show that this approximation outperforms the Gaussian and heuristic EM approaches from [70] in low and high SNR scenarios.

• In chapter 6, we also address the dimensionality and complexity of the Gaussian prior EM approach. We show how by using a one time eigenvalue decomposition of the received data symbols, the dimensionality of the measurements and hence complexity is reduced to be independent of the number of data symbols. We then use the eigenvalue decomposition based Gaussian algorithm to initialize the MPDR based algorithm to significantly speed up the convergence of the MPDR based algorithm.

Chapter 2

A GAMP Based Low Complexity Sparse Bayesian Learning Algorithm

2.1 Introduction

The SBL algorithm and its MMV extensions have shown excellent performance when it comes to solving the SSR problem. However, because they need matrix inversions at each iteration, the complexity level of such algorithms makes them unsuitable for the use on large scale problems. In this chapter, we develop an algorithm based on the same assumptions used in the SBL algorithms, but use the low complexity GAMP algorithm to recover the sparse signal. Compared to the original EM-SBL algorithms, the proposed approach results in linear complexity, which makes it perfect for large scale problems. On the other hand, approximate message passing algorithms which can achieve low complexity, suffer from divergence issues when the measurement matrix in not i.i.d. Gaussian. In the special case that the prior and likelihood are both independent Gaussian, [52] was able to provide a full characterization of GAMP's convergence. In particular, it was shown that Gaussian GAMP (GGAMP) algorithm will converge if and only if the peak to average ratio of the squared singular values in $\boldsymbol{A}$ is sufficiently small. When this

condition is not met, [52] proposed a damping technique that guarantees convergence of GGAMP at the expense of slowing its convergence rate. Although the case of Gaussian prior and likelihood is not suitable to handle the sparse signal recovery problem directly, it is sufficient to replace the costly matrix inversions in some the algorithms we develop in this chapter.

2.1.1 Chapter's Organization

The organization of the chapter is as follows. In Section 2.2, we review the SBL algorithm. In Section 2.3, we combine the damped GGAMP algorithm with the SBL approach to solve the SMV problem. We introduce the GGAMP-SBL algorithm and investigate its convergence behavior. In Section 2.4, we use a time-correlated multiple measurement factor graph to derive the GGAMP-TSBL algorithm. In Section 2.5, we present numerical results to compare the performance and complexity of the proposed algorithms with the original SBL and with other AMP algorithms for the SMV case, and with TMSBL for the MMV case.

2.2 Sparse Bayesian Learning for SSR

2.2.1 GSM Class of Priors

We restate the sparse signal recovery problem here for chapter completeness. In the single measurement vector (SMV) case, the SSR problem consists of recovering a sparse signal $\boldsymbol{x} \in \mathbb{R}^N$ from $M \leq N$ noisy linear measurements $\boldsymbol{y} \in \mathbb{R}^M$:

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

where $\boldsymbol{A} \in \mathbb{R}^{M \times N}$ is a known measurement matrix and $\boldsymbol{e} \in \mathbb{R}^M$ is additive noise modeled by $\boldsymbol{e} \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I})$. We will assume that the entries of $\boldsymbol{x}$ are independent and identically distributed, i.e. $p(\boldsymbol{x}) = \Pi_n p(x_n)$. The sparsity promoting prior $p(x_n)$ will be chosen from the GSM class and

so will admit the following representation

$$p(x_n) = \int \mathcal{N}(x_n; 0, \gamma_n) p(\gamma_n) d\gamma_n, \tag{2.2}$$

where $\mathcal{N}(x_n; 0, \gamma_n)$ denotes a Gaussian density with mean zero and variance γ_n. The mixing density on hyperprior $p(\gamma_n)$ controls the prior on x_n. For instance, if a Laplacian prior is desired for x_n, then an exponential density is chosen for $p(\gamma_n)$ [44].

In the empirical Bayesian approach, an estimate of the hyperparameter vector $\boldsymbol{\gamma}$ is iteratively estimated, often using evidence maximization. For a given estimate $\hat{\boldsymbol{\gamma}}$, the posterior $p(\boldsymbol{x}|\boldsymbol{y})$ is approximated as $p(\boldsymbol{x}|\boldsymbol{y}; \hat{\boldsymbol{\gamma}})$, and the mean of this posterior is used as a point estimate for $\hat{\boldsymbol{x}}$. This mean can be computed in closed form, as detailed below, because the approximate posterior is Gaussian. It was shown in [39] that this empirical Bayesian method, also referred to as a Type II maximum likelihood method, can be used to formulate a number of algorithms for solving the SSR problem by changing the mixing density $p(\gamma_n)$. There are many computational methods that can be employed for computing $\boldsymbol{\gamma}$ in the evidence maximization framework, e.g, [30, 31, 71]. In this work, we utilize the EM-SBL algorithm because of its synergy with the GAMP framework, as will be apparent below.

2.2.2 SBL's EM Algorithm

In EM-SBL, the EM algorithm is used to learn the unknown signal variance vector $\boldsymbol{\gamma}$ [72–74], and possibly also the noise variance σ^2. We focus on learning $\boldsymbol{\gamma}$, assuming the noise variance σ^2 is known. We later state the EM update rule for the noise variance σ^2 for completeness.

The goal of the EM algorithm is to maximize the posterior $p(\boldsymbol{\gamma}|\boldsymbol{y})$ or equivalently[1] to minimize $-\log p(\boldsymbol{y}, \boldsymbol{\gamma})$. For the GSM prior (2.2) and the additive white Gaussian noise model,

[1] Using Bayes rule, $p(\boldsymbol{\gamma}|\boldsymbol{y}) = p(\boldsymbol{y}, \boldsymbol{\gamma})/p(\boldsymbol{y})$ where $p(\boldsymbol{y})$ is a constant with respect to $\boldsymbol{\gamma}$. Thus for MAP estimation of $\boldsymbol{\gamma}$ we can maximize $p(\boldsymbol{y}, \boldsymbol{\gamma})$, or minimize $-\log p(\boldsymbol{y}, \boldsymbol{\gamma})$.

this results in the SBL cost function [30, 31],

$$\begin{aligned}\chi(\boldsymbol{\gamma}) &= -\log p(\boldsymbol{y},\boldsymbol{\gamma}) \\ &= \frac{1}{2}\log|\Sigma_y| + \frac{1}{2}\boldsymbol{y}^\top \boldsymbol{\Sigma}_y^{-1}\boldsymbol{y} - \log p(\boldsymbol{\gamma}), \\ \boldsymbol{\Sigma}_{\boldsymbol{y}} &= \sigma^2\boldsymbol{I} + \boldsymbol{A}\boldsymbol{\Gamma}\boldsymbol{A}^\top, \;\; \boldsymbol{\Gamma} \triangleq \mathrm{Diag}(\boldsymbol{\gamma}).\end{aligned} \tag{2.3}$$

In the EM-SBL approach, $\boldsymbol{x}$ is treated as the hidden variable and the parameter estimate is iteratively updated as follows:

$$\gamma^{i+1} = \underset{\boldsymbol{\gamma}}{\mathrm{argmax}}\, \mathbb{E}_{\boldsymbol{x}|\boldsymbol{y};\boldsymbol{\gamma}^i}\left[\log p(\boldsymbol{y},\boldsymbol{x},\boldsymbol{\gamma})\right], \tag{2.4}$$

where $p(\boldsymbol{y},\boldsymbol{x},\boldsymbol{\gamma})$ is the joint probability of the complete data and $p(\boldsymbol{x}|\boldsymbol{y};\boldsymbol{\gamma}^i)$ is the posterior under the old parameter estimate $\boldsymbol{\gamma}^i$, which is used to evaluate the expectation. In each iteration, an expectation has to be computed (E-step) followed by a maximization step (M-step). It is easy to show that at each iteration, the EM algorithm increases a lower bound on the log posterior $\log p(\boldsymbol{\gamma}|\boldsymbol{y})$ [72], and it has been shown in [74] that the algorithm will converge to a stationary point of the posterior under a fairly general set of conditions that are applicable in many practical applications.

Next we detail the implementation of the E and M steps of the EM-SBL algorithm.

SBL's E-step

The Gaussian assumption on the additive noise $\boldsymbol{e}$ leads to the following Gaussian likelihood function:

$$p(\boldsymbol{y}|\boldsymbol{x};\sigma^2) = \frac{1}{(2\pi\sigma^2)^{\frac{M}{2}}}\exp\left(-\frac{1}{2\sigma^2}\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|^2\right). \tag{2.5}$$

Due to the GSM prior (2.2), the density of $\boldsymbol{x}$ conditioned on $\boldsymbol{\gamma}$ is Gaussian:

$$p(\boldsymbol{x}|\boldsymbol{\gamma}) = \prod_{n=1}^{N} \frac{1}{(2\pi\gamma_n)^{\frac{1}{2}}} \exp\left(-\frac{x_n^2}{2\gamma_n}\right). \tag{2.6}$$

Putting (2.5) and (2.6) together, the density needed for the E-step is Gaussian:

$$p(\boldsymbol{x}|\boldsymbol{y},\boldsymbol{\gamma}) = \mathcal{N}(\boldsymbol{x};\hat{\boldsymbol{x}},\boldsymbol{\Sigma}_{\boldsymbol{x}}) \tag{2.7}$$

$$\hat{\boldsymbol{x}} = \sigma^{-2}\boldsymbol{\Sigma}_{\boldsymbol{x}}\boldsymbol{A}^{\top}\boldsymbol{y} \tag{2.8}$$

$$\begin{aligned}\boldsymbol{\Sigma}_{\boldsymbol{x}} &= (\sigma^{-2}\boldsymbol{A}^{\top}\boldsymbol{A} + \boldsymbol{\Gamma}^{-1})^{-1} \\ &= \boldsymbol{\Gamma} - \boldsymbol{\Gamma}\boldsymbol{A}^{\top}(\sigma^2\boldsymbol{I} + \boldsymbol{A}\boldsymbol{\Gamma}\boldsymbol{A}^{\top})^{-1}\boldsymbol{A}\boldsymbol{\Gamma}.\end{aligned} \tag{2.9}$$

We refer to the mean vector as $\hat{\boldsymbol{x}}$ since it will be used as the SBL point estimate of $\boldsymbol{x}$. In the sequel, we will use $\boldsymbol{\tau}_x$ when referring to the vector composed from the diagonal entries of the covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{x}}$. Although both $\hat{\boldsymbol{x}}$ and $\boldsymbol{\tau}_x$ change with the iteration i, we will sometimes omit their i dependence for brevity. Note that the mean and covariance computations in (2.8) and (2.9) are not affected by the choice of $p(\boldsymbol{\gamma})$. The mean and diagonal entries of the covariance matrix are needed to carry out the M-step as shown next.

SBL's M-Step

The M-step is then carried out as follows. First notice that

$$\begin{aligned}&\mathbb{E}_{\boldsymbol{x}|\boldsymbol{y};\boldsymbol{\gamma}^i,\sigma^2}\left[-\log p(\boldsymbol{y},\boldsymbol{x},\boldsymbol{\gamma};\sigma^2)\right] = \\ &\mathbb{E}_{\boldsymbol{x}|\boldsymbol{y};\boldsymbol{\gamma}^i,\sigma^2}\left[-\log p(\boldsymbol{y}|\boldsymbol{x};\sigma^2) - \log p(\boldsymbol{x}|\boldsymbol{\gamma}) - \log p(\boldsymbol{\gamma})\right].\end{aligned} \tag{2.10}$$

Since the first term in (2.10) does not depend on $\boldsymbol{\gamma}$, it will not be relevant for the M-step and thus can be ignored. Similarly, in the subsequent steps we will drop constants and terms that do not

depend on $\boldsymbol{\gamma}$ and therefore do not impact the M-step. Since $\mathbb{E}_{\boldsymbol{x}|\boldsymbol{y};\boldsymbol{\gamma}^i,\sigma^2}[x_n^2] = \hat{x}_n^2 + \tau_{x_n}$,

$$\mathbb{E}_{\boldsymbol{x}|\boldsymbol{y};\boldsymbol{\gamma}^i,\sigma^2}[-\log p(\boldsymbol{x}|\boldsymbol{\gamma}) - \log p(\boldsymbol{\gamma})] = \sum_{n=1}^{N}\left(\left(\frac{\hat{x}_n^2 + \tau_{x_n}}{2\gamma_n}\right) + \frac{1}{2}\log\gamma_n - \log p(\gamma_n)\right). \tag{2.11}$$

Note that the E-step only requires $\hat{x}_n$, the posterior mean from (2.8), and τ_{x_n},the posterior variance from (2.9), which are statistics of the marginal densities $p(x_n|\boldsymbol{y},\boldsymbol{\gamma}^i)$. In other words, the full joint posterior $p(\boldsymbol{x}|\boldsymbol{y},\boldsymbol{\gamma}^i)$ is not needed. This facilitates the use of message passing algorithms.

As can be seen from (2.7)-(2.9), the computation of $\hat{\boldsymbol{x}}$ and $\boldsymbol{\tau}_x$ involves the inversion of an $N \times N$ matrix, which can be reduced to $M \times M$ matrix inversion by the matrix inversion lemma. The complexity of computing $\hat{\boldsymbol{x}}$ and $\boldsymbol{\tau}_x$ can be shown to be $O(NM^2)$ under the assumption that $M \leq N$. This makes the EM-SBL algorithm computationally prohibitive and impractical to use with large dimensions.

From (2.4) and (2.11), the M-step for each iteration is as follows:

$$\boldsymbol{\gamma}^{i+1} = \underset{\boldsymbol{\gamma}}{\operatorname{argmin}}\left[\sum_{n=1}^{N}\left(\frac{\hat{x}_n^2 + \tau_{x_n}}{2\gamma_n} + \frac{\log\gamma_n}{2} - \log p(\gamma_n)\right)\right]. \tag{2.12a}$$

This reduces to N scalar optimization problems,

$$\gamma_n^{i+1} = \underset{\gamma_n}{\operatorname{argmin}}\left[\frac{\hat{x}_n^2 + \tau_{x_n}}{2\gamma_n} + \frac{1}{2}\log\gamma_n - \log p(\gamma_n)\right]. \tag{2.12b}$$

The choice of hyperprior $p(\boldsymbol{\gamma})$ plays a role in the M-step, and governs the prior for $\boldsymbol{x}$. However, from the computational simplicity of the M-step, as evident from (2.12b), the hyperprior rarely impacts the overall algorithmic computational complexity, which is mainly that of computing the quantities $\hat{\boldsymbol{x}}$ and $\boldsymbol{\tau}_x$ in the E-step.

Often a non-informative prior is used in SBL. For the purpose of obtaining the M-step update, we will also simplify and drop $p(\boldsymbol{\gamma})$ and compute the Maximum Likelihood estimate of $\boldsymbol{\gamma}$. From (2.12b), this reduces to, $\gamma_n^{i+1} = \hat{x}_n^2 + \tau_{x_n}$.

Similarly, if the noise variance σ^2 is unknown, it can be estimated using:

$$\begin{aligned}(\sigma^2)^{i+1} &= \underset{\sigma^2}{\operatorname{argmax}}\, \mathbb{E}_{\boldsymbol{x}|\boldsymbol{y},\boldsymbol{\gamma};(\sigma^2)^i}[p(\boldsymbol{y},\boldsymbol{x},\boldsymbol{\gamma};\sigma^2)] \\ &= \frac{\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|^2 + (\sigma^2)^i \sum_{n=1}^{N}\left(1-\frac{\tau_{x_n}}{\gamma_n}\right)}{M}. \end{aligned} \tag{2.13}$$

We note here that estimates obtained by (2.13) can be highly inaccurate as mentioned in [60]. Therefore, it suggests that experimenting with different values of σ^2 or using some other application based heuristic will probably lead to better results.

2.3 Damped Gaussian GAMP SBL

We now show how damped GGAMP can be used to simplify the E-step above, leading to the damped GGAMP-SBL algorithm. Then we examine the convergence behavior of the resulting algorithm.

2.3.1 GGAMP-SBL

Above we showed that, in the EM-SBL algorithm, the M-step is computationally simple but the E-step is computationally demanding. The GAMP algorithm can be used to efficiently approximate the quantities $\hat{\boldsymbol{x}}$ and $\boldsymbol{\tau}_x$ needed in the E-step, while the M-step remains unchanged. GAMP is based on the factor graph in Figure 2.1, where for a given prior $f_n(x) = p(x_n)$ and a likelihood function $g_m = p(y_m|\boldsymbol{x})$, GAMP uses quadratic approximations and Taylor series expansions, to provide approximations of MAP or MMSE estimates of $\boldsymbol{x}$. The reader can refer to [50] for detailed derivation of GAMP. The E-step in Table 2.1, uses the damped GGAMP algorithm from [52] because of its ability to enhance traditional GAMP algorithm divergence issues with non-i.i.d.-Gaussian $\boldsymbol{A}$. The damped GGAMP algorithm has an important modification over the original GAMP algorithm and also over the previously proposed AMP-SBL [67], namely

the introduction of damping factors $\theta_s, \theta_x \in (0,1]$ to slow down updates and enhance convergence. Setting $\theta_s = \theta_x = 1$ in the damped GGAMP algorithm will yield no damping, and reduces the algorithm to the original GAMP algorithm. We note here that the damped GGAMP algorithm from [52] is referred to by GGAMP, and therefore we will be using the terms GGAMP and damped GGAMP interchangeably in this chapter. Moreover, when the components of the matrix $\boldsymbol{A}$ are not zero-mean, one can incorporate the same mean removal technique used in [54]. The input and output functions $g_s(\boldsymbol{p}, \boldsymbol{\tau}_p)$ and $g_x(\boldsymbol{r}, \boldsymbol{\tau}_r)$ in Table 2.1 are defined based on whether the max-sum or the sum-product version of GAMP is being used. The intermediate variables $\boldsymbol{r}$ and $\boldsymbol{p}$ are interpreted as approximations of Gaussian noise corrupted versions of $\boldsymbol{x}$ and $\boldsymbol{z} = \boldsymbol{A}\boldsymbol{x}$, with the respective noise levels of $\boldsymbol{\tau}_r$ and $\boldsymbol{\tau}_p$. In the max-sum version, the vector MAP estimation problem is reduced to a sequence of scalar MAP estimates given $\boldsymbol{r}$ and $\boldsymbol{p}$ using the input and output functions, where they are defined as:

$$[g_s(\boldsymbol{p}, \boldsymbol{\tau}_p)]_m = p_m - \tau_{p_m} \text{prox}_{\frac{-1}{\tau_{p_m}} \ln p(y_m|z_m)}(\frac{p_m}{\tau_{p_m}}) \tag{2.14}$$

$$[g_x(\boldsymbol{r}, \boldsymbol{\tau}_r)]_n = \text{prox}_{-\tau_{r_n} \ln p(x_n)}(r_n) \tag{2.15}$$

$$\text{prox}_f(r) \triangleq \underset{x}{\text{argmin}}\, f(x) + \frac{1}{2}|x - r|^2. \tag{2.16}$$

Similarly, in the sum-product version of the algorithm, the vector MMSE estimation problem is reduced to a sequence of scalar MMSE estimates given $\boldsymbol{r}$ and $\boldsymbol{p}$ using the input and output functions, where they are defined as:

$$[g_s(\boldsymbol{p}, \boldsymbol{\tau}_p)]_m = \frac{\int z_m p(y_m|z_m) \mathcal{N}(z_m; \frac{p_m}{\tau_{p_m}}, \frac{1}{\tau_{p_m}}) dz_m}{\int p(y_m|z_m) \mathcal{N}(z_m; \frac{p_m}{\tau_{p_m}}, \frac{1}{\tau_{p_m}}) dz_m} \tag{2.17}$$

$$[g_x(\boldsymbol{r}, \boldsymbol{\tau}_r)]_n = \frac{\int x_n p(x_n) \mathcal{N}(x_n; r_n, \tau_{r_n}) dx_n}{\int p(x_n) \mathcal{N}(x_n; r_n, \tau_{r_n}) dx_n}. \tag{2.18}$$

For the parametrized Gaussian prior we imposed on $\boldsymbol{x}$ in (2.2), both sum-product and

max-sum versions of $g_x(\boldsymbol{r},\boldsymbol{\tau}_r)$ yield the same updates for $\hat{\boldsymbol{x}}$ and $\boldsymbol{\tau}_x$ [50, 52]:

$$g_x(\boldsymbol{r},\boldsymbol{\tau}_r) = \frac{\boldsymbol{\gamma}}{\boldsymbol{\gamma}+\boldsymbol{\tau}_r}\boldsymbol{r} \tag{2.19}$$

$$g'_x(\boldsymbol{r},\boldsymbol{\tau}_r) = \frac{\boldsymbol{\gamma}}{\boldsymbol{\gamma}+\boldsymbol{\tau}_r}. \tag{2.20}$$

Similarly, in the case of the likelihood $p(\boldsymbol{y}|\boldsymbol{x})$ given in (2.5), the max-sum and sum-product versions of $g_s(\boldsymbol{p},\boldsymbol{\tau}_p)$ yield the same updates for $\boldsymbol{s}$ and $\boldsymbol{\tau}_s$ [50, 52]:

$$g_s(\boldsymbol{p},\boldsymbol{\tau}_p) = \frac{(\boldsymbol{p}/\boldsymbol{\tau}_p - \boldsymbol{y})}{(\sigma^2 + 1/\boldsymbol{\tau}_p)} \tag{2.21}$$

$$g'_s(\boldsymbol{p},\boldsymbol{\tau}_p) = \frac{\sigma^{-2}}{\sigma^{-2}+\boldsymbol{\tau}_p}. \tag{2.22}$$

We note that, in equations (2.19),(2.20),(2.21) and (2.22), and for all equations in Table 2.1, all vector squares, divisions and multiplications are taken element wise.

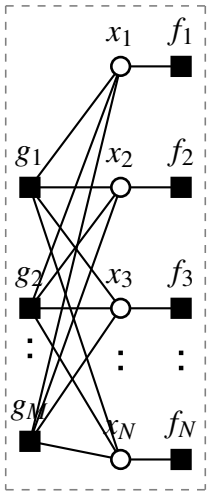


Figure 2.1: GAMP Factor Graph

In Table 2.1, $K_{\max}$ is the maximum allowed number of GAMP algorithm iterations, $\varepsilon_{\text{gamp}}$ is the GAMP normalized tolerance parameter, $I_{\max}$ is the maximum allowed number of EM iterations and ε_{em} is the EM normalized tolerance parameter. Upon the convergence of GAMP algorithm based E-step, estimates for the mean $\hat{\boldsymbol{x}}$ and covariance diagonal $\boldsymbol{\tau}_x$ are obtained. These

Table 2.1: GGAMP-SBL algorithm

Initialization	
$\boldsymbol{S} \leftarrow \lvert\boldsymbol{A}\rvert^2$ (component wise magnitude squared)	(I1)
Initialize $\breve{\boldsymbol{\tau}}_x^0, \boldsymbol{\gamma}^0, (\sigma^2)^0 > 0$	(I2)
$\breve{\boldsymbol{s}}^0, \breve{\boldsymbol{x}}^0 \leftarrow \boldsymbol{0}$	(I3)
for $i = 1, 2, ..., I_{\max}$	
Initialize $\boldsymbol{\tau}_x^1 \leftarrow \breve{\boldsymbol{\tau}}_x^{i-1}, \hat{\boldsymbol{x}}^1 \leftarrow \breve{\boldsymbol{x}}^{i-1}, \boldsymbol{s}^1 \leftarrow \breve{\boldsymbol{s}}^{i-1}$	
E-Step approximation	
for $k = 1, 2, ..., K_{\max}$	
$1/\boldsymbol{\tau}_p^k \leftarrow \boldsymbol{S}\boldsymbol{\tau}_x^k$	(A1)
$\boldsymbol{p}^k \leftarrow \boldsymbol{s}^{k-1} + \boldsymbol{\tau}_p^k \boldsymbol{A}\hat{\boldsymbol{x}}^k$	(A2)
$\boldsymbol{\tau}_s^k \leftarrow \boldsymbol{\tau}_p^k g_s'(\boldsymbol{p}^k, \boldsymbol{\tau}_p^k)$	(A3)
$\boldsymbol{s}^k \leftarrow (1-\theta_s)\boldsymbol{s}^{k-1} + \theta_s g_s(\boldsymbol{p}^k, \boldsymbol{\tau}_p^k)$	(A4)
$1/\boldsymbol{\tau}_r^k \leftarrow \boldsymbol{S}^\top \boldsymbol{\tau}_s^k$	(A5)
$\boldsymbol{r}^k \leftarrow \hat{\boldsymbol{x}}^k - \boldsymbol{\tau}_r^k \boldsymbol{A}^\top \boldsymbol{s}^k$	(A6)
$\boldsymbol{\tau}_x^{k+1} \leftarrow \boldsymbol{\tau}_r^k g_x'(\boldsymbol{r}^k, \boldsymbol{\tau}_r^k)$	(A7)
$\hat{\boldsymbol{x}}^{k+1} \leftarrow (1-\theta_x)\hat{\boldsymbol{x}}^k + \theta_x g_x(\boldsymbol{r}^k, \boldsymbol{\tau}_r^k)$	(A8)
if $\lVert\hat{\boldsymbol{x}}^{k+1} - \hat{\boldsymbol{x}}^k\rVert^2 / \lVert\hat{\boldsymbol{x}}^{k+1}\rVert^2 < \varepsilon_{\text{gamp}}$, break	(A9)
end for %end of k loop	
$\breve{\boldsymbol{s}}^i \leftarrow \boldsymbol{s}^k$, $\breve{\boldsymbol{x}}^i \leftarrow \hat{\boldsymbol{x}}^{k+1}$, $\breve{\boldsymbol{\tau}}_x^i \leftarrow \boldsymbol{\tau}_x^{k+1}$	
M-Step	
$\boldsymbol{\gamma}^{i+1} \leftarrow \lvert\breve{\boldsymbol{x}}^i\rvert^2 + \breve{\boldsymbol{\tau}}_x^i$	(M1)
$(\sigma^2)^{i+1} \leftarrow \frac{\lVert\boldsymbol{y} - \boldsymbol{A}\breve{\boldsymbol{x}}^i\rVert^2 + (\sigma^2)^i \sum_{n=1}^N \left(1 - \frac{\breve{\tau}_{x_n}^i}{\gamma_n^i}\right)}{M}$	(M2)
if $\lVert\breve{\boldsymbol{x}}^i - \breve{\boldsymbol{x}}^{i-1}\rVert^2 / \lVert\breve{\boldsymbol{x}}^i\rVert^2 < \varepsilon_{\text{em}}$, break	(M3)
end for %end of i loop	

estimates can be used in the M-step of the algorithm, given by equation (2.12b). These estimates, along with the $\boldsymbol{s}$ vector estimate, are also used to initialize the E-step at the next EM iteration to accelerate the convergence of the overall algorithm.

Defining $\boldsymbol{S}$ as the component wise magnitude squared of $\boldsymbol{A}$, the complexity of the GGAMP-SBL algorithm is dominated by the E-step, which in turn (from Table 2.1) is dominated by the matrix multiplications by $\boldsymbol{A}$, $\boldsymbol{A}^\top$, $\boldsymbol{S}$ and $\boldsymbol{S}^\top$ at each iteration, implying that the computational cost of the algorithm is $O(NM)$ operations per GAMP algorithm iteration multiplied by the total number of GAMP algorithm iterations. For large M, this is much smaller than $O(NM^2)$, the

complexity of standard SBL iteration.

In addition to the complexity of each iteration, for the proposed GGAMP-SBL algorithm to achieve faster runtimes it is important for GGAMP-SBL total number of iterations to not be too large, to the point where it over weighs the reduction in complexity per iteration, especially when heavier damping is used. We point out here that while SBL provides a one step exact solution for the E-step, GGAMP-SBL provides an approximate iterative solution. Based on that, the total number of SBL iterations is the number of EM iterations needed for convergence, while the total number of GGAMP-SBL iterations is based on the number of EM iterations it needs to converge and the number of E-step iterations for each EM iteration. First we consider the number of EM iterations for both algorithms. As explained in Section 2.3.2, the E-step of GGAMP-SBL algorithm provides a good approximation of the true posterior [75]. In addition to that the number of EM iterations is not affected by damping, since damping only affects the number of iterations of GGAMP in the E-step, but it does not affect its outcome upon convergence. Based on these two points, we can expect the number of EM iterations for GGAMP-SBL to be generally in the same range as the original SBL algorithm. This is also shown in Section 2.3.2 Figs. 2.2a and 2.2b, where we can see the two cost functions being reduced to their minimum values using approximately the same number of EM iterations, even when heavier damping is used. As for the GGAMP-SBL E-step iterations, because we are warm starting each E-step with $\boldsymbol{x}$ and $\boldsymbol{s}$ values from the previous EM iteration, it was found through numerical experiments that the number of required E-step iterations is reduced each time, to the point where the E-step converges to the required tolerance within 2-3 iterations towards the final EM iterations. When averaging the total number of E-step iterations over the number of EM iterations, it was found that for medium to large problem sizes the average number of E-step iterations was just a fraction of the measurements number M, even in the cases where heavier damping was used. Moreover, it was observed that the number of iterations required for the E-step to converge is independent of the problem size, which gives the algorithm a bigger advantage at larger problem sizes. Finally,

based on the complexity reduction per iteration and the total number of iterations required for GGAMP-SBL, we can expect it to have lower runtimes than SBL for medium to large problems, even when heavier damping is used. This runtime advantage is confirmed through numerical experiments in Section 2.5.

2.3.2 GGAMP-SBL Convergence

We now examine the convergence of the GGAMP-SBL algorithm. This involves two steps; the first step is to show that the approximate message passing algorithm employed in the E-step converges and the second step is to show that the overall EM algorithm, which consists of several E and M-steps, converges. For the second step, in addition to convergence of the E-step (first step), the accuracy of the resulting estimates is important. Therefore, in the second step of our convergence investigation, we use results from [75], in addition to numerical results to show that the GGAMP-SBL's E and M steps are actually descending on the original SBL's cost function (2.3) at each EM iteration.

Convergence of the E-step with Generic Transformations

For the first step, we use the analysis from [52] which shows that, in the case of generic $\boldsymbol{A}$, the damped GGAMP algorithm is guaranteed to globally converge (to some values $\hat{\boldsymbol{x}}$ and $\boldsymbol{\tau}_x$) when sufficient damping is used. In particular, since $\boldsymbol{\gamma}^i$ is fixed in the E-step, the prior is Gaussian and so based on results in [52], starting with an initial estimate $\boldsymbol{\tau}_x \geq \boldsymbol{\gamma}^i$ the variance updates $\boldsymbol{\tau}_x$, $\boldsymbol{\tau}_s$, $\boldsymbol{\tau}_r$ and $\boldsymbol{\tau}_p$ will converge to a unique fixed point. In addition, any fixed point $(\boldsymbol{s}, \hat{\boldsymbol{x}})$ for GGAMP is globally stable if $\theta_s \theta_x ||\tilde{\boldsymbol{A}}||_2^2 < 1$, where the matrix $\tilde{\boldsymbol{A}}$ is defined as given below and is based on the

fixed-point values of $\boldsymbol{\tau}_p$ and $\boldsymbol{\tau}_r$:

$$\tilde{\boldsymbol{A}} := \text{Diag}^{1/2}(\boldsymbol{\tau}_p \boldsymbol{q}_s)\boldsymbol{A}\ \text{Diag}^{1/2}(\boldsymbol{\tau}_r \boldsymbol{q}_x)$$
$$\boldsymbol{q}_s = \frac{\sigma^{-2}}{\sigma^{-2}+\boldsymbol{\tau}_p},\ \boldsymbol{q}_x = \frac{\boldsymbol{\gamma}}{\boldsymbol{\gamma}+\boldsymbol{\tau}_r}.$$

While the result above establishes that the GGAMP algorithm is guaranteed to converge when sufficient amount of damping is used at each iteration, in practice we do not recommend building the matrix $\tilde{\boldsymbol{A}}$ at each EM iteration and calculating its spectral norm. Rather, we recommend choosing sufficiently small damping factors θ_x and θ_s and fixing them for all GGAMP-SBL iterations. For this purpose, the following result from [52] for an i.i.d.-Gaussian prior $p(\boldsymbol{x}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{0}, \gamma_x \boldsymbol{I})$ can provide some guidance on choosing the damping factors. For the i.i.d.-Gaussian prior case, the damped GAMP algorithm is shown to converge if

$$\Omega(\theta_s, \theta_x) > \|\boldsymbol{A}\|_2^2 / \|\boldsymbol{A}\|_F^2, \tag{2.23}$$

where $\Omega(\theta_s, \theta_x)$ is defined as

$$\Omega(\theta_s, \theta_x) := \frac{2[(2-\theta_x)N + \theta_x M]}{\theta_x \theta_s MN}. \tag{2.24}$$

Experimentally, it was found that using a threshold $\Omega(\theta_s, \theta_x)$ that is 10% larger than (2.24) is sufficient for the GGAMP-SBL algorithm to converge in the scenarios we considered.

GGAMP-SBL Convergence

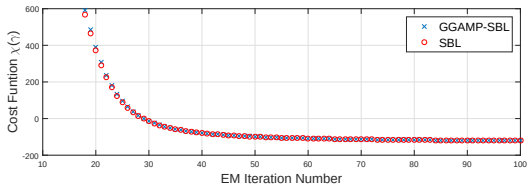
The result above guarantees convergence of the E-step to some vectors $\hat{\boldsymbol{x}}$ and $\boldsymbol{\tau}_x$ but it does not provide information about the overall convergence of the EM algorithm to the desired SBL fixed points. This convergence depends on the quality of the mean $\hat{\boldsymbol{x}}$ and variance $\boldsymbol{\tau}_x$ computed by the GGAMP algorithm. It has been shown that for an arbitrary $\boldsymbol{A}$ matrix, the fixed-point value

of $\hat{\boldsymbol{x}}$ will equal the true mean given in (2.8) [75]. As for the variance updates, based on the state evolution in [51], the vector $\boldsymbol{\tau}_x$ will equal the true posterior variance vector, i.e., the diagonal of (2.9), in the case that $\boldsymbol{A}$ is large and i.i.d. Gaussian, but otherwise will only approximate the true quantity.

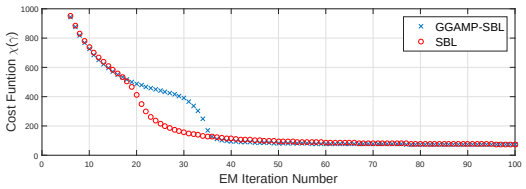
The approximation of $\boldsymbol{\tau}_x$ by the GGAMP algorithm in the E-step introduces an approximation in the GGAMP-SBL algorithm compared to the original EM-SBL algorithm. Fortunately, there is some flexibility in the EM algorithm in that the M-step need not be carried out to minimize the objective stated in (2.12a) but it is sufficient to decrease the objective function as discussed in the generalized EM algorithm literature [73, 74]. Given that the mean is estimated accurately, EM iterations may be tolerant to some error in the variance estimate. Some flexibility in this regards can also be gleaned from the results in [38], where it is shown how different iteratively reweighted algorithms correspond to a different choice in the variance. However, we have not been able to prove rigorously that the GGAMP approximation will guarantee descent of the original cost function given in (2.3).

Nevertheless, our numerical experiments suggest that the GGAMP approximation has negligible effect on algorithm convergence and ability to recover sparse solutions. We select two experiments to illustrate the convergence behavior and demonstrate that the approximate variance estimates are sufficient to decrease SBL's cost function (2.3). In both experiments $\boldsymbol{x}$ is drawn from a Bernoulli-Gaussian distribution with a non-zero probability λ set to 0.2, and we set $N = 1000$ and $M = 500$. Fig. 2.2 shows a comparison between the original SBL and the GGAMP-SBL's cost functions at each EM iteration of the algorithms. $\boldsymbol{A}$ in Fig. 2.2a is i.i.d.-Gaussian, while in Fig. 2.2b it is a column correlated transformation matrix, which is constructed according to the description given in Section 2.5, with correlation coefficient $\rho = 0.9$.

The cost functions in Fig. 2.2a and Fig. 2.2b show that, although we are using an approximate variance estimate to implement the M-step, the updates are decreasing the SBL's cost function at each iteration. As noted previously, it is not necessary for the M-step to provide the



(a) Cost functions for i.i.d.-Gaussian A



(b) Cost functions for column correlated A

Figure 2.2: Cost functions on SBL and GGAMP-SBL algorithms versus number of EM iterations

maximum cost function reduction, it is sufficient to provide some reduction in the cost function for the EM algorithm to be effective. The cost function plots confirm this principle, since GGAMP-SBL eventually reaches the same minimal value as the original EM-SBL. While the two numerical experiments do not provide a guarantee that the overall GGAMP-SBL algorithm will converge, they suggest that the performance of the GGAMP-SBL algorithm often matches that of the original EM-SBL algorithm, which is supported by the more extensive numerical results in Section 2.5.

2.4 GGAMP-TSBL for the MMV problem

In this section, we apply the damped GAMP algorithm to the MMV empirical Bayesian approach to derive a low complexity algorithm for the MMV case as well. Since the GAMP algorithm was originally derived for the SMV case using an SMV factor graph [50], extending it to the MMV case requires some more effort and requires going back to the factor graphs that are the basis of the GAMP algorithm, making some adjustments, and then utilizing the GAMP algorithm.

Once again we use an empirical Bayesian approach with a GSM model, and we focus on the ML estimate of $\boldsymbol{\gamma}$. We assume a common sparsity profile between all measured vectors, and also account for the temporal correlation that might exist between the non-zero signal elements. Previous Bayesian algorithms that have shown good recovery performance for the MMV problem

include extensions of the SMV SBL algorithm, such as MSBL [60], TSBL and TMSBL [61]. MSBL is a straightforward extension of SMV SBL, where no temporal correlation between non-zero elements is assumed, while TSBL and TMSBL account for temporal correlation. Even though the TMSBL algorithm has lower complexity compared to the TSBL algorithm, the algorithm still has complexity of $O(NM^2)$, which can limit its utility when the problem dimensions are large. Other AMP based Bayesian algorithms have achieved linear complexity in the problem dimensions, like AMP-MMV [65]. However AMP-MMV's robustness to generic $\boldsymbol{A}$ matrices is expected to be outperformed by an SBL based approach.

2.4.1 MMV Model and Factor Graph

The MMV model can be stated as:

$$\boldsymbol{y}^{(t)} = \boldsymbol{A}\boldsymbol{x}^{(t)} + \boldsymbol{e}^{(t)}, \;\; t = 1,2,...,T,$$

where we have T measurement vectors $[\boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)}..., \boldsymbol{y}^{(T)}]$ with $\boldsymbol{y}^{(t)} \in \mathbb{R}^M$. The objective is to recover $\boldsymbol{X} = [\boldsymbol{x}^{(t)}, \boldsymbol{x}^{(2)}..., \boldsymbol{x}^{(T)}]$ with $\boldsymbol{x}^{(t)} \in \mathbb{R}^N$, where in addition to the vectors $\boldsymbol{x}^{(t)}$ being sparse, they share the same sparsity profile. Similar to the SMV case, $\boldsymbol{A} \in \mathbb{R}^{M \times N}$ is known, and $[\boldsymbol{e}^{(1)}, \boldsymbol{e}^{(2)}..., \boldsymbol{e}^{(T)}]$ is a sequence of i.i.d. noise vectors modeled as $\boldsymbol{e}^{(t)} \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I})$. This model can be restated as:

$$\bar{\boldsymbol{y}} = \boldsymbol{D}(\boldsymbol{A})\bar{\boldsymbol{x}} + \bar{\boldsymbol{e}},$$

where $\bar{\boldsymbol{y}} \triangleq [\boldsymbol{y}^{(1)^\top}, \boldsymbol{y}^{(2)^\top}..., \boldsymbol{y}^{(T)^\top}]^\top$, $\bar{\boldsymbol{x}} \triangleq [\boldsymbol{x}^{(1)^\top}, \boldsymbol{x}^{(2)^\top}..., \boldsymbol{x}^{(T)^\top}]^\top$, $\bar{\boldsymbol{e}} \triangleq [\boldsymbol{e}^{(1)^\top}, \boldsymbol{e}^{(2)^\top}..., \boldsymbol{e}^{(T)^\top}]^\top$ and $\boldsymbol{D}(\boldsymbol{A})$ is a block-diagonal matrix constructed from T replicas of $\boldsymbol{A}$.

The posterior distribution of $\bar{\boldsymbol{x}}$ is given by:

$$p(\bar{\boldsymbol{x}}\,|\,\bar{\boldsymbol{y}}) \propto \prod_{t=1}^{T}\left[\prod_{m=1}^{M} p(y_m^{(t)}\,|\,\boldsymbol{x}^{(t)}) \prod_{n=1}^{N} p(x_n^{(t)}\,|\,x_n^{(t-1)})\right],$$

where

$$p(y_m^{(t)}\,|\,\boldsymbol{x}^{(t)}) = \mathcal{N}(y_m^{(t)}; \boldsymbol{a}_{m.}^{\top}\boldsymbol{x}^{(t)}, \sigma^2),$$

where $\boldsymbol{a}_{m.}^{\top}$ is the m^{th} row of the matrix $\boldsymbol{A}$. Similar to the previous work in [64, 65, 76], we use an AR(1) process to model the correlation between $x_n^{(t)}$ and $x_n^{(t-1)}$, i.e.,

$$\begin{aligned} x_n^{(t)} &= \beta x_n^{(t-1)} + \sqrt{1-\beta^2}\, v_n^{(t)} \\ p(x_n^{(t)}\,|\,x_n^{(t-1)}) &= \mathcal{N}(x_n^{(t)}; \beta x_n^{(t-1)}, (1-\beta^2)\gamma_n),\ t > 1 \\ p(x_n^{(1)}) &= \mathcal{N}(x_n^{(1)}; 0, \gamma_n), \end{aligned}$$

where $\beta \in (-1, 1)$ is the temporal correlation coefficient and $v_n^{(t)} \sim \mathcal{N}(0, \gamma_n)$. Following an empirical Bayesian approach similar to the one proposed for the SMV case, the hyperparameter vector $\boldsymbol{\gamma}$ is then learned from the measurements using the EM algorithm. The EM algorithm can also be used to learn the correlation coefficient β and the noise variance σ^2. Based on these assumptions we use the sum-product algorithm [77] to construct the factor graph in Fig. 2.3, and derive the MMV algorithm GGAMP-TSBL. In the MMV factor graph, the factors are $g_m^{(t)}(\boldsymbol{x}) = p(y_m^{(t)}\,|\,\boldsymbol{x}^{(t)})$, $f_n^{(t)}(x_n^{(t)}) = p(x_n^{(t)}\,|\,x_n^{(t-1)})$ for $t > 1$ and $f_n^{(1)}(x_n^{(1)}) = p(x_n^{(1)})$.

2.4.2 GGAMP-TSBL Message Phases and Scheduling (E-Step)

Due to the similarities between the factor graph for each time frame of the MMV model and the factor graph of the SMV model, we will use the algorithm in Table 2.1 as a building block and extend it to the MMV case. We divide the message updates into three steps as shown in

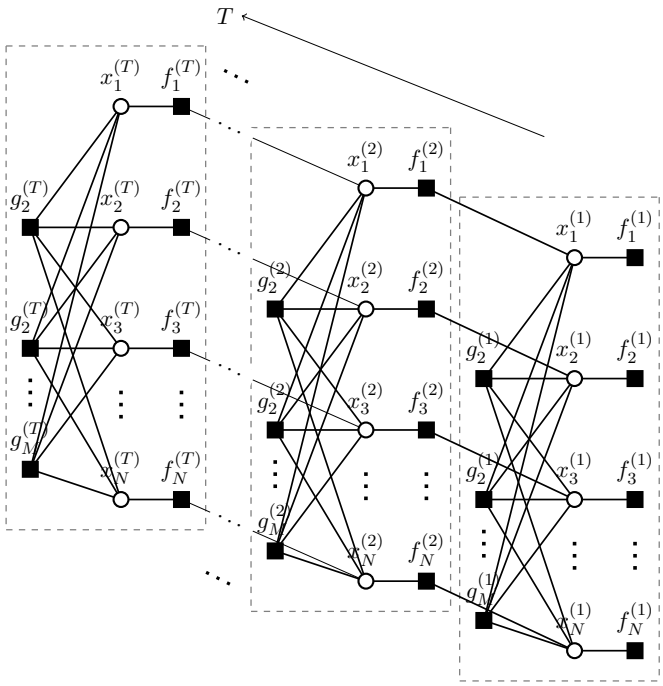


Figure 2.3: GGAMP-TSBL factor graph

Fig. 2.4.

For each time frame the "within" step in Fig. 2.4 is very similar to the SMV GAMP iteration, with the only difference being that each $x_n^{(t)}$ is connected to the factor nodes $f_n^{(t)}$ and $f_n^{(t+1)}$, while it is connected to one factor node in the SMV case. This difference is reflected in the calculation of the output function g_x and therefore in finding the mean and variance estimates for $\bar{\boldsymbol{x}}$. The details of finding g_x and therefore the update equations for $\boldsymbol{\tau}_x^{(t)}$ and $\hat{\boldsymbol{x}}^{(t)}$ are shown in Appendix A. The input function g_s is the same as (2.21), and the update equations for $\boldsymbol{\tau}_s^{(t)}$ and $\boldsymbol{s}^{(t)}$ are the same as (A3) and (A4) from Table 2.1, because an AWGN model is assumed for the noise. The second type of updates are passing messages forward in time from $x_n^{(t-1)}$ to $x_n^{(t)}$ through $f_n^{(t)}$. And the final type of updates is passing messages backward in time from $x_n^{(t+1)}$ to $x_n^{(t)}$ through $f_n^{(t)}$. The details for finding the "forward" and "backward" message passing steps are also shown in Appendix A.

We schedule the messages by moving forward in time first, where we run the "forward" step starting at $t = 1$ all the way to $t = T$. We then perform the "within" step for all time frames,

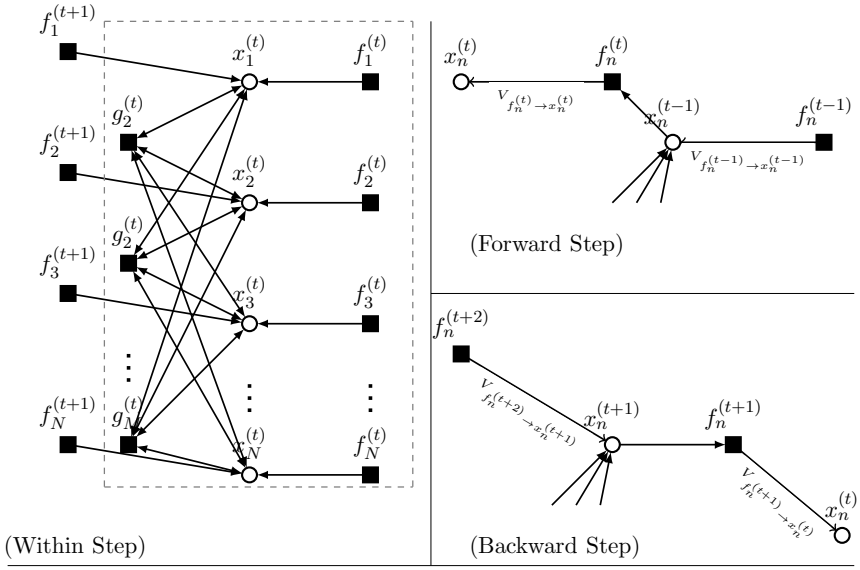


Figure 2.4: Message passing phases for GGAMP-TSBL

this step updates $\boldsymbol{r}^{(t)}$,$\boldsymbol{\tau}_r^{(t)}$, $\boldsymbol{x}^{(t)}$ and $\boldsymbol{\tau}_x^{(t)}$ that are needed for the "forward" and "backward" message passing steps. Finally we pass the messages backward in time using the "backward" step, starting at $t = T$ and ending at $t = 1$. Based on this message schedule, the GAMP algorithm E-step computation is summarized in Table 2.2. In Table 2.2 we use the unparenthesized superscript to indicate the iteration index, while the parenthesized superscript indicates the time frame index. Similar to Table 2.1, $K_{\max}$ is the maximum allowed number of GAMP iterations, $\varepsilon_{\text{gamp}}$ is the GAMP normalized tolerance parameter, $I_{\max}$ is the maximum allowed number of EM iterations and ε_{em} is the EM normalized tolerance parameter. In Table 2.2 all vector squares, divisions and multiplications are taken element wise.

The algorithm proposed can be considered an extension of the previously proposed AMP TSBL algorithm in [67]. The extension to GGAMP-TSBL includes removing the averaging of the matrix $\boldsymbol{A}$ in the derivation of the algorithm, and it includes introducing the same damping strategy used in the SMV case to improve convergence. The complexity of the GGAMP-TSBL algorithm is also dominated by the E-step which in turn is dominated by matrix multiplications by $\boldsymbol{A}$, $\boldsymbol{A}^\top$, $\boldsymbol{S}$ and $\boldsymbol{S}^\top$, implying that the computational cost is $O(MN)$ flops per iteration per frame.

Therefore the complexity of the proposed algorithm is $O(TMN)$ multiplied by the total number of GAMP algorithm iterations.

2.4.3 Derivation of GGAMP-TSBL Updates

The Within Step Updates

To make the factor graph for the within step in Fig. 2.4 exactly the same as the SMV factor graph we combine the product of the two messages incoming from $f_n^{(t)}$ and $f_n^{(t+1)}$ to $x_n^{(t)}$ into one message as follows:

$$\begin{aligned} V_{f_n^{(t)}\to x_n^{(t)}} &\propto \mathcal{N}(x_n^{(t)};\eta_n^{(t)},\psi_n^{(t)}) \\ V_{f_n^{(t+1)}\to x_n^{(t)}} &\propto \mathcal{N}(x_n^{(t)};\theta_n^{(t)},\phi_n^{(t)}) \\ V_{\bar{f}_n^{(t)}\to x_n^{(t)}} &\propto \mathcal{N}(x_n^{(t)};\rho_n^{(t)},\zeta_n^{(t)}) \\ &\propto \mathcal{N}(x_n^{(t)};\frac{\frac{\eta_n^{(t)}}{\psi_n^{(t)}}+\frac{\theta_n^{(t)}}{\phi_n^{(t)}}}{\frac{1}{\psi_n^{(t)}}+\frac{1}{\phi_n^{(t)}}},\frac{1}{\frac{1}{\psi_n^{(t)}}+\frac{1}{\phi_n^{(t)}}}). \end{aligned} \tag{2.25}$$

Combining these two messages reduces each time frame factor graph to an equivalent one to the SMV case with a modified prior on $x_n^{(t)}$ of (2.25). Applying the damped GAMP algorithm from [52] with $p(x_n^{(t)})$ given in (2.25):

$$\begin{aligned} g_x^{(t)} &= \frac{\frac{\boldsymbol{r}^{(t)}}{\boldsymbol{\tau}_r^{(t)}}+\frac{\boldsymbol{\rho}^{(t)}}{\boldsymbol{\zeta}^{(t)}}}{\frac{1}{\boldsymbol{\tau}_r^{(t)}}+\frac{1}{\boldsymbol{\zeta}^{(t)}}} = \frac{\frac{\boldsymbol{r}^{(t)}}{\boldsymbol{\tau}_r^{(t)}}+\frac{\boldsymbol{\eta}^{(t)}}{\boldsymbol{\psi}^{(t)}}+\frac{\boldsymbol{\theta}^{(t)}}{\boldsymbol{\phi}^{(t)}}}{\frac{1}{\boldsymbol{\tau}_r^{(t)}}+\frac{1}{\boldsymbol{\psi}^{(t)}}+\frac{1}{\boldsymbol{\phi}^{(t)}}} \\ \boldsymbol{\tau}_x^{(t)} &= \frac{1}{\frac{1}{\boldsymbol{\tau}_r^{(t)}}+\frac{1}{\boldsymbol{\zeta}^{(t)}}} = \frac{1}{\frac{1}{\boldsymbol{\tau}_r^{(t)}}+\frac{1}{\boldsymbol{\psi}^{(t)}}+\frac{1}{\boldsymbol{\phi}^{(t)}}}. \end{aligned}$$

Forward Message Updates

$$
\begin{aligned}
V_{f_n^{(1)}\to x_n^{(1)}} &\propto \mathcal{N}(x_n^{(1)};0,\gamma_n)\\
V_{f_n^{(t)}\to x_n^{(t)}} &\propto \mathcal{N}(x_n^{(t)};\eta_n^{(t)},\psi_n^{(t)})\\
&\propto \int\left(\prod_{l=1}^{M} V_{g_l^{(t-1)}\to x_n^{(t-1)}}\right) V_{f_n^{(t-1)}\to x_n^{(t-1)}} P(x_n^{(t)}\,|\,x_n^{(t-1)})\,\mathrm{d}x_n^{(t-1)}\\
&\propto \int \mathcal{N}(x_n^{(t-1)};r_n^{(t-1)},\tau_{r_n}^{(t-1)})\;\mathcal{N}(x_n^{(t-1)};\eta_n^{(t-1)},\psi_n^{(t-1)})\mathcal{N}(x_n^{(t)};\beta x_n^{(t-1)},(1-\beta^2)\gamma_n)\,\mathrm{d}x_n^{(t-1)}.
\end{aligned}
$$

Using rules for Gaussian pdf multiplication and convolution we get the $\eta_n^{(t)}$ and $\psi_n^{(t)}$ updates given in Table 2.2 equations (E3) and (E4).

Backward Message Updates

$$
\begin{aligned}
V_{f_n^{(t+1)}\to x_n^{(t)}} &\propto \mathcal{N}(x_n^{(t)};\theta_n^{(t)},\phi_n^{(t)})\\
&\propto \int\left(\prod_{l=1}^{M} V_{g_l^{(t+1)}\to x_n^{(t+1)}}\right) V_{f_n^{(t+2)}\to x_n^{(t+1)}} P(x_n^{(t+1)}\,|\,x_n^{(t)})\,\mathrm{d}x_n^{(t+1)}\\
&\propto \int \mathcal{N}(x_n^{(t+1)};r_n^{(t+1)},\tau_{r_n}^{(t+1)})\;\mathcal{N}(x_n^{(t+1)};\theta_n^{(t+1)},\phi_n^{(t+1)})\;\mathcal{N}(x_n^{(t+1)};\beta x_n^{(t)},(1-\beta^2)\gamma_n)\,\mathrm{d}x_n^{(t+1)}.
\end{aligned}
$$

Using rules for Gaussian pdf multiplication and convolution we get the $\theta_n^{(t)}$ and $\phi_n^{(t)}$ updates given in Table 2.2 equations (E13) and (E14).

2.4.4 GGAMP-TSBL M-Step

Upon the convergence of the E-step, the M-step learns $\boldsymbol{\gamma}$ from the data by treating $\boldsymbol{x}$ as a hidden variable and then maximizing $\mathbb{E}_{\bar{\boldsymbol{x}}|\bar{\boldsymbol{y}};\boldsymbol{\gamma}^i,\sigma^2,\beta}[\log p(\bar{\boldsymbol{y}},\bar{\boldsymbol{x}},\boldsymbol{\gamma};\sigma^2,\beta)]$.

$$
\begin{aligned}
\boldsymbol{\gamma}^{i+1} &= \operatorname*{argmin}_{\boldsymbol{\gamma}} \mathbb{E}_{\bar{\boldsymbol{x}}|\bar{\boldsymbol{y}};\boldsymbol{\gamma}^i,\sigma^2,\beta}[-\log p(\bar{\boldsymbol{y}},\bar{\boldsymbol{x}},\boldsymbol{\gamma};\sigma^2,\beta)]\\
\gamma_n^{i+1} &= \frac{1}{T}\left[|\hat{x}_n^{(1)}|^2+\tau_{x_n}^{(1)}+\frac{1}{1-\beta^2}\sum_{t=2}^{T}|\hat{x}_n^{(1)}|^2+\tau_{x_n}^{(t)}+\beta(|\hat{x}_n^{(t-1)}|^2+\tau_{x_n}^{(t-1)})-2\beta(\hat{x}_n^{(t)}\hat{x}_n^{(t-1)}+\beta\tau_{x_n}^{(t-1)})\right]
\end{aligned}
\tag{2.26}
$$

Table 2.2: GGAMP-TSBL algorithm

Definitions

$$F(\boldsymbol{r}^{k(t)},\boldsymbol{\tau}_r^{k(t)}) = \frac{\frac{\boldsymbol{r}^{k(t)}}{\boldsymbol{\tau}_r^{k(t)}}+\frac{\boldsymbol{\eta}^{k(t)}}{\boldsymbol{\psi}^{k(t)}}+\frac{\boldsymbol{\theta}^{k(t)}}{\boldsymbol{\phi}^{k(t)}}}{\frac{1}{\boldsymbol{\tau}_r^{k(t)}}+\frac{1}{\boldsymbol{\psi}^{k(t)}}+\frac{1}{\boldsymbol{\phi}^{k(t)}}} \quad \text{(D1)}$$

$$G(\boldsymbol{r}^{k(t)},\boldsymbol{\tau}_r^{k(t)}) = \frac{1}{\frac{1}{\boldsymbol{\tau}_r^{k(t)}}+\frac{1}{\boldsymbol{\psi}^{k(t)}}+\frac{1}{\boldsymbol{\phi}^{k(t)}}} \quad \text{(D2)}$$

Initialization

$\boldsymbol{S} \leftarrow |\boldsymbol{A}|^2$ (component wise magnitude squared) (N1)

Initialize $\forall t : \breve{\boldsymbol{\tau}}_x^{0(t)}, \boldsymbol{\gamma}^0 > 0$, $\breve{\boldsymbol{s}}^{0(t)} \leftarrow \boldsymbol{0}$ and $\breve{\boldsymbol{x}}^{0(t)} \leftarrow \boldsymbol{0}$ (N3)

for $i = 1, 2, ..., I_{\max}$

Initialize $\forall t : \boldsymbol{\tau}_x^{1(t)} \leftarrow \breve{\boldsymbol{\tau}}_x^{i-1(t)}, \hat{\boldsymbol{x}}^{1(t)} \leftarrow \breve{\boldsymbol{x}}^{i-1(t)}$, $\boldsymbol{s}^{1(t)} \leftarrow \breve{\boldsymbol{s}}^{i-1(t)}$

E-Step approximation

for $k = 1, 2, ..., K_{\max}$

$$\boldsymbol{\eta}^{k(1)} \leftarrow 0 \quad \text{(E1)}$$

$$\boldsymbol{\psi}^{k(1)} \leftarrow \boldsymbol{\gamma}^i \quad \text{(E2)}$$

for $t = 2 : T$

$$\boldsymbol{\eta}^{k(t)} \leftarrow \beta\left(\frac{\boldsymbol{r}^{k(t-1)}}{\boldsymbol{\tau}_r^{k(t-1)}}+\frac{\boldsymbol{\eta}^{k(t-1)}}{\boldsymbol{\psi}^{k(t-1)}}\right)\left(\frac{\boldsymbol{\psi}^{k(t-1)}\boldsymbol{\tau}_r^{k(t-1)}}{\boldsymbol{\psi}^{k(t-1)}+\boldsymbol{\tau}_r^{k(t-1)}}\right) \quad \text{(E3)}$$

$$\boldsymbol{\psi}^{k(t)} \leftarrow \beta^2\left(\frac{\boldsymbol{\psi}^{k(t-1)}\boldsymbol{\tau}_r^{k(t-1)}}{\boldsymbol{\psi}^{k(t-1)}+\boldsymbol{\tau}_r^{k(t-1)}}\right)+(1-\beta^2)\boldsymbol{\gamma}^i \quad \text{(E4)}$$

end for %end of t loop

for $t = 1 : T$

$$1/\boldsymbol{\tau}_p^{k(t)} \leftarrow \boldsymbol{S}\boldsymbol{\tau}_x^{k(t)} \quad \text{(E5)}$$

$$\boldsymbol{p}^{k(t)} \leftarrow \boldsymbol{s}^{k-1(t)}+\boldsymbol{\tau}_p^{k(t)}\boldsymbol{A}\hat{\boldsymbol{x}}^{k(t)} \quad \text{(E6)}$$

$$\boldsymbol{\tau}_s^{k(t)} \leftarrow \frac{\sigma^{-2}\boldsymbol{\tau}_p^{k(t)}}{\sigma^{-2}+\boldsymbol{\tau}_p^{k(t)}} \quad \text{(E7)}$$

$$\boldsymbol{s}^{k(t)} \leftarrow (1-\theta_s)\boldsymbol{s}^{k-1(t)}+\theta_s\frac{\left(\frac{\boldsymbol{p}^{k(t)}}{\boldsymbol{\tau}_p^{k(t)}}-\boldsymbol{y}^{(t)}\right)}{(\sigma^2+1/\boldsymbol{\tau}_p^{k(t)})} \quad \text{(E8)}$$

$$1/\boldsymbol{\tau}_r^{k(t)} \leftarrow \boldsymbol{S}^\top\boldsymbol{\tau}_s^{k(t)} \quad \text{(E9)}$$

$$\boldsymbol{r}^{k(t)} \leftarrow \hat{\boldsymbol{x}}^{(t)}-\boldsymbol{\tau}_r^{k(t)}\boldsymbol{A}^\top\boldsymbol{s}^{k(t)} \quad \text{(E10)}$$

$$\boldsymbol{\tau}_x^{k+1(t)} \leftarrow G(\boldsymbol{r}^{k(t)},\boldsymbol{\tau}_r^{k(t)}) \quad \text{(E11)}$$

$$\hat{\boldsymbol{x}}^{k+1(t)} \leftarrow (1-\theta_x)\hat{\boldsymbol{x}}^{k(t)}+\theta_x F_n(\boldsymbol{r}^{k(t)},\boldsymbol{\tau}_r^{k(t)}) \quad \text{(E12)}$$

end for %end of t loop

for $t = T-1 : 1$

$$\boldsymbol{\theta}^{k(t)} \leftarrow \frac{1}{\beta}\left(\frac{\boldsymbol{r}^{k(t+1)}}{\boldsymbol{\tau}_r^{k(t+1)}}+\frac{\boldsymbol{\theta}^{k(t+1)}}{\boldsymbol{\phi}^{k(t+1)}}\right)\left(\frac{\boldsymbol{\phi}^{k(t+1)}\boldsymbol{\tau}_r^{k(t+1)}}{\boldsymbol{\theta}^{k(t+1)}+\boldsymbol{\tau}_r^{k(t+1)}}\right) \quad \text{(E13)}$$

$$\boldsymbol{\phi}^{k(t)} \leftarrow \frac{1}{\beta^2}\left(\frac{\boldsymbol{\phi}^{k(t+1)}\boldsymbol{\tau}_r^{k(t+1)}}{\boldsymbol{\phi}^{k(t+1)}+\boldsymbol{\tau}_r^{k(t+1)}}+(1-\beta^2)\boldsymbol{\gamma}^i\right) \quad \text{(E14)}$$

end for %end of t loop

if $\frac{1}{T}\sum_{t=1}^{T}\left(\frac{\|\hat{\boldsymbol{x}}^{k+1(t)}-\hat{\boldsymbol{x}}^{k(t)}\|^2}{\|\hat{\boldsymbol{x}}^{k+1(t)}\|^2}\right) < \varepsilon_{\text{gamp}}$, break (E15)

end for %end of k loop

$\forall t, \breve{\boldsymbol{s}}^{i(t)} \leftarrow \boldsymbol{s}^{k+1(t)}, \breve{\boldsymbol{x}}^{i(t)} \leftarrow \hat{\boldsymbol{x}}^{k+1(t)}$, $\breve{\boldsymbol{\tau}}_x^{i(t)} \leftarrow \boldsymbol{\tau}_x^{k+1(t)}$

M-step

$$\gamma_n^{i+1} = \frac{1}{T}\Bigg[|\breve{x}_n^{i(1)}|^2+\breve{\tau}_{x_n}^{i(1)}+\sum_{t=2}^{T}\frac{|\breve{x}_n^{i(t)}|^2+\breve{\tau}_{x_n}^{i(t)}}{1-\beta^2} +\frac{\beta}{1-\beta^2}\sum_{t=2}^{T}\left(|\breve{x}_n^{i(t-1)}|^2+\breve{\tau}_{x_n}^{i(t-1)}\right) -\frac{2\beta}{1-\beta^2}\sum_{t=2}^{T}\left(\breve{x}_n^{i(t)}\breve{x}_n^{i(t-1)}+\beta\breve{x}_n^{i(t-1)}\right)\Bigg] \quad \text{(U1)}$$

if $\frac{1}{T}\sum_{t=1}^{T}\left(\frac{\|\breve{\boldsymbol{x}}^{i(t)}-\breve{\boldsymbol{x}}^{i-1(t)}\|^2}{\|\breve{\boldsymbol{x}}^{i(t)}\|^2}\right) < \varepsilon_{\text{em}}$, break (U2)

end for %end of i loop

The derivation of γ_n^{i+1} M-step update follows the same steps as the SMV case. The derivation is omitted here due to space limitation, and γ_n^{i+1} update is given in (2.26). We note here that the M-step $\boldsymbol{\gamma}$ learning rule in (2.26) is the same as the one derived in [64]. Both algorithms use the same AR(1) model for $\boldsymbol{x}^{(t)}$, but they differ in the implementation of the E-step. In the case that the correlation coefficient β or the noise variance σ^2 are unknown, the EM algorithm can be used to estimate their values as well.

2.5 Numerical Results

In this section we present a numerical study to illustrate the performance and complexity of the proposed GGAMP-SBL and GGAMP-TSBL algorithms. The performance and complexity were studied through two metrics. The first metric studies the ability of the algorithm to recover $\boldsymbol{x}$, for which we use the normalized mean squared error NMSE in the SMV case:

$$\text{NMSE} \triangleq \|\hat{\boldsymbol{x}} - \boldsymbol{x}\|^2 / \|\boldsymbol{x}\|^2,$$

and the time-averaged normalized mean squared error TNMSE in the MMV case:

$$\text{TNMSE} \triangleq \frac{1}{T} \sum_{t=1}^{T} \|\hat{\boldsymbol{x}}^{(t)} - \boldsymbol{x}^{(t)}\|^2 / \|\boldsymbol{x}^{(t)}\|^2.$$

The second metric studies the complexity of the algorithm by tracking the time the algorithm requires to compute the final estimate $\hat{\boldsymbol{x}}$. We measure the time in seconds. While the absolute runtime could vary if the same experiments were to be run on a different machine, the runtimes of the algorithms of interest in relationship to each other is a good estimate of the relative computational complexity.

Several types of non-i.i.d.-Gaussian matrix were used to explore the robustness of the proposed algorithms relative to the standard SBL and TMSBL. The four different types of matrices

are similar to the ones previously used in [54] and are described as follows:

-Column correlated matrices: The rows of $\boldsymbol{A}$ are independent zero-mean Gaussian Markov processes with the following correlation coefficient $\rho = \mathbb{E}\{\boldsymbol{a}_{.n}^{\top}\boldsymbol{a}_{.(n+1)}\}/\mathbb{E}\{|\boldsymbol{a}_{.n}|^2\}$, where $\boldsymbol{a}_{.n}$ is the n^{th} column of $\boldsymbol{A}$. In the experiments the correlation coefficient ρ is used as the measure of deviation from the i.i.d.-Gaussian matrix.

-Low rank product matrices: We construct a rank deficient $\boldsymbol{A}$ by $\boldsymbol{A} = \frac{1}{N}\boldsymbol{H}\boldsymbol{G}$ with $\boldsymbol{H} \in \mathbb{R}^{M\times R}$, $\boldsymbol{G} \in \mathbb{R}^{R\times N}$ and $R < M$. The entries of $\boldsymbol{H}$ and $\boldsymbol{G}$ are i.i.d.-Gaussian with zero mean and unit variance. The rank ratio R/N is used as the measure of deviation from the i.i.d.-Gaussian matrix.

-Ill conditioned matrices: we construct $\boldsymbol{A}$ with a condition number $\kappa > 1$ as follows. $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\top}$, where $\boldsymbol{U}$ and $\boldsymbol{V}^{\top}$ are the left and right singular vector matrices of an i.i.d.-Gaussian matrix, and $\boldsymbol{\Sigma}$ is a singular value matrix with $\Sigma_{i,i}/\Sigma_{i+1,i+1} = \kappa^{1/(M-1)}$ for $i = 1, 2, ..., M-1$. The condition number κ is used as the measure of deviation from the i.i.d.-Gaussian matrix.

-Non-zero mean matrices: The elements of $\boldsymbol{A}$ are $a_{m,n} \sim \mathcal{N}(\mu, \frac{1}{N})$. The mean μ is used as a measure of deviation from the zero-mean i.i.d.-Gaussian matrix. It is worth noting that in the case of non-zero mean $\boldsymbol{A}$, convergence of the GGAMP-SBL is not enhanced by damping but more by the mean removal procedure explained in [54]. We include it in the implementation of our algorithm, and we include it in the numerical results to make the study more inclusive of different types of generic $\boldsymbol{A}$ matrices.

Although we have provided an estimation procedure, based on the EM algorithm, for the noise variance σ^2 in (2.13), in all experiments we assume that the noise variance σ^2 is known. We also found that the SBL algorithm does not necessarily have the best performance when the exact σ^2 is used, and in our case, it was empirically found that using an estimate $\hat{\sigma}^2 = 3\sigma^2$ yields better results. Therefore $\hat{\sigma}^2$ is used for SBL, TMSBL, GGAMP-SBL and GGAMP-TSBL throughout our experiments.

2.5.1 SMV GGAMP-SBL Numerical Results

In this section we compare the proposed SMV algorithm (GGAMP-SBL) against the original SBL and against two AMP algorithms that have shown improvement in robustness over the original AMP/GAMP, namely the SwAMP algorithm [55] and the MADGAMP algorithm [54]. As a performance benchmark, we use a lower bound on the achievable NMSE which is similar to the one in [54]. The bound is found using a "genie" that knows the support of the sparse vector $\boldsymbol{x}$. Based on the known support, $\bar{\boldsymbol{A}}$ is constructed from the columns of $\boldsymbol{A}$ corresponding to non-zero elements of $\boldsymbol{x}$, and an MMSE solution using $\bar{\boldsymbol{A}}$ is computed.

$$\hat{\boldsymbol{x}} = \bar{\boldsymbol{A}}^{\top}(\bar{\boldsymbol{A}}\bar{\boldsymbol{A}}^{\top} + \sigma^2\boldsymbol{I})^{-1}\boldsymbol{y}.$$

In all SMV experiments, $\boldsymbol{x}$ had exactly K non-zero elements in random locations, and the nonzero entires were drawn independently from a zero-mean unit-variance Gaussian distribution. In accordance with the model (2.1), an AWGN channel was used with the SNR defined by:

$$\text{SNR} \triangleq \mathbb{E}\{\|\boldsymbol{A}\boldsymbol{x}\|^2\}/\mathbb{E}\{\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|^2\}.$$

Robustness to generic matrices at high SNR

The first experiment investigates the robustness of the proposed algorithm to generic $\boldsymbol{A}$ matrices. It compares the algorithms of interest using the four types of matrices mentioned above, over a range of deviation from the i.i.d.-Gaussian case. For each matrix type, we start with an i.i.d.-Gaussian $\boldsymbol{A}$ and increase the deviation over 11 steps. We monitor how much deviation the different algorithms can tolerate, before we start seeing significant performance degradation compared to the "genie" bound. The vector $\boldsymbol{x}$ was drawn from a Bernoulli-Gaussian distribution with non-zero probability $\lambda = 0.2$, with $N = 1000$, $M = 500$ and SNR $= 60$dB.

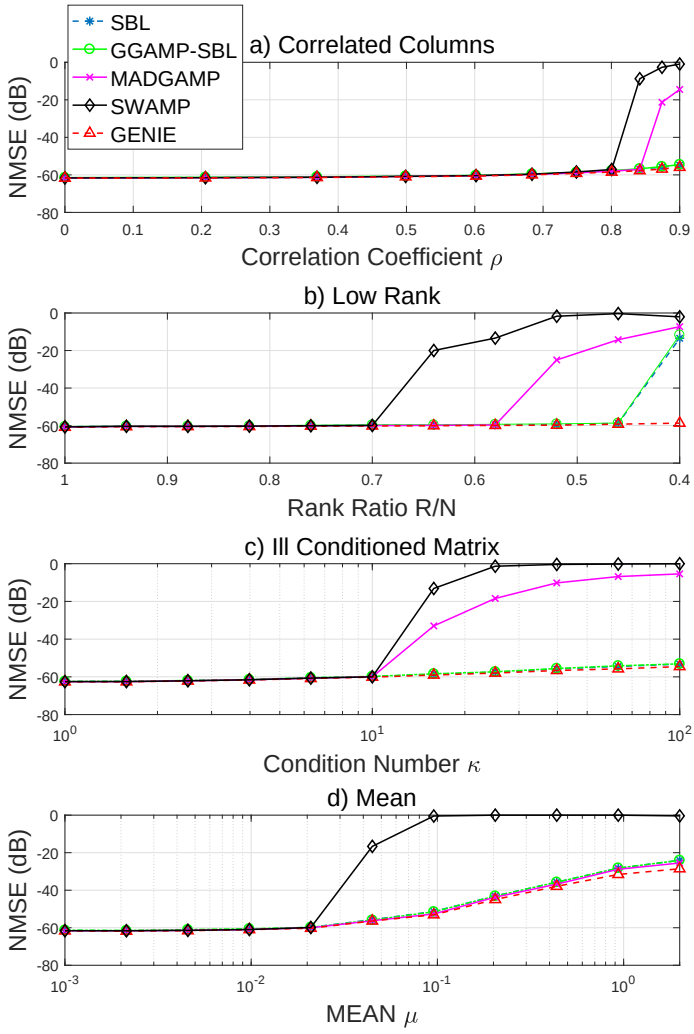


Figure 2.5: NMSE comparison of SMV algorithms under non-i.i.d.-Gaussian $\boldsymbol{A}$ matrices with SNR=60dB

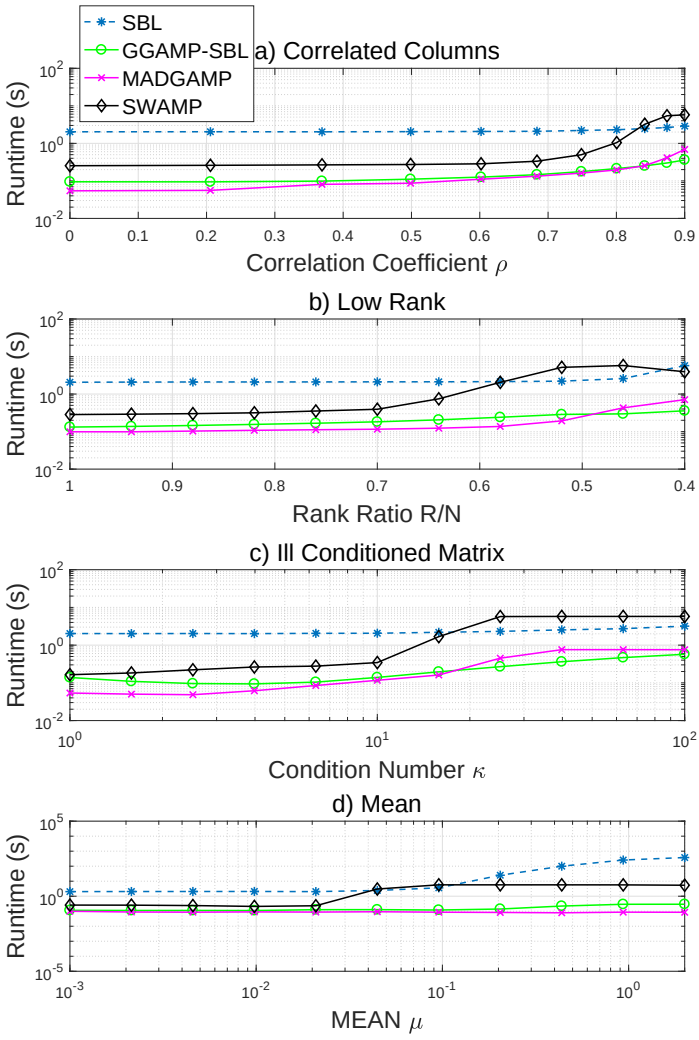


Figure 2.6: Runtime comparison of SMV algorithms under non-i.i.d.-Gaussian $\boldsymbol{A}$ matrices with SNR=60dB

The NMSE results in Fig. 2.5 show that the performance of GGAMP-SBL was able to match that of the original SBL even for $\boldsymbol{A}$ matrices with the most deviation from the i.i.d.-Gaussian case. Both algorithms nearly achieved the bound in most cases, with the exception when the matrix is low rank with a rank ratio less than 0.45 where both algorithms fail to achieve the bound. This supports the evidence we provided before for the convergence of the GGAMP-SBL algorithm, which predicted its ability to match the performance of the original SBL. As for other AMP implementations, despite the improvement in robustness they provide over traditional AMP/GAMP, they cannot guarantee convergence beyond a certain point, and their robustness is surpassed by GGAMP-SBL in most cases. The only exception is when $\boldsymbol{A}$ is non-zero mean, where the GGAMP-SBL and the MADGAMP algorithms share similar performance. This is due to the fact that both algorithms use mean removal to transform the problem into a zero-mean equivalent problem, which both algorithms can handle well.

The complexity of the GGAMP-SBL algorithm is studied in Fig. 2.6. The figure shows how the GGAMP-SBL was able to reduce the complexity compared to the original SBL implementation. It also shows that even when the algorithm is slowed down by heavier damping, the algorithm still has faster runtimes than the original SBL.

Robustness to generic matrices at lower SNR

In this experiment we examine the performance and complexity of the proposed algorithm at a lower SNR setting than the previous experiment. We lower the SNR to 30dB and collect the same data points as in the previous experiment. The results in Fig. 2.7 show that the performance of the GGAMP-SBL algorithm is still generally matching that of the original SBL algorithm with slight degradation. The MADGAMP algorithm provides slightly better performance than both SBL algorithms when the deviation from the i.i.d.-sub-Gaussian case is not too large. This can be due to the fact that we choose to run the MADGAMP algorithm with exact knowledge of the data model rather than learn the model parameters, while both SBL algorithms have information about

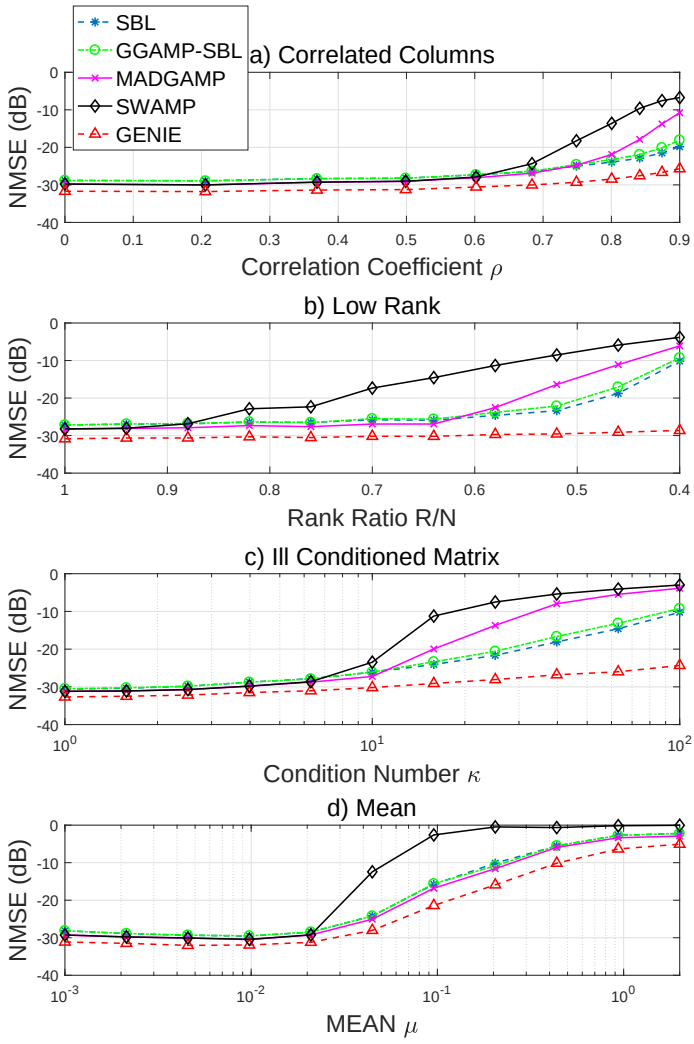


Figure 2.7: NMSE comparison of SMV algorithms under non-i.i.d.-Gaussian $\boldsymbol{A}$ matrices with SNR=30dB

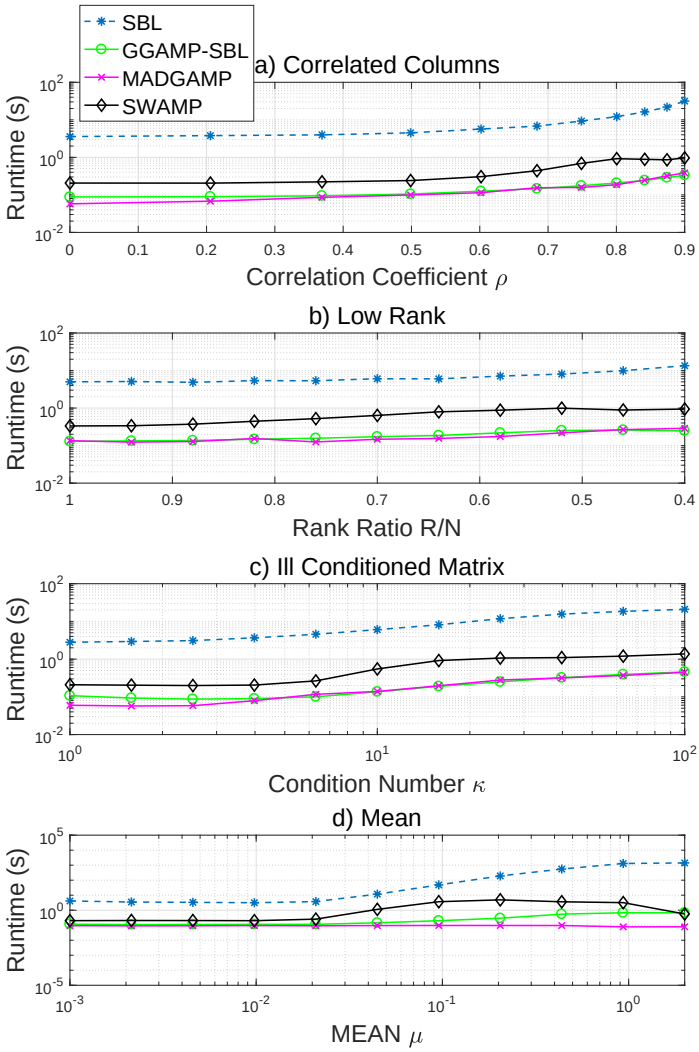


Figure 2.8: Runtime comparison of SMV algorithms under non-i.i.d.-Gaussian $\boldsymbol{A}$ matrices with SNR=30dB

the noise variance only. As the deviation in $\boldsymbol{A}$ increases, GGAMP-SBL's performance surpasses MADGAMP and SWAMP algorithms, providing better robustness at lower SNR.

On the complexity side, we see from Fig. 2.8 that the GGAMP-SBL continues to have reduced complexity compared to the original SBL.

Performance and complexity versus problem dimensions

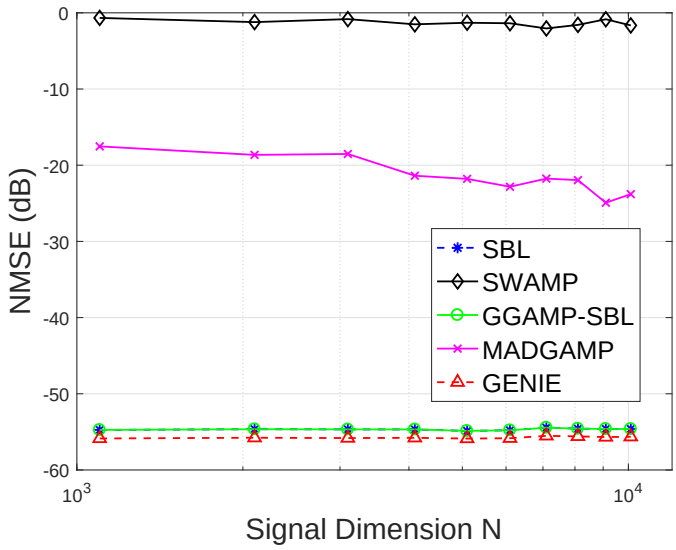
To show the effect of increasing the problem dimensions on the performance and complexity of the different algorithms, we plot the NMSE and runtime against N, while we keep an M/N ratio of 0.5, a K/N ratio of 0.2 and an SNR of 60dB. We run the experiment using column correlated matrices with $\rho = 0.9$.

As expected from previous experiments, Fig. 2.9a shows that only GGAMP-SBL and SBL algorithms can recover $\boldsymbol{x}$ when we use column correlated matrices with a correlation coefficient of $\rho = 0.9$. The comparison between the performance of SBL and GGAMP-SBL show almost identical NMSE.

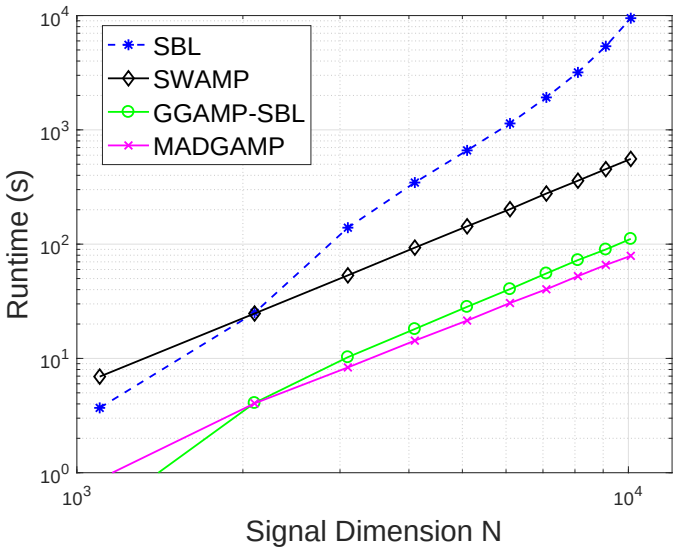
As problem dimensions grow, Fig. 2.9b shows that the difference in runtimes between the original SBL and GGAMP-SBL algorithms grows to become more significant, which suggests that the GGAMP-SBL is more practical for large size problems.

Performance versus undersampling ratio M/N

In this section we examine the ability of the proposed algorithm to recover a sparse vector from undersampled measurements at different undersampling ratios M/N. In the below experiments we fix N at 1000 and vary M. We set the Bernoulli-Gaussian non-zero probability λ so that M/K has an average of three measurements for each non-zero component. We plot the NMSE versus the undersampling ratio M/N for i.i.d.-Gaussian matrices $\boldsymbol{A}$ and for column correlated $\boldsymbol{A}$ with $\rho = 0.9$. We run the experiments at SNR=60dB and at SNR=30dB. In Fig. 2.10 we find that for SNR=60dB and i.i.d.-Gaussian $\boldsymbol{A}$, all algorithms meet the SKS bound when the



(a) NMSE versus N



(b) Runtime versus N

Figure 2.9: Performance and complexity comparison for SMV algorithms versus problem dimensions

undersampling ratio is larger than or equal to 0.25, while all algorithms fail to meet the bound at any ratio smaller than that. When $\boldsymbol{A}$ is column correlated, SBL and GGAMP-SBL are able to meet the SKS bound at $M/N \geq 0.3$, while MADGAMP and SwAMP do not meet the bound even at $M/N = 0.5$. We also note the MADGAMP's NMSE slowly improves with increased underasampling ratio, while SwAMP's NMSE does not. At SNR=30dB, with i.i.d.-Gaussian $\boldsymbol{A}$ all algorithms are close to the SKS bound when the undersampling ratio is larger than 0.3. At $M/N \leq 0.3$, SBL and GGAMP-SBL are slightly outperformed by MADGAMP, while SwAMP seems to have the best performance in this region. When $\boldsymbol{A}$ is column correlated, NMSE of SBL and GGAMP-SBL outperform the other two algorithms, and similar to the SNR=60dB case, MADGAMP's NMSE seems to slowly improve with increased undersampling ratio, while SwAMP's NMSE does not improve.

2.5.2 MMV GGAMP-TSBL Numerical Results

In this section, we present a numerical study to illustrate the performance and complexity of the proposed GGAMP-TSBL algorithm. Although the AMP MMV algorithm in [65] can be extended to incorporate damping, the current implementation of AMP MMV does not include damping and will diverge when used with the type of generic $\boldsymbol{A}$ matrices we are considering for our experiments. Therefore, we restrict the comparison of the performance and complexity of the GGAMP-TSBL algorithm to the TMSBL algorithm. We also compare the recovery performance against a lower bound on the achievable TNMSE by extending the support aware Kalman smoother (SKS) from [65] to include damping and hence be able to handle generic $\boldsymbol{A}$ matrices. The implementation of the smoother is straight forward, and is exactly the same as the E-step part in Table 2.2, when the true values of σ^2, $\boldsymbol{\gamma}$ and β are used, and when $\boldsymbol{A}$ is modified to include only the columns corresponding to the non-zero elements in $\boldsymbol{x}^{(t)}$. An AWGN channel was also assumed in the case of MMV.

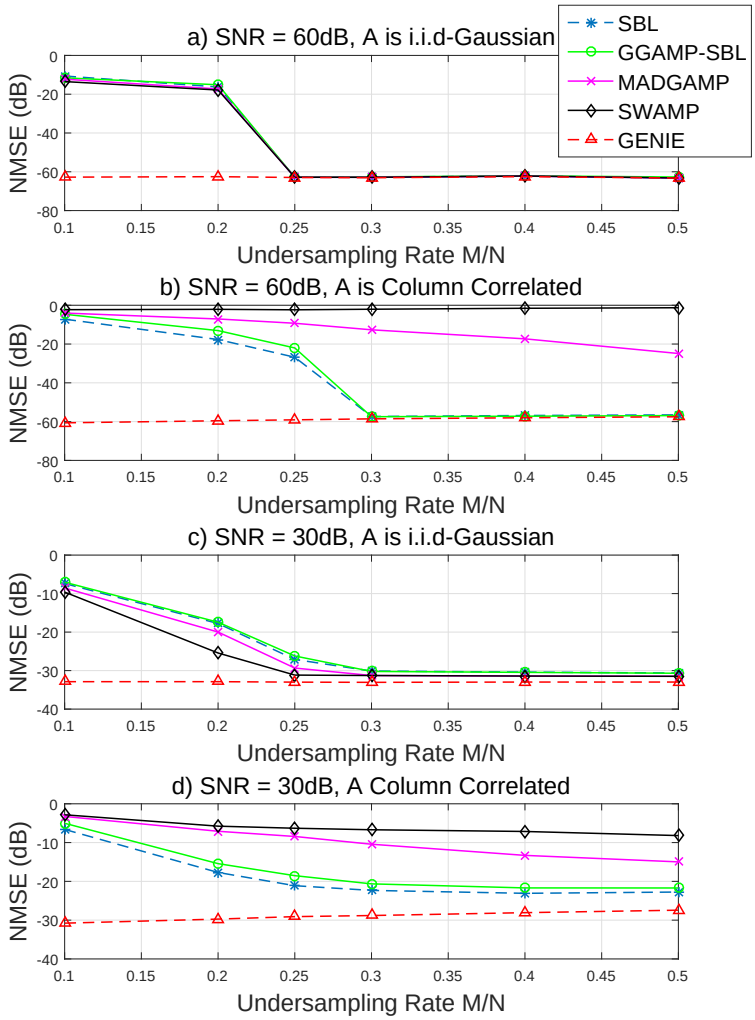


Figure 2.10: NMSE comparison of SMV algorithms versus the undersampling rate M/N

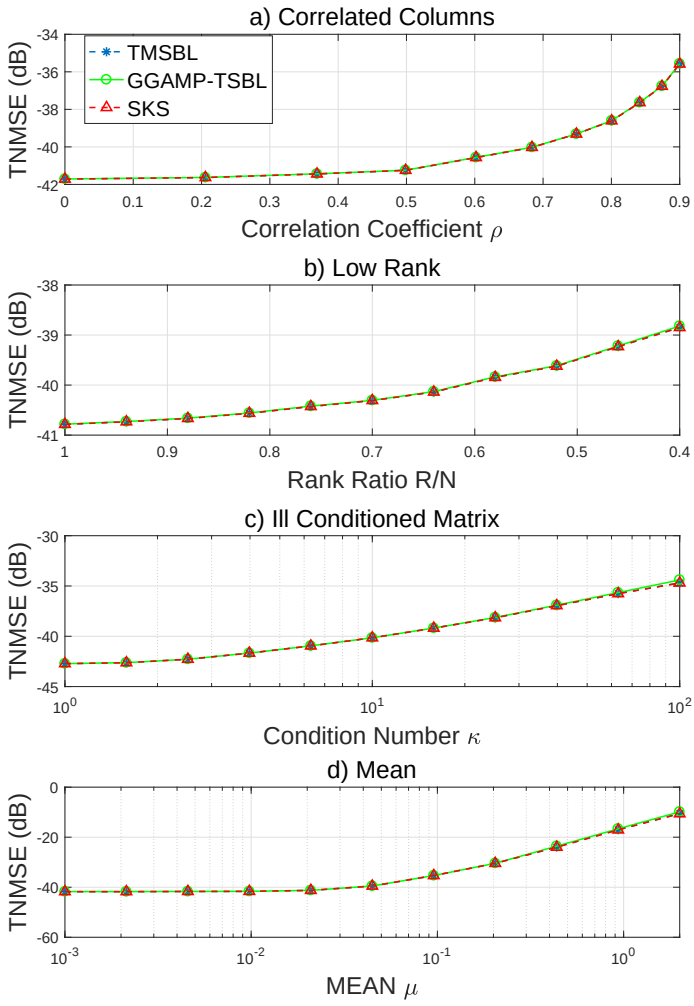


Figure 2.11: TNMSE comparison of MMV algorithms under non-i.i.d.-Gaussian $\boldsymbol{A}$ matrices with SNR=60dB

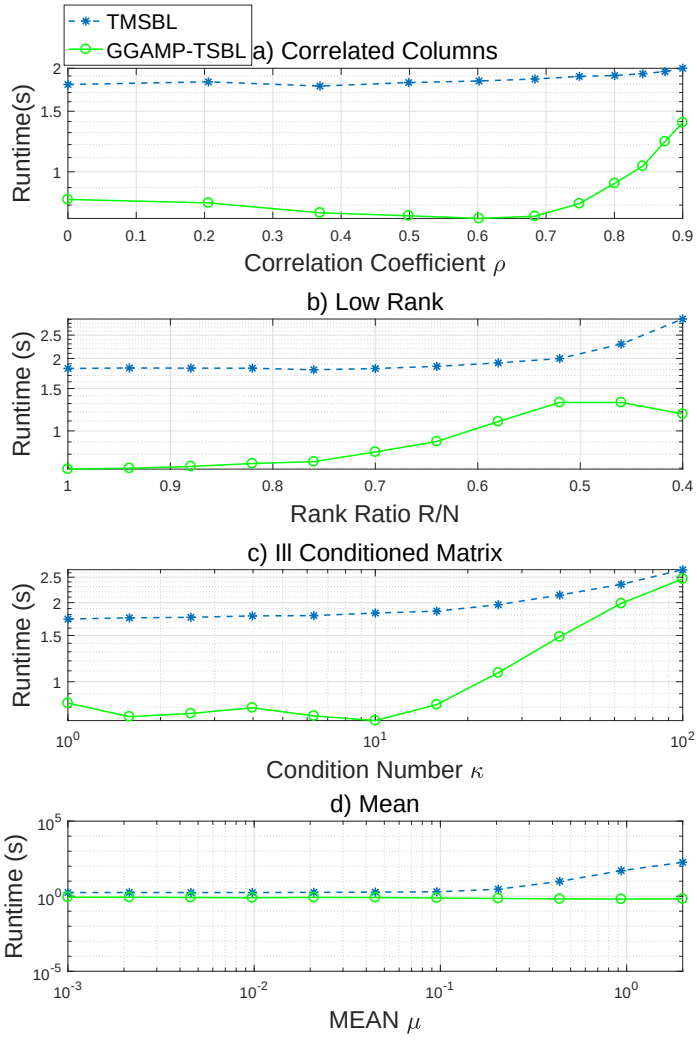


Figure 2.12: Runtime comparison of MMV algorithms under non-i.i.d.-Gaussian $\boldsymbol{A}$ matrices with SNR=60dB

Robustness to generic matrices at high SNR

The experiment investigates the robustness of the proposed algorithm by comparing it to the TMSBL and the support aware smoother. Once again we use the four types of matrices mentioned at the beginning of this section, over the same range of deviation from the i.i.d.-Gaussian case. For this experiment we set $N = 1000$, $M = 500$, $\lambda = 0.2$, $SNR = 60$dB and the temporal correlation coefficient β to 0.9. We choose a relatively high value for β to provide large deviation from the SMV case. This is due to the fact that the no correlation case is reduced to solving multiple SMV instances in the E-step, and then applying the M-step to update the hyperparameter vector $\boldsymbol{\gamma}$, which is common across time frames [60]. The TNMSE results in Fig. 2.11 show that the performance of GGAMP-TSBL was able to match that of TMSBL in all cases and they both achieved the SKS bound.

Once again Fig. 2.12 shows that the proposed GGAMP-TSBL was able to reduce the complexity compared to the TMSBL algorithm, even when damping was used. Although the complexity reduction does not seem to be significant for the selected problem size and SNR, we will see in the following experiments how this reduction becomes more significant as the problem size grows or as a lower SNR is used.

Robustness to generic matrices at lower SNR

The performance and complexity of the proposed algorithm are examined at a lower SNR setting than the previous experiment. We set the SNR to 30dB and collect the same data points collected as in the 60dB SNR case. Fig. 2.13 shows that the GGAMP-TSBL performance matches that of the TMSBL and almost achieves the bound in most cases. Similar to the previous cases, Fig. 2.14 shows that the complexity of GGAMP-TSBL is lower than that of TMSBL.

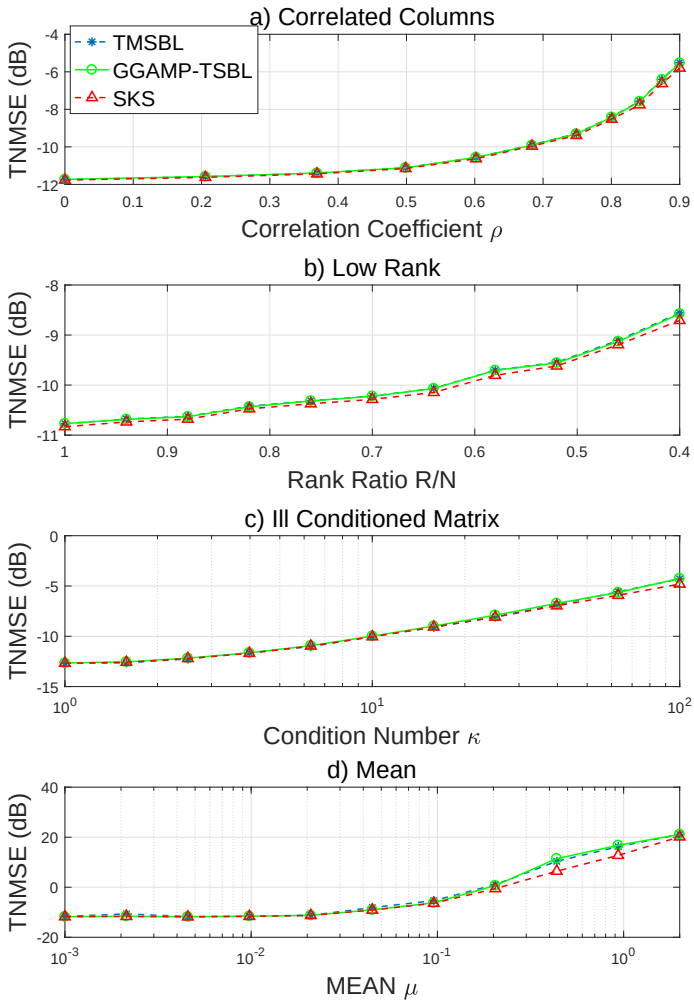


Figure 2.13: TNSME comparison of MMV algorithms under non-i.i.d.-Gaussian $\boldsymbol{A}$ matrices with SNR=30dB

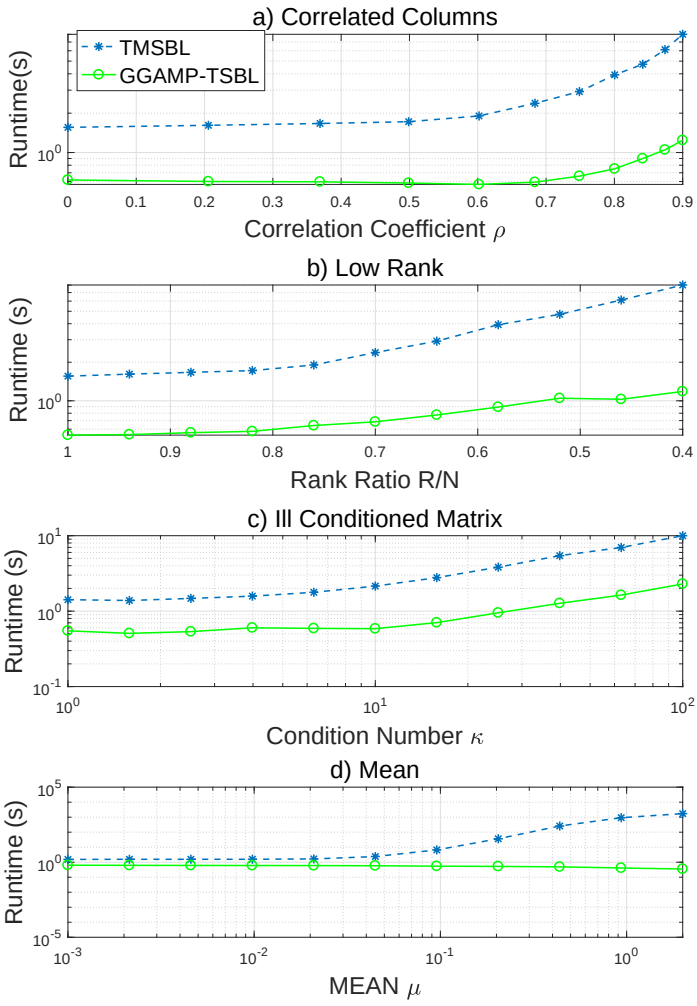
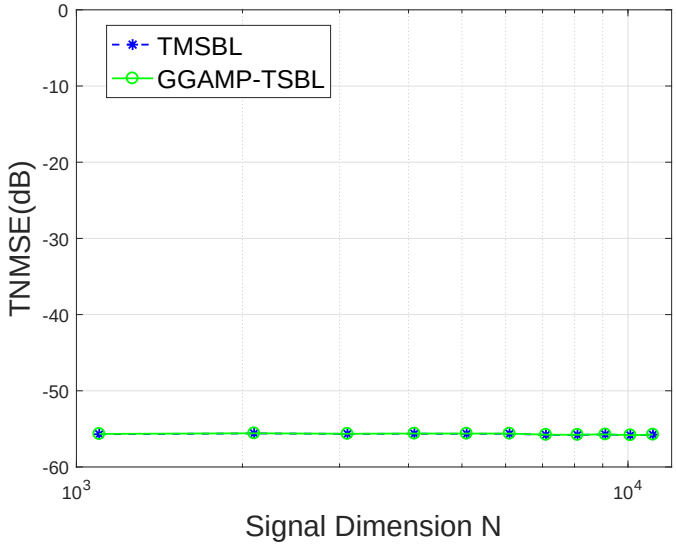
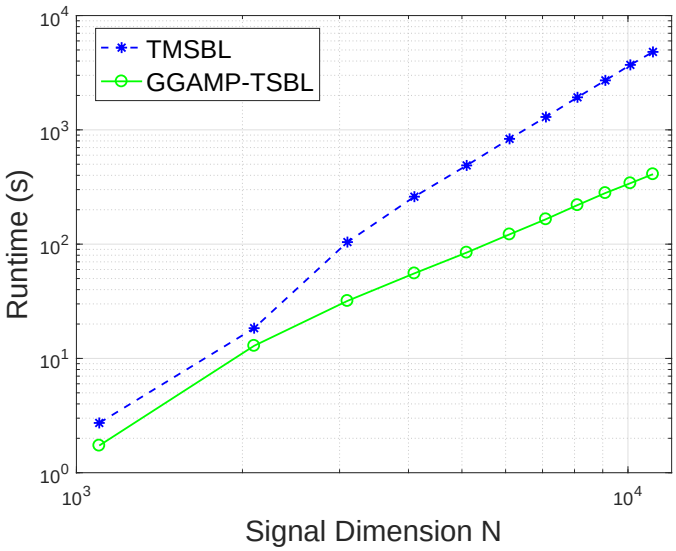


Figure 2.14: Runtime comparison of MMV algorithms under non-i.i.d.-Gaussian $\boldsymbol{A}$ matrices with SNR=30dB



(a) NMSE versus N



(b) Runtime versus N

Figure 2.15: Performance and complexity comparison for MMV algorithms versus problem dimensions

Performance and complexity versus problem dimension

To validate the claim that the proposed algorithm is more suited to deal with large scale problems we study the algorithms' performance and complexity against the signal dimension N. We keep an M/N ratio of 0.5, a K/N ratio of 0.2 and an SNR of 60dB. We run the experiment using column correlated matrices with $\rho = 0.9$. In addition, we set β to 0.9, high temporal correlation. In terms of performance, Fig. 2.15a shows that the proposed GGAMP-TSBL algorithm was able to match the performance of TMSBL. However, in terms of complexity, similar to the SMV case, Fig. 2.15b shows that the runtime difference becomes more significant as the problem size grows, making the GGAMP-SBL a better choice for large scale problems.

2.6 Conclusion

In this chapter, we presented a GAMP based SBL algorithm for solving the sparse signal recovery problem. SBL uses sparsity promoting priors on $\boldsymbol{x}$ that admit a Gaussian scale mixture representation. Because of the Gaussian embedding offered by the GSM class of priors, we were able to leverage the Gaussian GAMP algorithm along with it's convergence guarantees given in [52], when sufficient damping is used, to develop a reliable and fast algorithm. We numerically showed how this damped GGAMP implementation of the SBL algorithm also reduces the cost function of the original SBL approach. The algorithm was then extended to solve the MMV SSR problem in the case of generic $\boldsymbol{A}$ matrices and temporal correlation, using a similar GAMP based SBL approach. Numerical results show that both the SMV and MMV proposed algorithms were more robust to generic $\boldsymbol{A}$ matrices when compared to other AMP algorithms. In addition, numerical results also show the significant reduction in complexity the proposed algorithms offer over the original SBL and TMSBL algorithms, even when sufficient damping is used to slow down the updates to guarantee convergence. Therefore the proposed algorithms address the convergence limitations in AMP algorithms as well as the complexity challenges in traditional

SBL algorithms, while retaining the positive attributes namely the robustness of SBL to generic $\boldsymbol{A}$ matrices, and the low complexity of message passing algorithms.

2.7 Acknowledgment

This chapter, in full, is a reprint of material published in the article Maher Al-Shoukairi, Philip Schniter, and Bhaskar D. Rao, "A GAMP-based low complexity sparse Bayesian learning algorithm", IEEE Transactions on Signal Processing, 2017. I was the primary author and B. D. Rao supervised the research.

Chapter 3

A GAMP Based Algorithm with Hierarchical Priors for Recovering Non-Negative Sparse Signals

3.1 Introduction

We consider a constrained single measurement recovery problem, where the goal is to recover the unknown, non-negative and sparse vector $\boldsymbol{x} \in \mathbb{R}_+^N$ from $M \leq N$ noisy linear measurements $\boldsymbol{y} \in \mathbb{R}^M$:

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{e}, \tag{3.1}$$

where $\boldsymbol{A} \in \mathbb{R}^{M \times N}$ is a known measurement matrix and $\boldsymbol{e} \in \mathbb{R}^N$ is the additive noise modeled by $\boldsymbol{e} \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I})$. This problem arises in a number of applications, including imaging and non-negative matrix factorization among others. A number of techniques were previously proposed to address this problem, including greedy algorithms [4, 6], projected gradient descent [78] and

Bayesian techniques [79, 80]. In this chapter we focus on the Bayesian technique proposed in [80], where we compare it to the previously proposed Bayesian technique from [79], and illustrate some of the advantages of the former. The Bayesian formulation in [80] assumes a hierarchical prior on $\boldsymbol{x}$ and the prior is based on a rectified Gaussian scale mixture (RGSM). The formulation is similar to the sparse Bayesian learning (SBL) algorithm [31], which uses a Gaussian scale mixture hierarchical prior to address the sparse signal recovery (SSR) problem without the non-negativity constraint on $\boldsymbol{x}$. A number of alternative techniques are proposed in [80] to implement the inference algorithm, and in this chapter we focus on the technique based on the low complexity generalized approximate message passing algorithm (GAMP). Bayesian algorithms proposed in [79] impose direct sparsity promoting priors on $\boldsymbol{x}$ and also use GAMP to find the estimate of $\boldsymbol{x}$. Previous results on the unconstrained SSR problem, have observed that compared to other GAMP based techniques, using a hierarchical GSM prior with the GAMP algorithm simplified the derivation of the algorithm [67], and significantly improved convergence when the transformation matrix $\boldsymbol{A}$ was not i.i.d.-sub-Gaussian, which is a known limitation with AMP based algorithms [52–54]. In this chapter we observe that the previously shown advantages of derivation simplification and convergence improvement still apply in the non-negative constrained problem when an RGSM prior is used. In addition to that, we show how the RGSM prior allows to change the prior on $\boldsymbol{x}$ by changing the mixing density, which translates into a simple change in the overall algorithm. We finally show how extending the algorithm to address the multiple measurement vector (MMV) problem with common sparsity profile can also be achieved by a simple algorithm change.

3.1.1 Chapter's Organization

In section 3.2 we present the rectified GSM prior from [80]. In section 3.3 we revisit the GAMP algorithm in both its sum-product and max-sum versions. In section 3.4 we show how the prior on $\boldsymbol{x}$ can be changed by changing the mixing density of the scale mixture and introduce a

number of algorithms based on these priors. In section 3.5 we extend the approach to handle the NNLS MMV case. Finally, in section 3.6 we present numerical results to show the convergence and performance advantages the proposed approach can provide over existing algorithms.

3.2 Rectified Gaussian Scale Mixture Prior

Since it was shown that most super Gaussian priors that are sparsity promoting priors can be represented by Gaussian scale mixtures [45], Gaussian scale mixture priors have been successfully used in the general SSR problem [31]. Inspired by that success, [80] proposed the following rectified Gaussian scale mixture (RGSM) prior on $\boldsymbol{x}$ to address the non-negative SSR problem:

$$p(x_i|\gamma_i) = \mathcal{N}^R(x_i;0,\gamma_i), \tag{3.2}$$

$$\mathcal{N}^R(x_i;0,\gamma_i) = 2\mathcal{N}(x;0,\gamma_i)u(x_i), \tag{3.3}$$

$$p(x_i) = \int_0^\infty \mathcal{N}^R(x_i;0,\gamma_i)p(\gamma_i)d\gamma_i. \tag{3.4}$$

Where the prior on $\boldsymbol{x}$ is controlled by the choice of the mixing density $p(\gamma)$.

A type II framework is used to infer the hyper-parameters $\boldsymbol{\gamma}$, which is then used to obtain a point estimate of $\boldsymbol{x}$. The EM algorithm is chosen to estimate $\boldsymbol{\gamma}$, treating $\boldsymbol{x}$ as the hidden variable. Using the current estimate $\boldsymbol{\gamma}^n$, the E-step involves determining the conditional expectation of the complete data log-likelihood $Q(\boldsymbol{\gamma},\boldsymbol{\gamma}^n)$:

$$Q(\boldsymbol{\gamma},\boldsymbol{\gamma}^n) = E_{\boldsymbol{x}|\boldsymbol{y};\boldsymbol{\gamma}^n}\left[\ln p(\boldsymbol{y}|\boldsymbol{x}) + \ln p(\boldsymbol{x}|\boldsymbol{\gamma}) + \ln p(\boldsymbol{\gamma})\right] \tag{3.5}$$

$$\doteq \sum_{i=1}^{M} E_{\boldsymbol{x}|\boldsymbol{y};\boldsymbol{\gamma}^n}\left[-\frac{1}{2}\ln\gamma_i - \frac{x_i^2}{2\gamma_i} + \ln p(\gamma_i)\right] \tag{3.6}$$

In the M-step, $Q(\boldsymbol{\gamma},\boldsymbol{\gamma}^n)$ is maximized with respect to $\boldsymbol{\gamma}$ by taking the derivative and setting it equal

to zero. Note this maximization problem involves simple scalar decoupled optimization problems. In this chapter, we focus on the choice of using the GAMP algorithm to implement the E-step, where we aim to illustrate some of the advantages that using the RGSM prior within GAMP has over directly imposing a sparsity promoting prior on $\boldsymbol{x}$.

3.3 Generalized Approximate Message Passing (GAMP)

In this section we revisit the GAMP algorithm, and show how the E-step of the EM algorithm, mentioned in the previous section, can be implemented using GAMP. Approximate message passing algorithms use Gaussian and quadratic approximations to achieve low complexity. When given the the prior $p(\boldsymbol{x})$ and the likelihood $p(\boldsymbol{y}|\boldsymbol{x})$, GAMP can approximate the minimum mean square error (MMSE) estimate when its product-sum mode is implemented, or GAMP is able to approximate the MAP estimate of $\boldsymbol{x}$ when its max-sum version is implemented. In its sum-product version, GAMP approximates the marginal posteriors as follows:

$$p(x_i|r_i;\tau_{r_i}) \propto p(x_i)\mathcal{N}(x_i;r_i,\tau_{r_i}), \tag{3.7}$$

where in (3.7) r_i approximates an AWGN corrupted version of the true x_i, an approximation which becomes exact in the large system limit, when $\boldsymbol{A}$ is i.i.d sub-Gaussian [50, 81]:

$$r_i = x_i + \bar{r}_i \tag{3.8}$$

$$\bar{r}_i \sim \mathcal{N}(0,\tau_{r_i}). \tag{3.9}$$

Based on this approximation, in sum-product mode, GAMP sets the estimate $\hat{x}_i$ at the scalar MMSE estimate of x_i given r_i, and it sets the estimated variance of the marginal posteriors τ_{x_i} as

follows:

$$\hat{x}_i = \mathbb{E}\{x_i | r_i = r_i; \tau_{r_i}\} \tag{3.10}$$

$$\tau_{x_i} = \text{var}\{x_i | r_i = r_i; \tau_{r_i}\} \tag{3.11}$$

Applying the assumed rectified Gaussian scale mixture in (3.10) and (3.11) we find the first two moments of the approximate marginal posterior distributions using sum-product GAMP:

$$\hat{x}_i = \mathbb{E}\{x_i | r_i = r_i; \tau_{r_i}\} = \int_{x_i} p(x | r_i; \tau_{r_i}) \tag{3.12}$$

$$= \int_{+} x_i \mathcal{N}^R(x_i; 0, \gamma_i) \mathcal{N}(x_i, r_i, \tau_{r_i}) \tag{3.13}$$

using Gaussian pdf multiplication rule, and the mean of a rectified Gaussian

$$\hat{x}_i = \eta_i + \sqrt{\nu_i} h(\frac{\eta_i}{\nu_i}) \tag{3.14}$$

$$\eta_i = \frac{r_i \gamma_i}{\tau_{r_i} + \gamma_i} \tag{3.15}$$

$$\nu_i = \frac{\tau_{r_i} \gamma_i}{\tau_{r_i} + \gamma_i} \tag{3.16}$$

$$h(a) = \frac{\varphi(a)}{\Phi_c(a)}, \tag{3.17}$$

where φ refers to the pdf and Φ_c refers to the complimentary cdf.

$$\tau_{x_i} = var\{x_i | r_i = r_i; \tau_{r_i}\} = \int_{x_i} x_i^2 p(x | r_i; \tau_{r_i}) - \hat{x}_i^2 \tag{3.18}$$

$$= \int_{+} x_i^2 \mathcal{N}^R(x_i; 0, \gamma_i) \mathcal{N}(x_i, r_i, \tau_{r_i}) - \hat{x}_i^2 \tag{3.19}$$

using the Gaussian pdf multiplication rule, and the variance of a rectified Gaussian:

$$\tau_{x_i} = \nu_i g(\frac{\eta_i}{\nu_i}) \tag{3.20}$$

$$g(a) = 1 - h(a)\,(h(a) - a)\,. \tag{3.21}$$

In the max-sum version, GAMP sets $\hat{x}_i$ at the scalar MAP estimate given r_i using the proximal operator, and it sets the parameter τ_{x_i} as follows:

$$\hat{x}_i = \text{prox}_{-\ln p(x_i)}(r_i; \tau_{r_i}) \tag{3.22}$$

$$\tau_{x_i} = \tau_r \text{prox}'_{-\ln p(x_i)}(r_i; \tau_{r_i}) = \frac{\tau_{r_i}}{1 - (-\ln p(x_i))''} \tag{3.23}$$

$$\text{prox}_f(\hat{a}, \tau^a) \triangleq \underset{x \in \boldsymbol{R}}{\text{argmin}}\, f(x) + \frac{1}{2\tau^a}|x - \hat{a}|^2 \tag{3.24}$$

In the case of max-sum GAMP implementation, we evaluate (3.22) and (3.23) with the assumed prior on $\boldsymbol{x}$:

$$\hat{x}_i = \underset{x \geq 0}{\text{argmin}}\, \frac{1}{2\gamma_i} + \frac{1}{2\tau_{r_i}}|x - r_i| \tag{3.25}$$

$$\hat{x}_i = \begin{cases} \frac{r_i \gamma_i}{\tau_{r_i} + \gamma_i} = \eta_i & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases} \tag{3.26}$$

$$\tau_{x_i} = \frac{\tau_{r_i} \gamma_i}{\tau_{r_i} + \gamma_i} = \nu_i \tag{3.27}$$

We point out here, that the computations we just showed in both the sum-product and max-sum modes are relatively simple and generally readily available from previous results, unlike the more involved computations typically needed for algorithms that impose non-negative direct sparsity promoting priors.

The approximate marginal distributions can be obtained upon convergence of the max-

sum GAMP, by computing (3.14) and (3.20). As shown in [80], there is no closed form for the posterior distribution based on the RGSM prior, therefore one option is to use the the approximate posterior computed by GAMP in its sum-product version. However in this chapter we would like to use GAMP in its max-sum mode, where GAMP does not provide marginal distribution approximations. Nonetheless, an extra step can be added to compute marginal distributions using (3.14) and (3.20), and the computed marginals can be used in the E-step, which is similar to the approach in [79]. This step is valid, because the assumption that r_i is an AWGN corrupted version of the true $\boldsymbol{x}$ still holds in the max-sum case, in the large system limit and under i.i.d sub-Gaussian $\boldsymbol{A}$ [81].

As we mentioned, we focus in this chapter on the max-sum version of GAMP, where we use it to find the approximate MAP estimate of $\boldsymbol{x}$, and to implement the E-step as described above. The GAMP portion of the algorithm is stated under the E-step portion of Table 3.1, where we note here that we are using the version of GAMP from [52] that uses damping to enhance convergence, where θ_s is the damping factor. We emphasize here that since $\boldsymbol{\gamma}$ is fixed during the E-step, the E-step is not affected by the chosen mixing density $p(\boldsymbol{\gamma})$, and remains unchanged when the choice of $p(\boldsymbol{\gamma})$ is changed.

3.4 Examples of Mixing Densities

In this section we demonstrate the flexibility of the RGSM framework by showing how different priors can be imposed on $\boldsymbol{x}$ by changing the mixing density $p(\boldsymbol{\gamma})$, resulting in a change only to the M-step.

RGGAMP-SBL

For the first example choose a non-informative prior on $\boldsymbol{\gamma}$, this choice results in an algorithm that is equivalent to the algorithm in [80] when implemented using GAMP. In [80] the

algorithm with non-informative prior was named rectified sparse Bayesian learning (RSBL). A number of differernt techniques were used to implement the E-step of the algorithm including GAMP. Here we would like to refer to the algorithm using a convention similar to [66], where we call the algorithm rectified Gaussian GAMP (RGGAMP-SBL). For the non informative prior on $\boldsymbol{\gamma}$ the M-step is:

$$\gamma_i^{n+1} = \hat{x_i}^2 + \tau_{x_i} \tag{3.28}$$

Where as mentioned in the previous section $\hat{x_i}$ and τ_{x_i} are obtained using (3.14) and (3.20).

RGGAMP-LASSO

For the second example, we choose to demonstrate how the non-negative LASSO algorithm can be implemented using this framework. The following mixing density is chosen for $p(\gamma)$:

$$p(\gamma) = \frac{\lambda^2}{2} e^{-\frac{\lambda^2 \gamma}{2}} u(\gamma) \tag{3.29}$$

Based on this selection the effective prior on $\boldsymbol{x}$ becomes a rectified Laplacian:

$$\begin{aligned} p(x) =& 2u(x) \int_0^\infty \mathcal{N}(x|0,\gamma_i) \frac{\lambda^2}{2} e^{-\frac{\lambda^2 \gamma}{2}} u(\gamma) d\gamma \\ =& \lambda e^{-\lambda x} u(x). \end{aligned}$$

It was previously shown in [79] and [80] that imposing a rectified Laplacian prior on $\boldsymbol{x}$ is equivalent to solving the non-negative LASSO problem. Based on that, the M-step updates for λ and γ are

be found to be the same as the GSM based LASSO from [39]:

$$\gamma_i^{n+1} = -1 + \sqrt{1 + 4\lambda^n(\hat{x}_i^2 + \tau_{x_i})} \tag{3.30}$$

$$\lambda^{n+1} = 2N / \sum_i \gamma_i^{n+1} \tag{3.31}$$

The full EM implementation of both algorithms is found in Table I

3.5 Multiple Measurement vector extension

In this section, we extend the SMV RGGAMP-SBL algorithm to the multiple measurement vectors (MMV) problem. The MMV model can be represented as an extension of the SMV model in (3.1) as follows:

$$\boldsymbol{y}^{(t)} = \boldsymbol{A}\boldsymbol{x}^{(t)} + \boldsymbol{e}^{(t)}, \;\; t = 1, 2...T \tag{3.32}$$

$$\boldsymbol{x}^{(t)} \in \mathbb{R}_+^{\mathbb{N}}, \; \boldsymbol{y}^{(t)} \in \mathbb{R}^{\mathbb{M}}, \tag{3.33}$$

where a common sparsity profile is assumed, meaning that all vectors share the same non-zero locations. No correlation is assumed between non-zero elements. The prior assumed on each element of the vector $\boldsymbol{x}^{(t)}$ is:

$$p(x_i^{(t)}|\gamma_i) = \mathcal{N}^R(x_i^{(t)}; 0, \gamma_i) \tag{3.34}$$

In this formulation, $\boldsymbol{\gamma}$ is the sparsity controlling parameter and is shared across t. To demonstrate how this problem can be solved based on the RGSM prior we employed, we include here the factor graphs of both the SMV and MMV cases in Fig. 3.1a and Fig. 3.1b, where in the factor graphs g_i corresponds to the likelihood function and f_i corresponds to the prior. The SMV factor graph corresponds to the E-step update implemented by GAMP, keeping in mind that during

Table 3.1: RGGAMP based algorithms

Define

$h(a) = \frac{\varphi(a)}{\Phi_c(a)}$,

$\varphi(a)$ is the pdf and $\Phi_c(a)$ is the complementary cdf of $\mathcal{N}(a;0,1)$

$g(a) = 1 - h(a)(h(a) - a)$

Initialization

$\boldsymbol{S} \leftarrow |\boldsymbol{A}|^2$ (component wise magnitude squared)

Initialize $\dot{\boldsymbol{\tau}}_x^0 > 0$

$\boldsymbol{s}^0, \dot{\boldsymbol{x}}^0 \leftarrow \boldsymbol{0}$

for $n = 1, 2, ..., N_{\max}$

  Initialize $\boldsymbol{\tau}_x^1 \leftarrow \dot{\boldsymbol{\tau}}_x^{n-1}, \boldsymbol{x}^1 \leftarrow \dot{\boldsymbol{x}}^{n-1}$

  E-Step approximation

  for $k = 1, 2, ..., K_{\max}$

    $1/\boldsymbol{\tau}_p^k \leftarrow \boldsymbol{S}\boldsymbol{\tau}_x^k$

    $\boldsymbol{p}^k \leftarrow \boldsymbol{s}^{k-1} + \boldsymbol{\tau}_p^k \boldsymbol{A}\boldsymbol{x}^k$

    $\boldsymbol{\tau}_s^k \leftarrow \frac{\sigma^{-2}\boldsymbol{\tau}_p^k}{\sigma^{-2}+\boldsymbol{\tau}_p^k}$

    $\boldsymbol{s}^k \leftarrow (1-\theta_s)\boldsymbol{s}^{k-1} + \theta_s(\boldsymbol{p}^k/\boldsymbol{\tau}_p^k - \boldsymbol{y})/(\sigma^2 + 1/\boldsymbol{\tau}_p^k)$

    $1/\boldsymbol{\tau}_r^k \leftarrow \boldsymbol{S}^\top \boldsymbol{\tau}_s^k$

    $\boldsymbol{r}^k \leftarrow \boldsymbol{x}^k - \boldsymbol{\tau}_r^k \boldsymbol{A}^\top \boldsymbol{s}^k$

    $\boldsymbol{\tau}_x^{k+1} \leftarrow \boldsymbol{\nu}^k$

    $\hat{\boldsymbol{x}}^{k+1} \leftarrow \boldsymbol{\eta}^k u(\boldsymbol{r}^k)$

    if $\|\hat{\boldsymbol{x}}^{k+1} - \hat{\boldsymbol{x}}^k\|^2 / \|\hat{\boldsymbol{x}}^k\|^2 < \varepsilon_{\text{gamp}}$, break

  end for %end of k loop

    $\dot{\boldsymbol{x}}^n \leftarrow \boldsymbol{\eta}^{k+1} + \sqrt{\boldsymbol{\nu}^{k+1}} h(\frac{\boldsymbol{\eta}^{k+1}}{\boldsymbol{\nu}^{k+1}})$, $\dot{\boldsymbol{\tau}}_x^n \leftarrow \boldsymbol{\nu}^{k+1} g(\frac{\boldsymbol{\eta}^{k+1}}{\boldsymbol{\nu}^{k+1}})$

  M-Step

  for GGAMP-RSBL

    $\boldsymbol{\gamma}^{n+1} \leftarrow |\dot{\boldsymbol{x}}^n|^2 + \dot{\boldsymbol{\tau}}_x^n$

  for GGAMP-Lasso

    $\boldsymbol{\gamma}^{n+1} \leftarrow \frac{-1+\sqrt{1+4\lambda^n(|\dot{\boldsymbol{x}}^n|^2 + \dot{\boldsymbol{\tau}}_x^n)}}{2\lambda^n}$

    $\lambda^{n+1} \leftarrow \frac{2N}{\sum_i \gamma_i^{n+1}}$

  if $\|\dot{\boldsymbol{x}}^n - \dot{\boldsymbol{x}}^{n-1}\|^2 / \|\dot{\boldsymbol{x}}^{n-1}\|^2 < \varepsilon_{\text{em}}$, break

end for %end of i loop

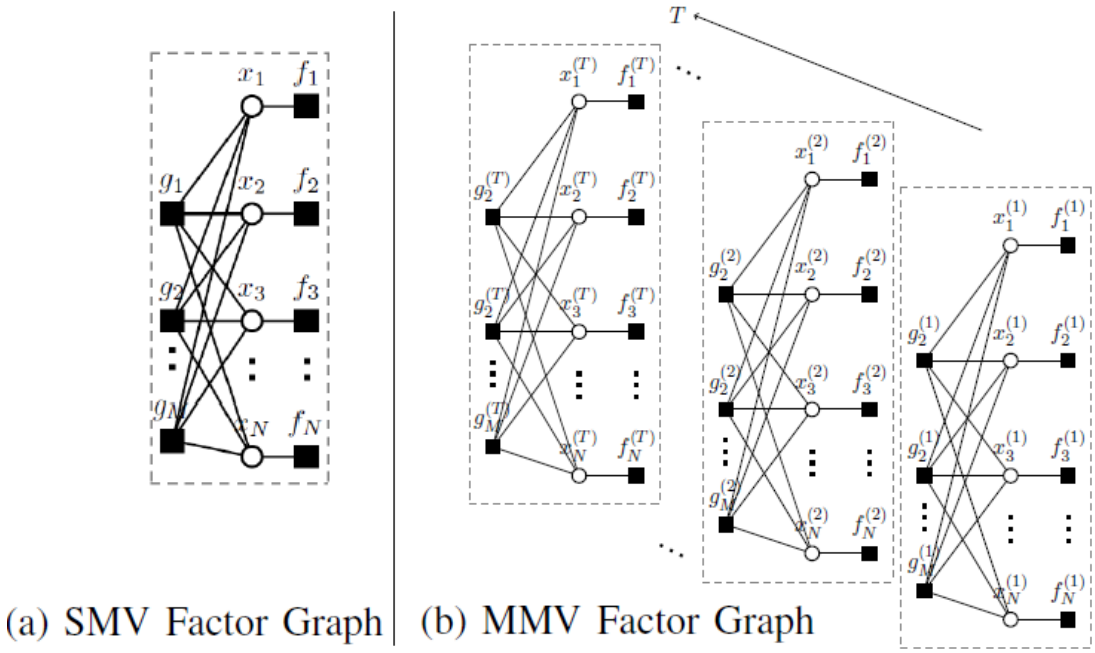


Figure 3.1: SMV Versus MMV factor graphs

the E-step $\boldsymbol{\gamma}$ is fixed and therefore is not included as a variable in the factor graph. When the SMV factor graph is extended to the MMV case, we clearly see that it corresponds to solving a number of independent E-step problems, since no connections between the individual factor graphs exist. We mention here, that using a Bernoulli-Gaussian prior to directly impose sparsity on $\boldsymbol{x}$, would have required introducing a new variable on the factor graph which represents the common support, and connects all SMV factor graphs together, which would add significant complexity to deriving the algorithm. Therefore, the RGSM framework provides a significant advantage in terms of simplicity. Assuming a non-informative prior on $\boldsymbol{\gamma}$, once the individual E-step estimates are obtained, the M-step can be implemented to update $\boldsymbol{\gamma}$ as follows:

$$\gamma_i^{n+1} = \frac{1}{T}\sum_{t=1}^{T} |\hat{x}_i^{(t)}|^2 + \tau_{x_i^{(t)}} \tag{3.35}$$

3.6 Numerical Results

SMV Numerical Results

In this section we conduct a numerical study to compare the performance and runtimes of the two SMV algorithms proposed in this chapter, and the algorithms previously proposed in [79], where all algorithms are implemented using GAMP. The study also aims to illustrate the convergence enhancement the RGSM hierarchical prior provides. The performance is studied through the normalized mean squared error (NMSE):

$$\text{NMSE} \triangleq \|\hat{\boldsymbol{x}} - \boldsymbol{x}\|^2 / \|\boldsymbol{x}\|^2$$

In the following we compare the two algorithms proposed in this chapter, namely RGGAMP-SBL and RGGAMP-LASSO in addition to the two algorithms from [79] which are the non-negative Gaussian mixture approximate message passing (NNGMAMP) and the non-negative LASSO approximate message passing (NNLAMP) algorithms. In all subsequent numerical experiments we chose $N = 1000$, $M = 500$ and the number of non-zero elements $K = 250$. The experiments were run over three different types on non-iid Gaussian matrices $\boldsymbol{A}$, which are a) column correlated, b) Low rank and c) Ill conditioned matrices. The details for how these matrices are built can be found in [54] and [66].

Setting the SNR to 60dB, Fig. 3.2 shows the NMSE for all of the four algorithms over a range of deviation from the i.i.d.-Gaussian $\boldsymbol{A}$. We can clearly see that the RGGAMP-SBL algorithm provides significant convergence improvement over the NNGMAMP algorithm as $\boldsymbol{A}$ deviates from the i.i.d-Gaussian case. Moreover, despite the slight degradation in performance RGGAMP-LASSO also shows better convergence properties. We also note here that the NNLAMP experiences improved convergence when compared to the NNGMAMP algorithm.

We next reduce the SNR to 30dB and repeat the experiments. We can see in Fig. 3.3 that RGGAMP-SBL still provides improved convergence compared to NNGMAMP. We also note

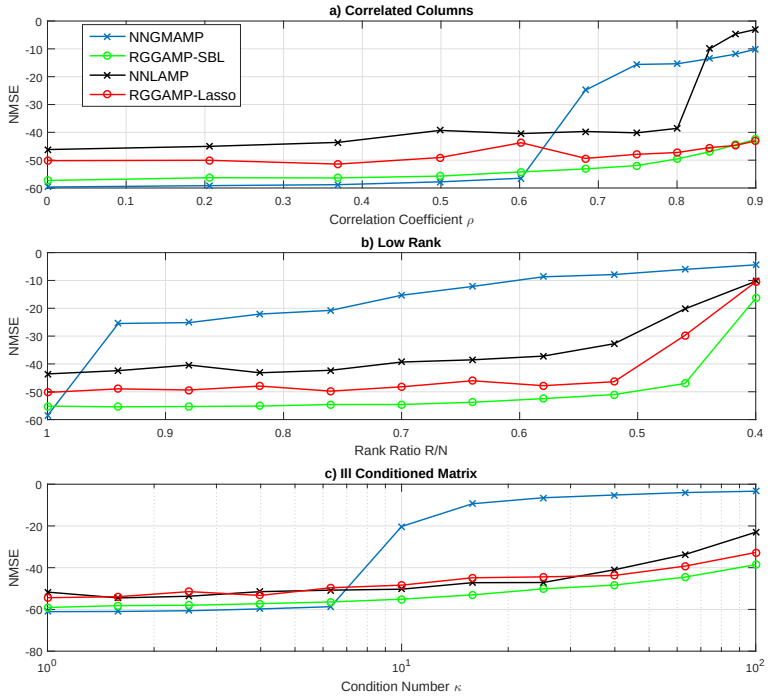


Figure 3.2: NMSE comparison of SMV algorithms under non-i.i.d.-Gaussian $\boldsymbol{A}$ matrices with SNR=60dB

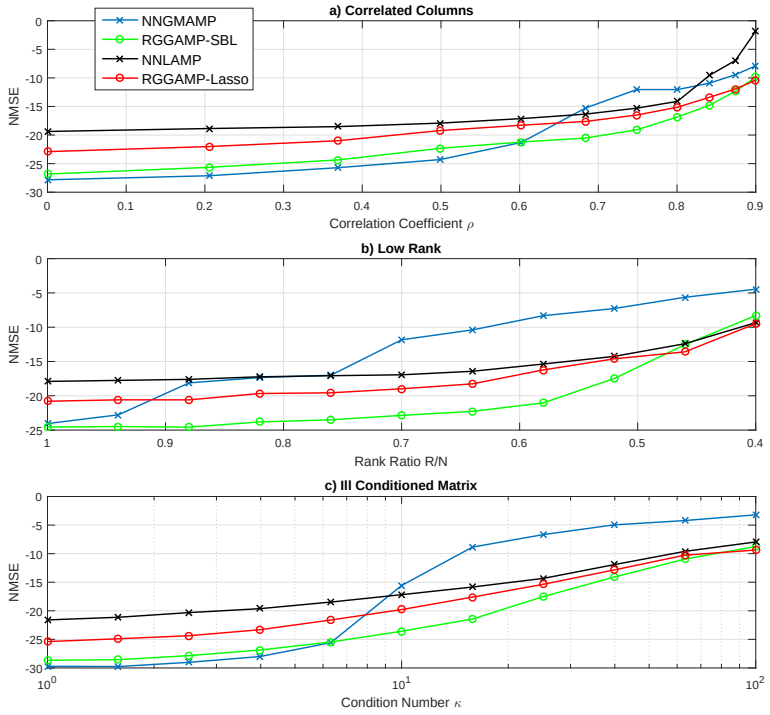


Figure 3.3: NMSE comparison of SMV algorithms under non-i.i.d.-Gaussian $\boldsymbol{A}$ matrices with SNR=30dB

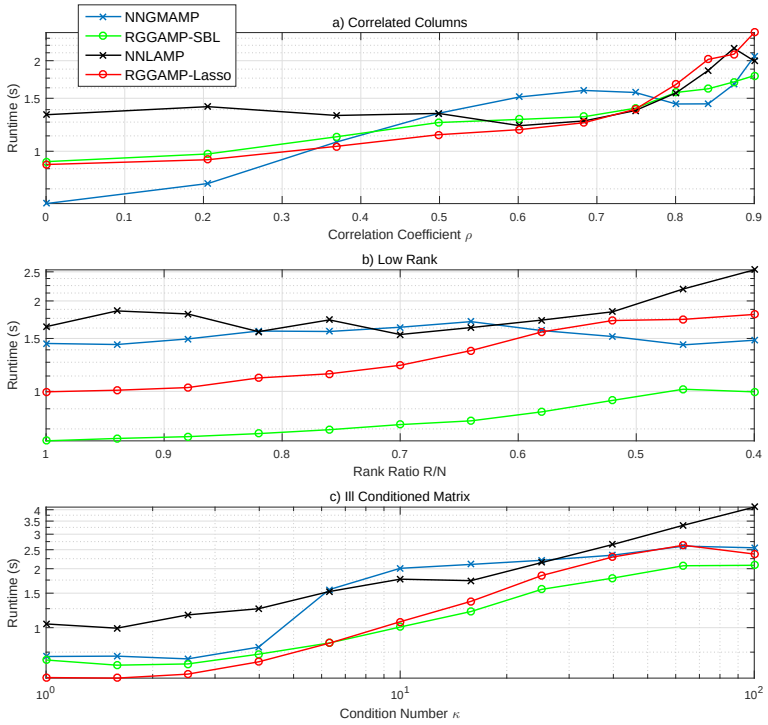


Figure 3.4: Runtime comparison of SMV algorithms under non-i.i.d.-Gaussian $\boldsymbol{A}$ matrices with SNR=30dB

that except for the NNGMAMP, all other algorithms seem to have close performance when the deviation from the i.i.d.-Gaussian case grows larger.

Finally, we present in Fig. 3.4 a comparison between the runtimes of different algorithms at 30dB SNR. The figure shows that although there are slight variations in runtimes, all algorithms have runtimes within the same range, which is expected since all algorithms are using the low complexity GAMP.

MMV Numerical Results

In the MMV case we compare the developed RGGAMP-MSBL MMV algorithm to the SMV algorithms, RGGAMP-SBL and NNGMAMP. The comparison aims to show the advantage of jointly using the common support information, instead of solving for each vector $\boldsymbol{x}^{(t)}$

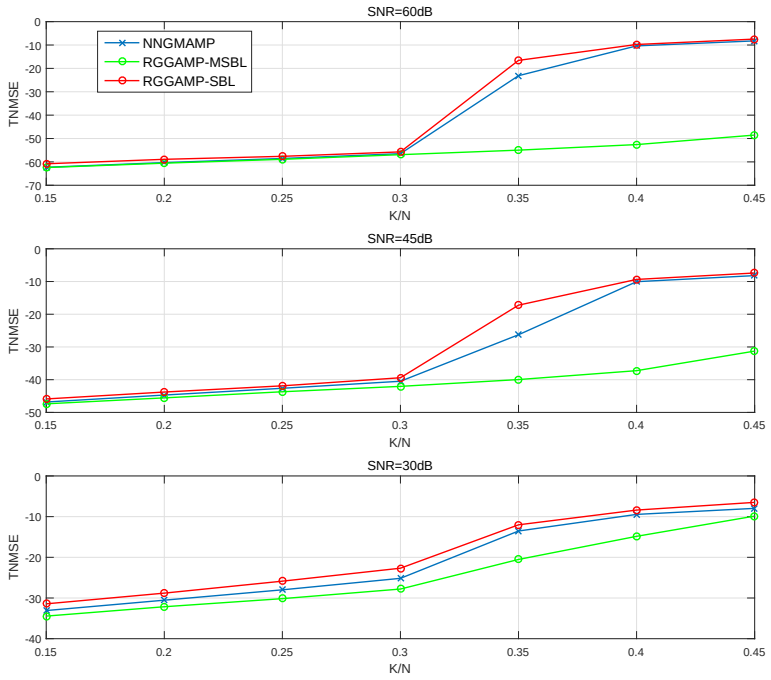


Figure 3.5: TNMSE comparison of the MMV algorithm and SMV algorithms

independently. While more comparison to existing MMV algorithms should be done, we leave that to future work, which should also include studying the case of having correlation between non-zero elements of $\boldsymbol{x}$, as was done before in the unconstrained case. We use time-averaged normalized mean squared error TNMSE as the performance measure in the MMV case:

$$\text{TNMSE} \triangleq \frac{1}{T}\sum_{t=1}^{T} \|\hat{\boldsymbol{x}}^{(t)} - \boldsymbol{x}^{(t)}\|^2 / \|\boldsymbol{x}^{(t)}\|^2$$

The experiments in Fig. 3.5 show the TNMSE for each algorithm against the sparsity ratio K/N, where K is the number of non-zero elements in $\boldsymbol{x}$. The experiments were performed over different values of SNR with $N = 1000$, $M = 500$ and $T = 5$. We use a randomly generated i.i.d.-Gaussian matrix $\boldsymbol{A}$ for each of the experiments. From the experiments, it is clear that RGGAMP-MSBL is able to provide significant improvement in performance as K/M grows

larger, where the improvement is more noticeable at higher SNRs. We re-emphasize here, that this improvement comes at a small price of modifying the M-step of the algorithm.

3.7 Conclusion

In this chapter we showed the advantages of considering a hierarchical rectified Gaussian scale mixture prior combined with the GAMP algorithm to solve the sparse non-negative least squares problem. We focused on comparing the approach against other GAMP based algorithms that directly impose a sparsity promoting prior on the unknown signal. We showed that the advantages include simpler algorithm development, better convergence properties when the transformation matrix is non-i.i.d.-Gaussian, the ability to change the prior on the unknown signal through a simple change in the algorithm and the ability to extend the algorithm to the MMV case through a simple change as well.

3.8 Acknowledgment

This chapter, in full, is a reprint of material published in the article Maher Al-Shoukairi, and Bhaskar D. Rao, "A GAMP based algorithm with hierarchical priors for recovering non-negative sparse signals", Asilomar Conference on Signals, Systems, and Computers, 2017. I was the primary author and B. D. Rao supervised the research.

Chapter 4

An MPDR Perspective and Extension of Sparse Bayesian Learning

4.1 Introduction

In this chapter we offer a novel interpretation of the SBL algorithm based on the minimum power distortionless response (MPDR) beamforming method in array processing. The SBL's iterations are interpreted as the MPDR beamformer (BF) iteratively applied to the measurements and the beamformer parameters updated based on the estimates at each iteration, along with the ability to add a denoising step at the output of the MPDR BF when a prior is imposed on $\boldsymbol{x}$. The potential of the new insight is demonstrated by extending the algorithm to incorporate priors other than GSM priors. We present a low complexity version of the algorithm by replacing the MPDR BF step with an approximate message passing (AMP) based MPDR implementation. We refer to this methodology as the MPDR-SSR framework. We provide an interpretation of the proposed algorithm's convergence, where we show the Gaussian nature of the output noise of the MPDR BF, and how the noise variance at the output of the MPDR BF and the output of the denoiser are reduced at each iteration. These two concepts provide important intuition into designing better

algorithms when more information about $\boldsymbol{x}$ is available.

4.1.1 Chapter's Organization

The organization of the chapter is as follows. In Section 4.2, we present the general idea of addressing the SSR problem from a source localization perspective. We then propose the MPDR-SSR framework and use it to derive the LSBL algorithm. In section 4.3 we list some denoising options based on different priors. In section 4.4 we numerically analyze the MPDR-SSR convergence. Finally, in Section 4.5, we present numerical results to show the benefits of the algorithms that we proposed in the light of the new understanding of SBL algorithm.

4.2 SSR as a Source Localization Problem

In this section we first restate the SSR problem and re-summarize the SBL algorithm for completeness. We then present our main idea of treating the SSR problem as a source localization one. We will rewrite the SBL updates to provide a new perspective of the algorithm based on an MPDR interpretation. Using this new perspective we will extend the algorithm to a more general SSR framework that can accept non-GSM priors that do not have closed form posteriors.

4.2.1 SSR problem and SBL Algorithm Summary

We start by restating the SSR problem and re-summarizing the SBL algorithm for completeness. The single measurement vector (SMV) sparse signal recovery (SSR) problem is formulated as:

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{e}, \tag{4.1}$$

where $\boldsymbol{y} \in \mathbb{R}^M$ is the observed measurement vector, $\boldsymbol{x} \in \mathbb{R}^N$ is the sparse vector to be recovered from the measurement vector, $\boldsymbol{A} \in \mathbb{R}^{M \times N}$ is a known transformation measurement matrix and $\boldsymbol{e} \in \mathbb{R}^M$ is additive Gaussian noise that can be modeled by $\boldsymbol{e} \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I})$, where in this chapter we will assume that σ^2 is known. Using a Gaussian scale mixture, the SBL algorithm imposes a separable sparsity promoting prior on $\boldsymbol{x}$, i.e. $p(\boldsymbol{x}) = \Pi p(x_i)$. The prior of each element x_i is a zero mean Gaussian with the hyperparameter γ_i as its variance, $p(x_i) = \mathcal{N}(x_i; 0, \gamma_i)$. The chosen prior on γ_i specifies the mixing density and therefore determines the effective prior on $\boldsymbol{x}$:

$$p(x_i) = \int \mathcal{N}(x_i; 0, \gamma_i) p(\gamma_i) d\gamma_i, \tag{4.2}$$

It was shown in [45] that a number of sparsity promoting priors can be imposed by a GSM prior, when using a proper mixing density $p(\gamma_i)$. The goal of the algorithm is to estimate the hyperparameter vector $\boldsymbol{\gamma} = [\gamma_1, \ldots, \gamma_N]^T$, which can be achieved using evidence maximization. Once an estimate $\hat{\boldsymbol{\gamma}}$ is obtained, it can be used to obtain an approximation for the posterior $p(\boldsymbol{x}|\boldsymbol{y})$ by using $p(\boldsymbol{x}|\boldsymbol{y}; \hat{\boldsymbol{\gamma}})$. One of the reasons this GSM prior is unique and desirable is that the posterior has a closed form solution. One option to iteratively learn the hyperparameter vector $\boldsymbol{\gamma}$ is to use the EM algorithm resulting in the EM-SBL [31]. To obtain an estimate of the hyperparameter, EM-SBL uses the EM algorithm to minimize the SBL's cost function given by [30, 31]:

$$\begin{aligned} \chi(\boldsymbol{\gamma}) &= -\log p(\boldsymbol{y}, \boldsymbol{\gamma}) = \frac{1}{2} \log |\boldsymbol{\Sigma}_y| + \frac{1}{2} \boldsymbol{y}^\top \boldsymbol{\Sigma}_y^{-1} \boldsymbol{y} - \log p(\boldsymbol{\gamma}) \\ \boldsymbol{\Sigma}_y &= \sigma^2 \boldsymbol{I} + \boldsymbol{A} \boldsymbol{\Gamma} \boldsymbol{A}^\top, \;\; \boldsymbol{\Gamma} \triangleq \mathrm{Diag}(\boldsymbol{\gamma}). \end{aligned} \tag{4.3}$$

$\boldsymbol{x}$ is treated as the hidden variable in EM-SBL and the hyperparameter update becomes:

$$\boldsymbol{\gamma}^{l+1} = \underset{\boldsymbol{\gamma}}{\mathrm{argmax}} \, \mathbb{E}_{\boldsymbol{x}|\boldsymbol{y}; \boldsymbol{\gamma}^l} \left[\log p(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{\gamma}) \right], \tag{4.4}$$

where $p(\boldsymbol{y},\boldsymbol{x},\boldsymbol{\gamma})$ is the joint probability of the complete data and $p(\boldsymbol{x}|\boldsymbol{y};\boldsymbol{\gamma}^{l})$ is the posterior under the current hyperparameter estimate $\boldsymbol{\gamma}^{l}$. Each iteration constitutes of an expectation step (E-step) followed by a maximization step (M-step) and the algorithm iterates between the two until convergence. In the following we will summarize the EM-SBL algorithm's steps, and we will rewrite them in subsequent subsections to provide an MPDR interpretation.

SBL's E-step Based on the system model in (4.1), the likelihood function $p(\boldsymbol{y}|\boldsymbol{x};\sigma^2)$ is given by:

$$p(\boldsymbol{y}|\boldsymbol{x};\sigma^2) = \frac{1}{(2\pi\sigma^2)^{\frac{M}{2}}} \exp\left(-\frac{1}{2\sigma^2}\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\|^2\right). \tag{4.5}$$

Due to the GSM prior, for a given value of the hyperparameter $\boldsymbol{\gamma}$ the posterior $p(\boldsymbol{x}|\boldsymbol{y},\boldsymbol{\gamma})$ is Gaussian, and is defined by its mean and variance given by:

$$p(\boldsymbol{x}|\boldsymbol{y},\boldsymbol{\gamma}) = \mathcal{N}(\boldsymbol{x};\hat{\boldsymbol{x}},\boldsymbol{\Sigma}_{\boldsymbol{x}}) \tag{4.6}$$

$$\hat{\boldsymbol{x}} = \boldsymbol{\Gamma}\boldsymbol{A}^{\top}\boldsymbol{\Sigma}_{y}^{-1}\boldsymbol{y}, \;\; \boldsymbol{\Sigma}_{\boldsymbol{x}} = \boldsymbol{\Gamma} - \boldsymbol{\Gamma}\boldsymbol{A}^{\top}\boldsymbol{\Sigma}_{y}^{-1}\boldsymbol{A}\boldsymbol{\Gamma}. \tag{4.7}$$

We will refer to the posterior mean by $\hat{\boldsymbol{x}}$ and we will use it as the algorithm's point estimate of $\boldsymbol{x}$. We will also use $\boldsymbol{\tau}_x$ to refer to the vector of the diagonal entries of the covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{x}}$. The mean and variance defining the posterior $p(\boldsymbol{x}|\boldsymbol{y},\boldsymbol{\gamma})$ will be used to execute the M-step.

SBL's M-Step: The M-step carried out by minimizing $\mathbb{E}_{\boldsymbol{x}|\boldsymbol{y};\boldsymbol{\gamma}^{l},\sigma^2}\left[-\log p(\boldsymbol{y},\boldsymbol{x},\boldsymbol{\gamma};\sigma^2)\right]$ over $\boldsymbol{\gamma}$ where:

$$\log p(\boldsymbol{y},\boldsymbol{x},\boldsymbol{\gamma};\sigma^2) = \log p(\boldsymbol{y}|\boldsymbol{x};\sigma^2) + \log p(\boldsymbol{x}|\boldsymbol{\gamma}) + \log p(\boldsymbol{\gamma}) \tag{4.8}$$

For the rest of this chapter we will assume that a non-informative prior is imposed on $p(\gamma_i)$. Since

$\log p(\boldsymbol{y}|\boldsymbol{x})$ does not depend on $\boldsymbol{\gamma}$ and $p(\boldsymbol{\gamma})$ is a non informative prior, the M-step becomes:

$$\textbf{(EM-SBL)}\ \ \gamma_i^{l+1} = \underset{\gamma_i}{\operatorname{argmin}} \left[\frac{\hat{x}_i^2 + \tau_{x_i}}{2\gamma_i} + \frac{1}{2}\log(\gamma_i) \right] = \hat{x}_i^2 + \tau_{x_i}. \tag{4.9}$$

where $\hat{x}_i$ and τ_{x_i} depend on $\boldsymbol{\gamma}^l$ and computed using Equation (4.7).

Another option for estimating $\boldsymbol{\gamma}$ is finding a fixed point iteration by the differentiation of (4.3) and equating to zero. This leads to the algorithm from [30], referred to as FP-SBL, with the following update:

$$\textbf{(FP-SBL)}\ \ \gamma_i^{l+1} = \underset{\gamma_i}{\operatorname{argmin}}[\chi(\boldsymbol{\gamma})] = \gamma_i^l \hat{x}_i^2 / (\gamma_i^l - \tau_{x_i}). \tag{4.10}$$

Where (4.10) assumes a non-informative prior on $\boldsymbol{\gamma}$ as well. We will drop the dependency of $\boldsymbol{\gamma}$ on the iteration l in the sequel for brevity.

4.2.2 A Source Localization Perspective of the SSR Problem

Applying the SSR algorithms to address the source localization problem was previously proposed in papers like [11]. In this section however, we present the novel idea of interpreting the SSR SBL algorithm from a source localization perspective. Compared to the pure Bayesian formulation of the SSR problem above, the proposed MPDR framework provides a deeper and more intuitive understanding of the algorithm's intermediate steps, allowing for improving and expanding the algorithm based on this understanding.

The idea is to treat the non-zero elements of $\boldsymbol{x}$ as unknown source signals. In the direction of arrival problem in array processing, each column of $\boldsymbol{A}$ is determined by the array manifold and is computed using an incoming source angle and the array geometry. In the general SSR formulation, the matrix $\boldsymbol{A}$ can be thought of sampling a fictitious array manifold, where each column of $\boldsymbol{A}$ is associated with a fictitious location. $\boldsymbol{A}$ does not necessarily have structure, and the

locations associated with the columns will not always have a physical interpretation. However we can still apply the source localization concepts and benefit from the physical intuition of array processing even for general $\boldsymbol{A}$ matrices as we will demonstrate in this chapter. Based on the source localization interpretation, one can iteratively construct a linear receiver or BF to estimate these unknown sources or non-zeros elements. In the SSR problem, the linear receiver or BF build for source localization will not have very good performance at first. However, based on the sparsity of the original vector $\boldsymbol{x}$, the output of the estimator can be further improved by proper soft thresholding [82]. The choice of linear receiver/BF should allow for this new estimate of $\boldsymbol{x}$ to be used to adaptively improve the receiver. This improved receiver's output can be used to get an even better estimate of $\boldsymbol{x}$, which will be used to improve the receiver further, and so on until convergence.

We point out that the stagewise orthogonal matching pursuit (StOMP) algorithm [82] uses this concept, where it applies a matched filter to the measurement as the choice of its linear receiver/BF. The output of the matched filter represents a noise corrupted version of $\boldsymbol{x}$ where each element experiences interference from the other non-zero elements of $\boldsymbol{x}$. The algorithm estimates the largest non-zero entry out of the matched filter. Based on this estimate, the matched filter output is improved by removing the effect of the detected non-zero element from the measurement vector. At the output of the matched filter the second largest non-zero element can now be detected given that the interference from the largest element is removed. This process is repeated until all the non-zero elements are recovered. In [82] the connection of (StOMP) is made to the multiple user communication systems, and the interference at the output of the matched filter is referred to as the multiple access interference (MAI), where the process of estimating elements and removing them is analogous to the successive interference cancellation in multi-user communications.

Rather than using a matched filter, in this chapter we focus on a class of sparse signal recovery algorithms associated with the MPDR/MVDR estimators and involves joint estimation of sources. The approach is inspired by a novel understanding of the SBL algorithm, where instead

of the posterior evaluation using Bayesian techniques, a multi-step array processing analysis of the algorithm is proposed. The MPDR/MVDR algorithms depend on the covariance matrix of the measurements which can be modeled using the powers of the elements of $\boldsymbol{x}$. Therefore instead of using successive interference cancellation like StOMP, we can use estimates of the powers of $\boldsymbol{x}$ to improve the receivers/BFs at each iteration as we will show in the following sections.

4.2.3 Iterative Array Processing

By definition, when detecting x_i the MPDR receiver minimizes the output variance of the linear BF while leaving x_i undistorted. This is expressed by the following optimization:

$$\underset{\boldsymbol{w}_i}{\operatorname{argmin}} \boldsymbol{w}_i^\top \boldsymbol{\Sigma}_y \boldsymbol{w}_i \quad \text{s.t. } \boldsymbol{w}_i^\top \boldsymbol{a}_{.i} = 1, \tag{4.11}$$

where $\boldsymbol{\Sigma}_y$ is the measurement covariance matrix and $\boldsymbol{a}_{.i}$ is the column i of the matrix $\boldsymbol{A}$. Using Lagrange multipliers to solve (4.11), the MPDR receiver is found to be [68]:

$$\boldsymbol{w}_i = \frac{\boldsymbol{\Sigma}_y^{-1} \boldsymbol{a}_{.i}}{(\boldsymbol{a}_{.i}^\top \boldsymbol{\Sigma}_y^{-1} \boldsymbol{a}_{.i})} \tag{4.12}$$

$$r_i = \boldsymbol{w}_i^\top \boldsymbol{y}, \ \hat{\tau}_{r_i} = 1/(\boldsymbol{a}_{.i}^\top \boldsymbol{\Sigma}_y^{-1} \boldsymbol{a}_{.i}) \tag{4.13}$$

where $\boldsymbol{r}$ is the output vector of the separate MPDR BFs utilized to estimate all sources, and $\hat{\boldsymbol{\tau}}_r$ is the vector of the corresponding total output variance. In the case that $\boldsymbol{\Sigma}_y$ is unknown, one can use a model for the measurement covariance matrix based on (4.1):

$$\boldsymbol{\Sigma}_y = (\boldsymbol{A}\boldsymbol{\Gamma}\boldsymbol{A}^\top + \sigma^2 \boldsymbol{I}). \tag{4.14}$$

We reemphasize that γ_is are the diagonal elements of $\boldsymbol{\Gamma}$ and they represent the powers of x_is. In this model, the powers γ_i are not known, and the closer they are to the true x_i powers the better

the performance of the MPDR BF becomes. Based on this model, we define the output noise variance associated with each element r_i as (total power - signal power):

$$\tau_{r_i} = 1/(\boldsymbol{a}_{.i}^{\top}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{a}_{.i}) - \gamma_i \tag{4.15}$$

Previous work has suggested iteratively learning these powers using optimization techniques [83]. However, this work did not go as far as making the connection to Bayesian algorithms, and it is missing the important denoising step that is based on the prior on $\boldsymbol{x}$. In the following sections we will show how the SBL algorithm provides a framework for learning the covariance matrix model in (4.14) by learning $\boldsymbol{\Gamma}$, and how it can be extended to incorporate different priors on $\boldsymbol{x}$.

4.2.4 An MPDR perspective of SBL

We now proceed to rewrite the equations of the EM-SBL algorithm in element wise form, then we rewrite them in a form that allows us to see a new angle of the SBL algorithm:

$$\begin{aligned}\hat{x}_i &= \gamma_i \boldsymbol{a}_{.i}^{\top}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{y} = \frac{\gamma_i \boldsymbol{a}_{.i}^{\top}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{y}(\boldsymbol{a}_{.i}^{\top}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{a}_{.i})}{(\boldsymbol{a}_{.i}^{\top}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{a}_{.i})} \\ &= \frac{\boldsymbol{w}_i^{\top}\boldsymbol{y}\gamma_i}{(\frac{1}{\boldsymbol{a}_{.i}^{\top}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{a}_{.i}} - \gamma_i) + \gamma_i} = \frac{r_i\gamma_i}{\tau_{r_i} + \gamma_i} \end{aligned} \tag{4.16}$$

$$\begin{aligned}\tau_{x_i} &= \gamma_i - \gamma_i^2 \boldsymbol{a}_{.i}^{\top}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{a}_{.i} = \boldsymbol{a}_{.i}^{\top}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{a}_{.i}(\frac{\gamma_i}{\boldsymbol{a}_{.i}^{\top}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{a}_{.i}} - \gamma_i^2) \\ &= \frac{\gamma_i(\frac{1}{\boldsymbol{a}_{.i}^{\top}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{a}_{.i}} - \gamma_i)}{(\frac{1}{\boldsymbol{a}_{.i}^{\top}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{a}_{.i}} - \gamma_i) + \gamma_i} = \frac{\gamma_i\tau_{r_i}}{\gamma_i + \tau_{r_i}} \end{aligned} \tag{4.17}$$

Now that we have rewritten the SBL's mean and variance estimates we can easily reformulate the SBL's iterations into the following three steps. The steps are detailed in Fig. 4.1, the steps of the algorithm are also explained below.

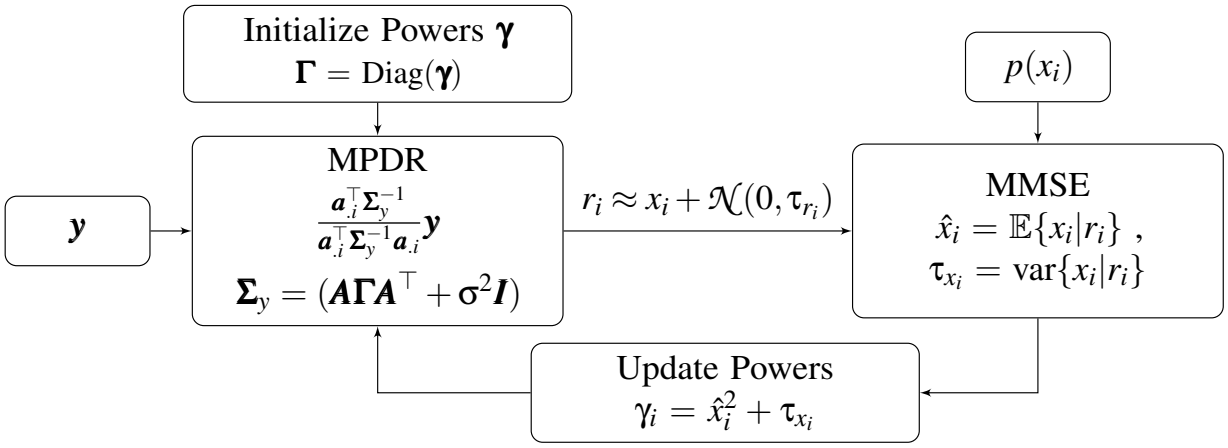


Figure 4.1: MPDR SSR Algorithm

A) MPDR Estimation: The algorithm builds an MPDR receiver with a measurement covariance matrix $\boldsymbol{\Sigma}_y$ modeled as (4.14), since the powers of x_i are unknown they are initialized to the same value, typically all ones. The MPDR receiver is then applied to the measurement vector, where based on the MPDR's definition the output is the desired x_i plus noise,

$$\boldsymbol{w}_i^\top \boldsymbol{y} = r_i = x_i + v_i. \tag{4.18}$$

With enough number of interfering sources (columns of $\boldsymbol{A}$), this noise can be justifiably modeled as Gaussian, $v_i \approx \mathcal{N}(0, \tau_{r_i})$, where τ_{r_i} is given by (4.15). Previous work had shown that for linear receivers the output noise corrupting the signal is Gaussian in the large system limit. In the next section we will list some of the references and verify this claim as part of the convergence analysis of our algorithm.

B) MMSE Estimation: Based on the AWGN assumption, the algorithm carries out an MMSE estimate of x_i using the prior $p(x_i)$. In the case of the SBL, equations (4.16) and (4.17) show this MMSE step based on r_i and τ_{r_i} and the GSM prior with a non-informative prior on $p(\boldsymbol{\gamma})$. The MMSE step enhances the estimate of x_i.

C) Model Update: In this step the algorithm updates the powers of $\boldsymbol{x}$ using the MMSE

estimate $\gamma_i = \mathbb{E}[x_i^2] = \hat{x}_i^2 + \tau_{x_i}$. Now that we have improved our power estimates, the output of the updated MPDR will have lower noise variance which is reduced even further by the MMSE step, which in turn results in more improvement of the MPDR and so on.

This interpretation of the SBL algorithm provides significant improvement in the understanding of the algorithm's iterations, an understanding that goes beyond the mathematical reduction of a cost function. The breakdown of the E-step of the algorithm into an MPDR and a de-noising step allows for improvements and potential extensions of each of these steps based on its function. Some of the enhancements can include:

1- Generalizing the algorithm beyond GSM priors, because it is possible to compute the MMSE estimate for an AWGN corrupted version of the decoupled x_is for a wide range of priors. Based on this flexibility in the MMSE step we present an MPDR based SSR framework that we refer to by MPDR-SSR. MPDR-SSR uses the algorithm from Fig. 4.1 with the choice of sparsity promoting prior. The prior can reflect restrictions on $\boldsymbol{x}$, like a non-negative constraint. It can reflect more information, like knowing the exact generating prior of $\boldsymbol{x}$. It can be due to choosing a different modeling prior like a non-identically distributed Laplace prior, which did not previously have a way to be evaluated directly and was solved by modeling it using a GSM [39, 84]. In the following subsections we will provide details of the MMSE step for a Laplace prior, we also apply some of the previously used priors in the MMSE step to produce different SSR algorithms.

2- If the imposed prior on $\boldsymbol{x}$ has some unknown hyperparameters, they can be updated at each iteration applying the EM algorithm and using the approximate posterior $\mathbb{E}[\boldsymbol{x}|\boldsymbol{r}]$. The same approach is also used to learn any unknown parameters of the problem in the case that they are not given beforehand.

3- The MMSE step can be replaced by a general noise filtering step, it can apply known successful denoising filters, like the previously developed image filters for example.

4.2.5 An MPDR perspective of FP-SBL

Although our main focus in this chapter is the EM-SBL algorithm, we also provide an MPDR based interpretation of the FP-SBL algorithm (4.10) to illustrate that variants can also understood intuitively based on this work. We rewrite the γ_i update from (4.10) in terms of the MPDR outputs given in Eqs. (4.16) and (4.17) as follows:

$$\gamma_i \leftarrow \gamma_i \frac{(\boldsymbol{a}_{.i}^{\top}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{y})^2}{\boldsymbol{a}_{.i}^{\top}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{a}_{.i}} = \gamma_i \frac{(\boldsymbol{a}_{.i}^{\top}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{y})^2/(\boldsymbol{a}_{.i}^{\top}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{a}_{.i})^2}{1/(\boldsymbol{a}_{.i}^{\top}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{a}_{.i})}. \tag{4.19}$$

Examining (4.19) we point out that the numerator represents the measured instantaneous output power of the MPDR BF, while the denominator represents the expected output power. The algorithm adjusts the current estimate of γ_i by the ratio of these two values. When examining a zero element x_i if its power in the model γ_i is non-zero, the measured output power of the MPDR will likely be lower than the expected power based on γ_i and therefore γ_i will be reduced at each iteration until it reaches zero. If x_i is a non-zero element, then its value will be adjusted at each iteration until the measured and expected powers are equal and iteration for γ_i converges. Other variants of the SBL algorithm that have been developed subsequently use these two quantities to update γ_i with different variations; some use the square root of the ratio [85] and others like the IAA algorithm use the measured output power of the MPDR to directly update $\boldsymbol{\gamma}$ skipping the MMSE step in the EM-SBL [86]. An important insight this interpretation provides is that the mentioned algorithms do not include a denoising step. They are effectively agnostic to the prior on $\boldsymbol{x}$ and they only uses MPDR powers to iteratively update $\boldsymbol{\gamma}$. This can explain the better performance EM-SBL has been reported to achieve over other SBL variants.

4.3 Sparsity Promoting Priors and their MMSE Estimates

4.3.1 A Non-Identical Laplace Prior Example

In this section we present a non-identical Laplacian density based algorithm that we call Laplace sparse Bayesian learning (LSBL). We choose this particular prior to demonstrate the capabilities of the new framework. Previous methods could not directly apply a such a prior, and therefore resorted to expressing it as a GSM prior [39]. Other methods like VAMP can only impose an i.i.d. Laplace prior on $\boldsymbol{x}$ but cannot handle the type of prior we use in this section. The prior on $\boldsymbol{x}$ is:

$$p(x_i) = 1/(2\beta_i)\ \exp\big(-|x_i|/\beta_i\big). \tag{4.20}$$

The prior has the unknown hyperparameter vector $\boldsymbol{\beta}$, which needs to be estimated from the data. We note here that this prior is a special case of the Laplace scale mixture prior with a non-informative prior on $\boldsymbol{\beta}$. As mentioned before, for a chosen prior, the MPDR model and model update steps do not change. We only need to change the MMSE step according to the prior, and update any unknown parameters in the prior itself. In the following we detail the equations for the MMSE step and $\boldsymbol{\beta}$ update. The derivation is very similar to the one in [87] with the difference that the prior in [87] is an i.i.d. Laplace prior where all β_is are equal. Therefore we will omit the details of the derivation for space considerations. Given the output of the MPDR (4.11) with output vector $\boldsymbol{r}$ and noise variance $\boldsymbol{\tau}_r$ we carry out the MMSE step as:

$$\hat{x}_i = \frac{1}{2\beta_i\Psi_i}\left[e^{-\alpha_i^-}\gamma_i^- Q\left(\frac{\gamma_i^-}{\sqrt{\tau_{r_i}}}\right) + e^{-\alpha_i^+}\gamma_i^+ Q\left(\frac{-\gamma_i^+}{\sqrt{\tau_{r_i}}}\right)\right] \tag{4.21}$$

$$\begin{aligned}\tau_{x_i} = \frac{1}{2\beta_i\Psi_i}\Bigg[&\big((\gamma_i^+)^2+\tau_{r_i}\big)e^{-\alpha_i^+}\gamma_i^+ Q\left(\frac{-\gamma_i^+}{\sqrt{\tau_{r_i}}}\right)\\ &+\big((\gamma_i^-)^2+\tau_{r_i}\big)e^{-\alpha_i^-}\gamma_i^+ Q\left(\frac{\gamma_i^-}{\sqrt{\tau_{r_i}}}\right) - \frac{2(\tau_{r_i})^2}{\beta_i\sqrt{2\pi\tau_{r_i}}}e^{\frac{-r_i^2}{2\tau_{r_i}}}\Bigg]\end{aligned} \tag{4.22}$$

$$\Psi_i = \frac{1}{2\beta_i}\left[e^{-\alpha_i^-}\,Q\left(\frac{\gamma_i^-}{\sqrt{\tau_{r_i}}}\right) + e^{-\alpha_i^+}\,Q\left(\frac{-\gamma_i^+}{\sqrt{\tau_{r_i}}}\right)\right] \tag{4.23}$$

$$\alpha_i^+ = -\frac{r_i}{\beta_i} - \frac{\tau_{r_i}}{2\beta_i^2},\ \alpha_i^- = \frac{r_i}{\beta_i} - \frac{\tau_{r_i}}{2\beta_i^2} \tag{4.24}$$

$$\gamma_i^+ = r_i + \frac{\tau_{r_i}}{\beta_i},\ \gamma_i^- = r_i - \frac{\tau_{r_i}}{\beta_i}, \tag{4.25}$$

where $Q(.)$ is the standard Q-function, $Q(x) = \frac{1}{2\pi}\int_x^\infty e^{\frac{u^2}{2}}\mathrm{du}$. $\boldsymbol{\beta}$ is updated using the EM algorithm, the update rule is:

$$\beta_i = \frac{1}{\xi_i}\frac{2\tau_{r_i}}{\sqrt{2\pi\tau_{r_i}}}e^{-\frac{r_i^2}{\tau_{r_i}}} + \frac{\xi_i'}{\xi_i} \tag{4.26}$$

$$\xi_i = e^{-\alpha_i^-}\,Q(\frac{\gamma_i^-}{\sqrt{\tau_{r_i}}}) + e^{-\alpha_i^+}\,Q(\frac{-\gamma_i^+}{\sqrt{\tau_{r_i}}}) \tag{4.27}$$

$$\xi_i' = e^{-\alpha_i^+}\gamma_i^+\,Q(\frac{-\gamma_i^+}{\sqrt{\tau_{r_i}}}) - e^{-\alpha_i^-}\gamma_i^-\,Q(\frac{\gamma_i^-}{\sqrt{\tau_{r_i}}}) \tag{4.28}$$

In addition to the GSM and Laplace priors, other sparsity promoting priors can be imposed on $\boldsymbol{x}$. The MMSE estimate for of an AWGN signal $\boldsymbol{x}$ with a sparsity promoting prior $p(\boldsymbol{x})$ was previously found for a number of priors. In the following we will summarize some the priors with their corresponding AWGN MMSE.

4.3.2 Bernoulli-Gaussian Prior

The Bernoulli-Gaussian prior is given by:

$$p(x_i) = (1-\lambda)\delta(x_i) + \lambda\mathcal{N}(x_i;\theta,\phi) \tag{4.29}$$

The MMSE estimate for an AWGN corrupted $\boldsymbol{x}$ with a Bernoulli-Gaussian prior is [88]:

$$\hat{x}_i = \mathbb{E}[x_i|r_i] = \pi_i \mu_i \tag{4.30}$$

$$\tau_{x_i} = var[x_i|r_i] = \pi_i(v_i + |\mu_i^2|) - (\pi_i^2|\mu_i^2|) \tag{4.31}$$

$$\pi_i = \left[1 + \left(\frac{\lambda}{1-\lambda}\frac{\mathcal{N}(r_i;\theta,\phi+\tau_{r_i})}{\mathcal{N}(r_i;0,\tau_{r_i})}\right)^{-1}\right]^{-1} \tag{4.32}$$

$$\mu_i = \frac{r_i/\tau_{r_i} + \theta/\phi}{1/\tau_{r_i} + 1/\phi},\ v_i = \frac{1}{1/\tau_{r_i} + 1/\phi} \tag{4.33}$$

4.3.3 Non-Negative Gaussian Scale Mixture

To enforce a non-negativity constraint on $\boldsymbol{x}$ a rectified GSM prior can be imposed on $\boldsymbol{x}$ as follows [80]:

$$p(x_i) = \mathcal{N}^R(x_i;0,\gamma_i) = 2\mathcal{N}(x_i;0,\gamma_i)u(x_i). \tag{4.34}$$

The MMSE estimate for a non-negative GSM prior is:

$$\hat{x}_i = \mathbb{E}[x_i|r_i] = \eta_i + \sqrt{\nu_i}h(\frac{\eta_i}{\nu_i}),\ \tau_{x_i} = var[x_i|r_i] = \nu_i g(\frac{\eta_i}{\nu_i}) \tag{4.35}$$

$$\eta_i = \frac{r_i\gamma_i}{\tau_{r_i}+\gamma_i},\ \nu_i = \frac{\tau_{r_i}\gamma_i}{\tau_{r_i}+\gamma_i} \tag{4.36}$$

$$h(a) = \frac{\varphi(a)}{\Phi_c(a)},\ \ g(a) = 1 - h(a)\,(h(a)-a) \tag{4.37}$$

where φ refers to the pdf and Φ_c refers to the complimentary cdf of a zero-mean and unit-variance Gaussian distribution.

Similar to the GSM case in EM-SBL, the hyperparameter vector $\boldsymbol{\gamma}$ is unknown and can be

learned from the MMSE estimates using the EM algorithm:

$$\gamma_i = \mathbb{E}[x_i^2] = \hat{x}_i^2 + \tau_{x_i} \tag{4.38}$$

4.3.4 Non-Negative Bernoulli-Gaussian

Another option to impose a non-negativity constraint on $\boldsymbol{x}$ is to impose a non-negative Bernoulli-Gaussian prior on $\boldsymbol{x}$ as follows [79]:

$$p(x_i) = (1-\lambda)\delta(x_i) + \lambda \mathcal{N}_+(x_i, \theta, \phi) \tag{4.39}$$

$$\mathcal{N}_+(x_i, \theta, \phi) = \begin{cases} \dfrac{\mathcal{N}(x_i, \theta, \phi)}{\Phi_c(-\theta/\sqrt{\phi})}, & \text{if } x_i > 0 \\ 0, & \text{if } x_i \leq 0 \end{cases} \tag{4.40}$$

The MMSE estimate is given by:

$$\hat{x}_i = \mathbb{E}[x_i|r_i] = \frac{\lambda}{\xi_i}\pi_i\big(\mu_i + \sqrt{v_i}h(\alpha_i)\big) \tag{4.41}$$

$$\tau_{x_i} = var[x_i|r_i] = \frac{\lambda}{\xi_i}\pi_i\left(v_i g(\alpha_i) + \big(\mu_i + \sqrt{v_i}h(\alpha_i)\big)^2\right) - \hat{x_i}^2 \tag{4.42}$$

$$\xi_i = (1-\lambda)\mathcal{N}(0; r_i, \tau_{r_i}) + \lambda\pi_i \tag{4.43}$$

$$\alpha_i = \frac{-\mu_i}{\sqrt{v_i}},\ \mu_i = \frac{r_i/\tau_{r_i} + \theta/\phi}{1/\tau_{r_i} + 1/\phi} \tag{4.44}$$

$$v_i = \frac{1}{1/\tau_{r_i} + 1/\phi},\ \pi_i = \frac{\mathcal{N}(r_i; \theta, \tau_{r_i} + \phi)\Phi_c(\alpha_i)}{\Phi_c(-\theta/\sqrt{\phi})} \tag{4.45}$$

where φ refers to the pdf and Φ_c refers to the complimentary cdf of a zero-mean and unit-variance Gaussian distribution. $h(.)$ and $g(.)$ are given in (4.37).

4.3.5 A Low Complexity Implementation

Examining the three steps of the proposed algorithm, we can identify that the step with the highest complexity is the MPDR step, since it requires a matrix inversion of order $M \times M$. In this

section we propose using the low complexity generalized approximate message passing algorithm (GAMP), to implement the MPDR step to reduce the complexity of the algorithm. In [66] it was shown that the LMMSE estimator can be efficiently implemented using the Gaussian GAMP (GGAMP) given in Table 4.1. It was also shown that when used as an LMMSE estimator GGAMP had enhanced convergence properties compared to other AMP implementations. These properties included robustness to non i.i.d. Gaussian transformation matrices $\boldsymbol{A}$ when proper damping was used, exact mean upon convergence and it was numerically shown that the approximate variance did not affect the overall performance of the algorithm. Using the linear relationship between the MPDR and the LMMSE [68], the MPDR estimator can be implemented by scaling the GGAMP LMMSE output. The low complexity MPDR algorithm is summarized in Table 4.1.

Table 4.1: GGAMP Low Complexity LMMSE/MPDR

Inputs $\boldsymbol{A}, \boldsymbol{\gamma}, \sigma^2$
Initializations $\boldsymbol{S} \leftarrow \lvert\boldsymbol{A}\rvert^2$ (component wise magnitude squared) $\boldsymbol{z}^0, \hat{\boldsymbol{x}}^1 \leftarrow \boldsymbol{0}$ and $\boldsymbol{\tau}_x^1 > 0$
for $k = 1, 2, ..., K_{\max}$ $\boldsymbol{\tau}_z^k \leftarrow 1/(\sigma^2 + \boldsymbol{S}\boldsymbol{\tau}_x^k)$ $\boldsymbol{z}^k \leftarrow (1-\theta_z)\boldsymbol{z}^{k-1} + \theta_z \frac{\boldsymbol{y} - \boldsymbol{z}^{k-1}\boldsymbol{S}\boldsymbol{\tau}_x^k - \boldsymbol{A}\hat{\boldsymbol{x}}^k}{\boldsymbol{S}\boldsymbol{\tau}_x^k + \sigma^2}$ $\boldsymbol{\tau}_r^k \leftarrow 1/(\boldsymbol{S}^\top \boldsymbol{\tau}_z^k)$ $\boldsymbol{r}^k = (\hat{\boldsymbol{x}}^k + \boldsymbol{\tau}_r^k \boldsymbol{A}^\top \boldsymbol{z}^k)$ $\boldsymbol{\tau}_x^{k+1} \leftarrow \frac{\boldsymbol{\tau}_r^k \boldsymbol{\gamma}^i}{\boldsymbol{\tau}_r^k + \boldsymbol{\gamma}^i}$ $\hat{\boldsymbol{x}}^{k+1} \leftarrow (1-\theta_x)\hat{\boldsymbol{x}}^k + \theta_x \frac{\boldsymbol{r}\boldsymbol{\gamma}^i}{\boldsymbol{\tau}_r^k + \boldsymbol{\gamma}^i}$ end for %end of k loop

In the algorithm, $K_{\max}$ is the maximum number of iterations and θ_z and θ_x are damping factors that are used to guarantee the convergence of the algorithm with non-i.i.d Gaussian $\boldsymbol{A}$, whose values can be determined according to an SVD operation on $\boldsymbol{A}$ [52, 66], or an adaptive procedure of setting the factors can be implemented as in [54]. Instead of scaling the output of the GGAMP algorithm to find the MPDR output, we notice that in the algorithm in Table 4.1, $\boldsymbol{x}$ itself is computed as the LMMSE estimate given $\boldsymbol{r}$. Therefore, upon convergence of the GGAMP, $\boldsymbol{r}$ and

$\boldsymbol{\tau}_r$ can be directly used as the MPDR's output and noise variance. The effect this low complexity approximation might have on the convergence of the algorithm will be studied in Section 4.4, and in the numerical analysis.

With the introduction of this low complexity version of the algorithm, we point out that the VAMP algorithm [32] is another low complexity algorithm that can use general priors for solving the SSR problem. However, while VAMP uses averaged variance vectors, MPDR-SSR uses full vectors which allows the algorithm to address more general priors that are not necessarily identically distributed. Another difference is that the VAMP reduces the complexity using a one time SVD decomposition of $\boldsymbol{A}$, which can be avoided in the case of MPDR-SSR. The SVD computation might be problematic for extra large $\boldsymbol{A}$ matrices, especially that the MPDR-SSR can be fully distributable, where $\boldsymbol{A}$ can be broken down into smaller blocks and does not need to be fully stored when there are memory limitations. Since this chapter focuses on the connection between SBL and its variants to MPDR, extensive comparisons between the proposed algorithm and VAMP are beyond the scope of this chapter and is left for future research.

4.4 Convergence of the Algorithm

In this section we analyze the convergence of the proposed MPDR-SSR algorithm using two main concepts. The first concept is that the MPDR output can be modeled as an AWGN corrupted version of the sparse signal $\boldsymbol{x}$. The second concept, is that proper denoising of the MPDR output reduces the MPDR noise variance and provides a better estimate of $\boldsymbol{x}$. Because the iterative MPDR in our model depends on the power estimates of the elements of $\boldsymbol{x}$, the better estimate of $\boldsymbol{x}$ can be used at the next iteration to improve the MPDR receiver and produce a better AWGN corrupted estimate which can be denoised and so on until convergence. In the following we will refer to previous work that had shown these two concepts, then we will numerically demonstrate their validity.

When the MPDR receiver is applied to an element x_i the interference and noise term v_i from (4.18) is given by:

$$v_i = \sum_{n \neq i} \frac{\boldsymbol{a}_{.i}^{\top} \boldsymbol{a}_{.i} x_i}{\boldsymbol{a}_{.i}^{\top} \boldsymbol{\Sigma}_y^{-1} \boldsymbol{a}_{.i}} + \boldsymbol{w}_i^{\top} \boldsymbol{e}. \tag{4.46}$$

Given the linear relationship between the MPDR and the LMMSE, we can exploit LMMSE results for the statistics of the multi-user interference (MAI) in multi-user communication systems. Intuitively, given that the interference corrupting x_i is a summation of the dot product of columns of $\boldsymbol{A}$ and the independent elements x_j, in the large system limit the central limit theorem will apply, and the interference can be modeled as additive Gaussian noise corrupting the signal. In addition to that the term $\boldsymbol{w}_i^{\top} \boldsymbol{e}$ is a linear transformation of the Gaussian noise, and is also Gaussian. The noise and interference combined are assumed to be Gaussian with zero mean and variance $\boldsymbol{\tau_r}$. The Gaussianity assumption of the interference was shown to be true in the large system limit using different methods, for example [89] used the central limit theorem, while [90, 91] used random matrix theory and finally [92] used statistical physics to show that. In addition to that, the low complexity AMP based MPDR approximation was also shown to produce an AWGN corrupted version of $\boldsymbol{x}$ when the columns of $\boldsymbol{A}$ are i.i.d. Gaussian. In the following we will use QQ-plots to demonstrate the validity of this assumption. We will use column correlated and ill-conditioned $\boldsymbol{A}$ matrices to show that the Gaussianity assumption still holds under non-i.i.d Gaussian $\boldsymbol{A}$, even when using the low complexity versions of the algorithm. Finally, we use multiple priors to show that the result holds beyond the GSM prior assumption. Fig. 4.2 shows QQ-plots for the first, a middle and last iterations of the MPDR output. We can see that empirically the MAI Guassianity assumption is valid throughout the algorithm's iterations. We can also see that the assumption is valid for cases when the entries of $\boldsymbol{A}$ are non i.i.d Gaussian, even for the low complexity MPDR. Finally the assumption is valid when the prior on $\boldsymbol{x}$ is not imposed by a GSM.

For the second part of the convergence study we will examine how the noise variance

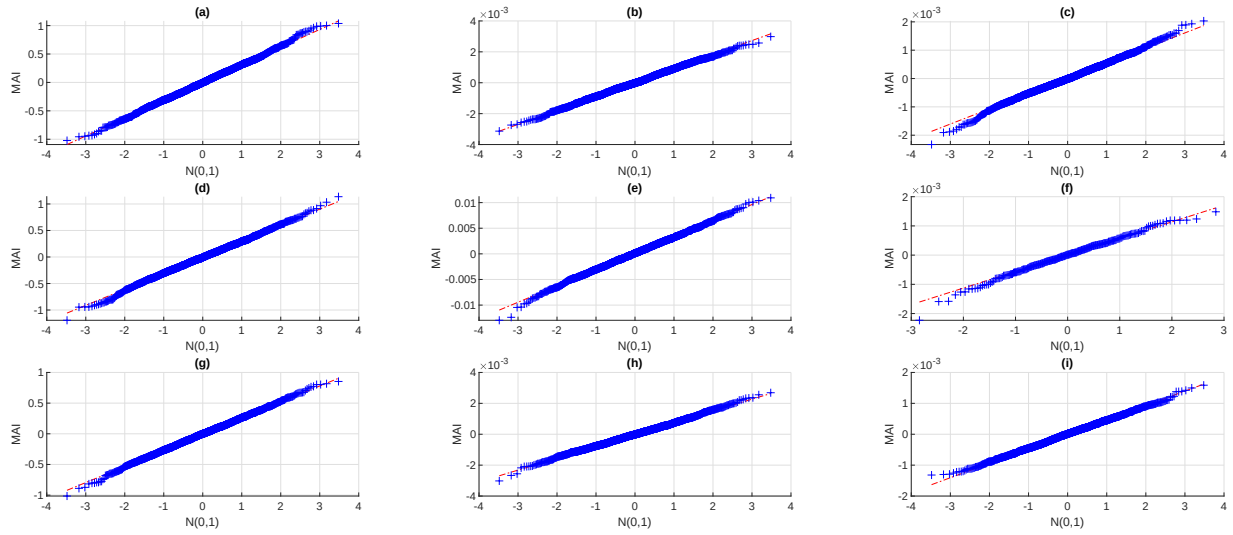


Figure 4.2: QQ plots comparing MAI with a Gaussian distribution. (Left column) First iteration. (Middle column) Middle iteration. (Right column) Last iteration. (Top row) SBL GSM prior with i.i.d Gaussian A and full complexity MPDR. (Middle row) Laplace prior with column correlated A and full complexity MPDR. (Bottom row) Bernoulli-Gaussian prior with ill-conditioned A and low complexity MPDR.

evolves with each iteration. We will show how the MMSE estimate based on the AWGN corrupted $\boldsymbol{x}$ and the prior on $\boldsymbol{x}$ can reduce the noise variance. We note here that the variance estimate $\boldsymbol{\tau_r}$ from (4.15) is not exact, it is an approximation based on assuming the modeled $\boldsymbol{\Sigma}_y$ matches the actual covariance matrix of the measuerment vector, therefore we will also plot the evolution of the actual and estimated variance of the output noise from the MPDR. Based on the results from [82] proper thresholding of a sparse signal corrupted with AWGN can reduce the noise variance. In our case the thresholding is provided by the MMSE estimate, and in the cases of scale mixture priors in addition to the MMSE step we also apply pruning, where the $\boldsymbol{\gamma}$ values less than a predetermined threshold are set to zero to provide an actual thresholding function. We expect the noise variance from the MPDR output to be reduced by the MMSE step, producing a better estimate of $\boldsymbol{x}$. For the experimental study, we introduce the MPDR-BG algorithm that uses the exact Bernoulli-Gaussian prior on $\boldsymbol{x}$, where the MMSE step details for this prior can be found in [88].

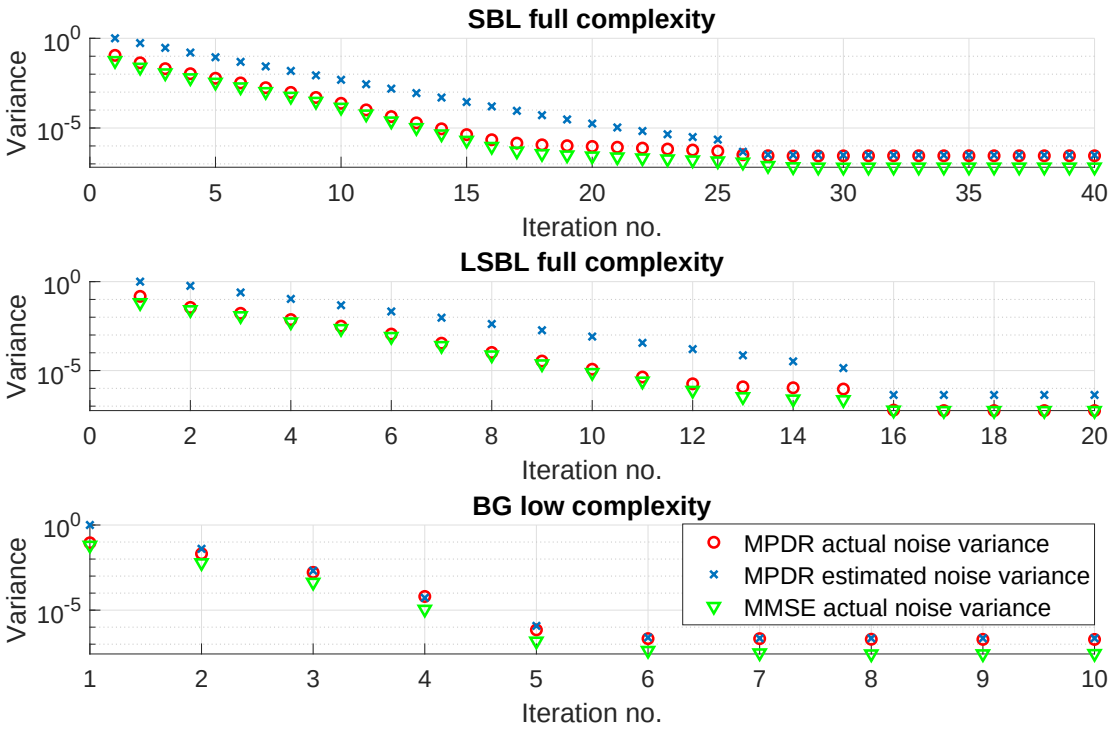


Figure 4.3: Noise variance development with each iteration.(Top) SBL GSM prior with i.i.d Gaussian A and full complexity MPDR. (Middle) Laplace prior with column correlated A and full complexity MPDR. (Bottom) Bernoulli-Gaussian prior with ill-conditioned A and low complexity MPDR.

From Fig. 4.3, it is clear that the MMSE estimate reduces the noise variance compared to the output of the MPDR as expected from the results in [82]. It is also evident that this leads to further reduction in the output noise variance at the next iteration and so on until convergence. We observe that using an exact Bernoulli-Gaussian prior yields the most noise reduction at each iteration and therefore requires the least number of iterations. Moreover, the Laplace based algorithm requires a fewer number of iterations to converge compared to GSM one. We also notice that for the scale mixture algorithms, when we start applying pruning, the difference between the output of the MPDR and the thresholding noise becomes very minimal, and we see significant improvement in noise variance. Finally, we notice that the output variance of the noise is overestimated at first, until the algorithm starts converging and the estimate becomes more accurate.

4.5 Numerical Results

4.5.1 MPDR-SSR Numerical Analysis

In this section we present a numerical study to show the potential benefits of using a non-GSM prior within the MPDR-SSR framework. We will present results for the LSBL algorithm detailed in section 4.3.1. We will also present results for the MPDR-SSR framework with a known Bernoulli-Gaussian prior, assuming that the prior matches the distribution used to generate the signal $\boldsymbol{x}$. Both algorithms will be compared with the EM-SBL algorithm. In addition to the full complexity versions of the algorithms, we will also present results for the low complexity versions of all three algorithms. We will focus on the high SNR case in our numerical analysis, since the estimation of the noise variance can affect the algorithms' performance. It is left for future research to exploit the new understanding of the SBL and corresponding algorithms to find an optimal estimate of the noise variance.

For the first experiment we set $N = 1000$, $M = 500$ and $SNR = 60$dB. Elements of $\boldsymbol{x}$ are independently drawn from a Bernoulli-Gaussian distribution with zero mean and unit variance, where the sparsity of $\boldsymbol{x}$ spans a range of K/N from 0.05 to 0.15. The performance metric we use is the normalized means square error given by, NMSE $\triangleq \|\hat{\boldsymbol{x}} - \boldsymbol{x}\|^2 / \|\boldsymbol{x}\|^2$.

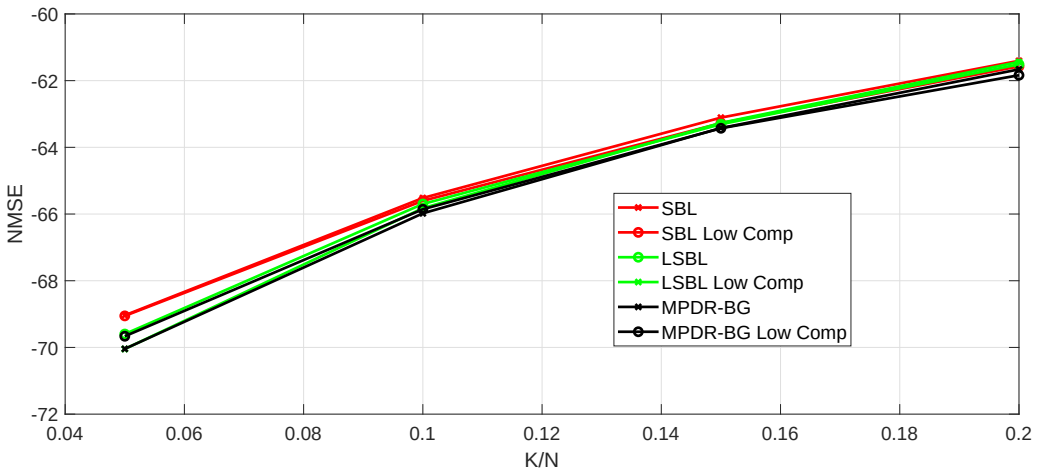


Figure 4.4: Performance of SSR Algorithms with BG Elements

From Fig. 4.4 we can see that Bernoulli-Gaussian algorithms provide a slight performance improvement compared to the original SBL algorithm. In addition to that, the LSBL provides some improvement over SBL as well. However, in general all three algorithms perform within the same range. We also can conclude that the low complexity implementation of the algorithms does not cause significant performance degradation compared to the full complexity algorithms.

To highlight some of the benefits of the other algorithms compared to the EM-SBL, we will examine the runtimes of the algorithms. We run a comparison between runtimes of the low complexity EM-SBL, LSBL and MPDR-BG algorithms. We exclude the full complexity algorithms because when complexity is of interest, using the high complexity algorithms is impractical given that the low complexity ones provide similar performance with significantly lower complexity as shown in [66, 67]. We run the experiments using $N = 10000$, $M = 2500$ and $SNR = 60$dB. We compute the runtimes for a range of K/N from 0.025 to 0.125. We plot the runtimes in seconds of the three algorithms in Fig. 4.5.

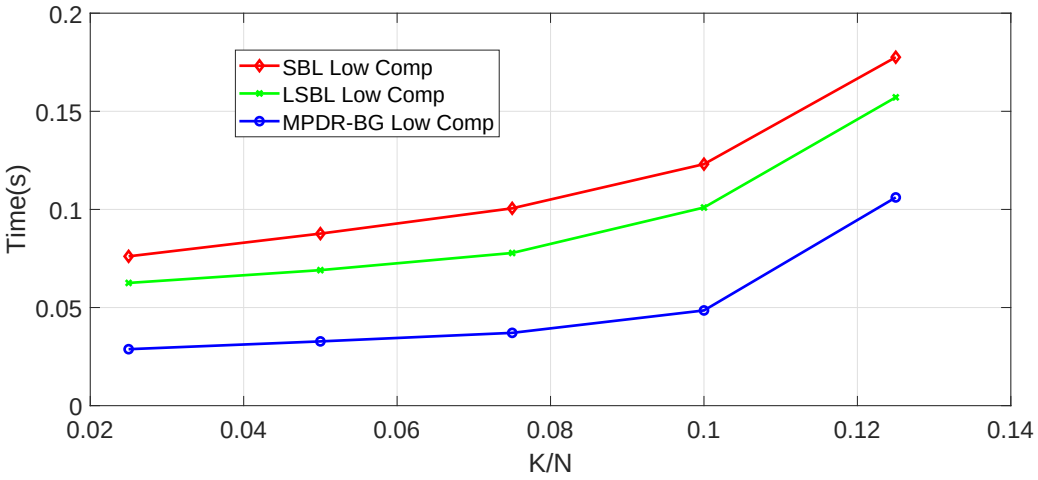


Figure 4.5: Runtime of Fast SSR Algorithms

From section 4.4, we saw how the LSBL requires fewer iterations to converge compared to the EM-SBL, and the Bernouli-Gaussian algorithms required fewer iterations than both algorithms. It is clear from Fig. 4.5 that this translates into faster runtimes for the LSBL compared to EM-SBL. This result reinforces the conclusions from [38] since the LSBL prior is equivalent to a reweighed

ℓ_1 and the EM-SBL is equivalent to a reweighed ℓ_2 algorithm. The performance and runtimes results suggest that the LSBL can provide a faster algorithm compared to the EM-SBL without any performance degradation. They also indicate that when the prior on $\boldsymbol{x}$ is known, the MPDR-BG algorithm can provide better performance and faster runtimes than the other two algorithms.

To highlight the benefits of the knowledge of the prior on $\boldsymbol{x}$, we repeat the performance experiment with samples of $\boldsymbol{x}$ generated from a Bernoulli-Gaussian with a non-zero mean. We use a mean value $\theta = 5$ and repeat the experiment from Fig 4.4 with $N = 1000$, $M = 500$ and $SNR = 60$dB.

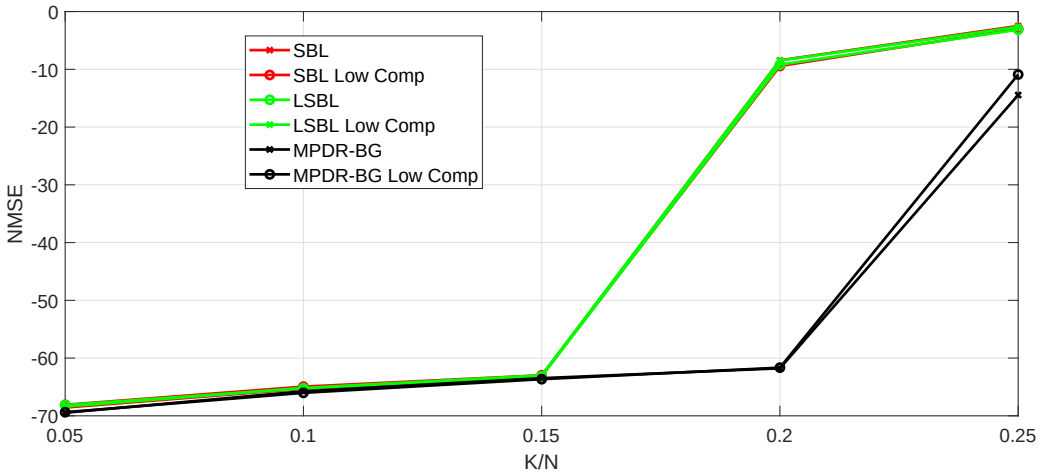


Figure 4.6: Performance of SSR Algorithms with BG Non-Zero Mean Elements

Fig. 4.6 shows how the MPDR-BG algorithm can perform better than the other two algorithms when the non-zero elements of $\boldsymbol{x}$ are non-zero mean. Compared to the other two algorithms, the MPDR-BG algorithm was able to handle lower sparsity levels before it failed. This serves as an example of how our proposed framework provides more flexibility in choosing the prior which can translate into better performance when more information about $\boldsymbol{x}$ is available.

4.6 Conclusion

We presented an MPDR based interpretation of the SBL algorithm. A more intuitive understanding of the algorithm can be achieved based on this interpretation, which can provide important insight into the algorithm. We showed how this insight can be of significant value by proposing modifications to the original SBL in the light of this new understanding. The modifications enabled us to lower the complexity of the algorithm, while enabling incorporation of a wider range of sparsity promoting priors, resulting in the MPDR-SSR framework. Improvements to the SBL algorithm and its variants based on the new insights are not limited to the examples we have presented in this work. Future research can include finding better noise estimates to improve the denoising step, or using denoisers that are specifically designed for a certain application like imaging for example.

4.7 Acknowledgment

This chapter, in full, is a reprint of material published in the article Maher Al-Shoukairi, and Bhaskar D. Rao, "An MPDR Perspective and Extension of Sparse Bayesian Learning", under review at IEEE Transactions on Signal Processing. I was the primary author and B. D. Rao supervised the research.

Chapter 5

An Array Processing Perspective of Sparse Bayesian Learning Variants

5.1 Introduction

The fast-SBL algorithm was proposed in [71]. The proposed algorithm descends on the SBL's cost function sequentially using a greedy approach which adds basis to the model one at a time to avoid inverting a large matrix. Similar to the original SBL the fast-SBL algorithm is limited in the choice of prior to the GSM class and cannot directly incorporate additional information about $\boldsymbol{x}$. In this chapter, we show how the array processing interpretation of the EM-SBL algorithm can be extended to interpret the fast-SBL algorithm. Moreover, we demonstrate how this novel interpretation allows us to incorporate more general priors into the fast-SBL algorithm, enhancing its performance in certain cases.

We also demonstrate how the how the array processing interpretation extends to the MMV SBL variant that is TMSBL. Based on this interpretation we propose a novel AMP based low complexity MMV algorithm that reduces the complexity of the TMSBL algorithm. The proposed algorithm outperforms other AMP algorithms such as AMP-MMV algorithm from [65] and the

GGAMP-TSBL from [66]. The proposed algorithm also offer complexity advantages over the AMP algorithms when the number of measurements is large.

5.1.1 Chapter's Organization

In Section 5.2 we provide an MVDR interpretation of the sequential fast-SBL algorithm, and we propose an algorithm that extends it beyond GSM priors. In Section 5.3 we apply the MPDR framework to the MMV problem, where we re-derive the TMSBL algorithm and propose a low complexity version of it. In section 5.4 we present numerical results to show the benefits of the algorithms that we proposed based on the array processing interpretation of the original algorithms.

5.2 An Array Processing Perspective of the Sequential Fast-SBL Algorithm

The sequential fast-SBL algorithm [71] was proposed to address the high complexity of the SBL algorithm. The algorithm is a greedy version of the SBL and is based on updating a single hyperparameter γ_i at each iteration to avoid solving a system of N linear equations. In the following we summarize the algorithm from [71] and then we introduce an MVDR based explanation of the algorithm. Similar to previous sections, the new interpretation provides important insight into the algorithm's iterations. As an example, we will show how this insight allows us to extend the algorithm to incorporate additional information about the sparse vector $\boldsymbol{x}$ when available. We demonstrate how incorporating this information can improve the algorithm's performance.

5.2.1 Fast SBL Algorithm Summary

The original SBL algorithm suggests that when estimating the full vector $\boldsymbol{\gamma}$ any γ_i that becomes close to zero can be pruned out of the system. The column of $\boldsymbol{A}$ denoted by the basis vector $\boldsymbol{a}_{.i}$ corresponding to a pruned γ_i will be pruned out, lowering the complexity of each iteration. When the algorithm is close to convergence the problem size becomes closer to the number of non-zero elements in $\boldsymbol{x}$ which is typically much smaller than the length of $\boldsymbol{x}$. The sequential fast-SBL algorithm in [71] exploits the sparsity of the vector $\boldsymbol{x}$ and the corresponding hyperparameter vector $\boldsymbol{\gamma}$. The algorithm starts with an empty set assuming all γ_is are equal to zero, and then sequentially adds, deletes or updates the values of γ_is and the corresponding columns of $\boldsymbol{A}$, leading to a more computationally efficient algorithm. We will summarize the main idea behind the algorithm, while the detailed derivation can be found in [71]. In the fast-SBL algorithm [71], sequential updates of γ_is are achieved by decomposing the covariance matrix Σ_y in the cost function of the SBL algorithm (4.3):

$$\boldsymbol{\Sigma}_y = \sigma^2 \boldsymbol{I} + \sum_{j \neq i} \gamma_j \boldsymbol{a}_{.j} \boldsymbol{a}_{.j}^\top + \gamma_i a_i a_i^\top = \boldsymbol{\Sigma}_{I_i} + \gamma_i a_i a_i^\top, \tag{5.1}$$

where Σ_{I_i} is the measurement covariance matrix with the contribution of γ_i removed. We refer to Σ_{I_i} as the covariance matrix of the interference from all the other x_js when the element of interest to be recovered is x_i. This notation is consistent with the array processing perspective of the problem at hand. Based on this decomposition the SBL cost function (4.3) can be decomposed into a term which is independent of γ_i and a term dependent on a single γ_i, and the cost function can be minimized with respect to a single element at a time. In [71] it was shown that the unique minimum of $\chi(\boldsymbol{\gamma})$ with respect to γ_i is:

$$\gamma_i = \begin{cases} \dfrac{q_i^2 - s_i}{s_i^2}, & \text{if } q_i^2 > s_i \\ 0, & \text{if } q_i^2 \leq s_i \end{cases} \tag{5.2}$$

$$q_i = \boldsymbol{a}_{.i}^{\top} \boldsymbol{\Sigma}_{I_i}^{-1} \boldsymbol{y}, \;\; s_i = \boldsymbol{a}_{.i}^{\top} \boldsymbol{\Sigma}_{I_i}^{-1} \boldsymbol{a}_{.i}. \tag{5.3}$$

Based on (5.2), the fast-SBL algorithm starts by assuming all γ_i values are zeros, and the set of corresponding effective columns or basis chosen of the matrix $\boldsymbol{A}$ is empty at this stage. For a single chosen γ_i the values q_i and s_i are computed and a decision on whether to include it and its corresponding basis, or exclude it from the model representation is made based on (5.2). The schedule for choosing which γ_i to update next can be random, but in general the algorithm chooses the next γ_i that results in the maximum reduction of the SBL's cost function. $\hat{\boldsymbol{x}}$ and Σ_x can be updated based on the updated γ_i and corresponding $\boldsymbol{a}_{.i}$ using (4.7) with a modified $\boldsymbol{A}$ matrix that includes active columns only.

5.2.2 An MVDR Interpretation of Fast SBL

The algorithm can be interpreted as a greedy way of constructing an MVDR beamformer. We start with rewriting the fast-SBL's $\boldsymbol{\gamma}$ update rules to make them more inline with the MVDR receiver. Using s_i and q_i definitions, (5.2) can be rewritten as:

$$\gamma_i = \begin{cases} (\frac{\boldsymbol{a}_{.i}^{\top} \boldsymbol{\Sigma}_{I_i}^{-1} \boldsymbol{y}}{\boldsymbol{a}_{.i}^{\top} \boldsymbol{\Sigma}_{I_i}^{-1} \boldsymbol{a}_{.i}})^2 - \frac{1}{\boldsymbol{a}_{.i}^{\top} \boldsymbol{\Sigma}_{I_i}^{-1} \boldsymbol{a}_{.i}}, & \text{if } (\frac{\boldsymbol{a}_{.i}^{\top} \boldsymbol{\Sigma}_{I_i}^{-1} \boldsymbol{y}}{\boldsymbol{a}_{.i}^{\top} \boldsymbol{\Sigma}_{I_i}^{-1} \boldsymbol{a}_{.i}})^2 > \frac{1}{\boldsymbol{a}_{.i}^{\top} \boldsymbol{\Sigma}_{I_i}^{-1} \boldsymbol{a}_{.i}} \\ 0, & \text{if } (\frac{\boldsymbol{a}_{.i}^{\top} \boldsymbol{\Sigma}_{I_i}^{-1} \boldsymbol{y}}{\boldsymbol{a}_{.i}^{\top} \boldsymbol{\Sigma}_{I_i}^{-1} \boldsymbol{a}_{.i}})^2 \leq \frac{1}{\boldsymbol{a}_{.i}^{\top} \boldsymbol{\Sigma}_{I_i}^{-1} \boldsymbol{a}_{.i}} \end{cases} \tag{5.4}$$

Similar to the previous section, the algorithm starts by modeling the measurement covariance matrix by $\boldsymbol{\Sigma}_y = (\sigma^2 \boldsymbol{I} + \boldsymbol{A}^{\top} \boldsymbol{\Gamma} \boldsymbol{A})$, where $\boldsymbol{\Gamma} = diag(\boldsymbol{\gamma})$. In this case the algorithm starts with an all zero vector $\boldsymbol{\gamma}$, where single γ_is are added to the model at each iteration, and later on they can updated or deleted. After initializing the algorithm with the first γ_i value, and based on the covariance matrix model $\boldsymbol{\Sigma}_y$, the algorithm seeks detecting non-zero elements of $\boldsymbol{x}$ one at a time. To do so the algorithm builds an MVDR receiver for x_i, where unlike the MPDR receiver, the

MVDR receiver uses the noise and interference covariance matrix $\mathbf{\Sigma}_{I_i}$ from (5.1), the receiver is:

$$\hat{\boldsymbol{w}}_i = \frac{\boldsymbol{a}_{.i}^\top \mathbf{\Sigma}_{I_i}^{-1} \boldsymbol{y}}{\boldsymbol{a}_{.i}^\top \mathbf{\Sigma}_{I_i}^{-1} \boldsymbol{a}_{.i}}, \;\; \hat{\boldsymbol{w}}_i^\top \boldsymbol{y} = x_i + v_i, \tag{5.5}$$

where v_i is refered to by MAI because it represents the interference from other elements of $\boldsymbol{x}$ corrupting the element x_i at the output of the MVDR. Since this is a linear receiver, MAI results referred to in previous sections hold here as well. Therefore the output of the receiver is an AWGN corrupted version of x_i and v_i is modeled as a zero mean Gaussian with its variance given by $\frac{1}{\boldsymbol{a}_{.i}^\top \mathbf{\Sigma}_{I_i}^{-1} \boldsymbol{a}_{.i}}$ [68]. Our interpretation of the fast SBL algorithm is based on two values of interest associated with the given MVDR. The first value is the total measured output power of the receiver given by $(\frac{\boldsymbol{a}_{.i}^\top \mathbf{\Sigma}_{I_i}^{-1} \boldsymbol{y}}{\boldsymbol{a}_{.i}^\top \mathbf{\Sigma}_{I_i}^{-1} \boldsymbol{a}_{.i}})^2$ which represents the total output power from the MVDR BF, combining the powers of the desired x_i and the MAI power from all other elements of $\boldsymbol{x}$. The second value is the expected MAI power given by the model which is the variance of the output AWGN noise given by $\frac{1}{\boldsymbol{a}_{.i}^\top \mathbf{\Sigma}_{I_i}^{-1} \boldsymbol{a}_{.i}}$. We note here that the accuracy of the noise power estimate depends on how good $\mathbf{\Sigma}_y$ approximates the actual measurement covariance matrix, and therefore as more γ_is are estimated properly this MAI power estimate will become more accurate. Based on these definitions, the algorithm proceeds by comparing the total measured output power of the MVDR BF to the expected output noise power at a chosen x_i. If the total output power of the MVDR exceeds the expected noise power, the algorithm attributes the difference to a non-zero x_i at that location, and sets the power of that source γ_i to that difference. If γ_i was already previously added to the model, it's value is updated as the difference as well. If the output power is equal to or less than the noise power, the algorithm attributes the output power to noise only and assumes the absence of a non-zero element at that location, therefore γ_i is removed from the model. Because the estimate of the covariance matrices $\mathbf{\Sigma}_{I_i}$s are improving as we improve the estimates of γ_is, the noise variance estimates are also improving with each iteration. Therefore the algorithm needs to revisit its decisions on γ_is as we go forward. Similar to the convergence analysis in previous

sections, each iteration improves the estimate of $\boldsymbol{\gamma}$ which improves the estimate of the output noise, which in turn leads to further improvement in the estimate of $\boldsymbol{\gamma}$, and so on until convergence. Based on the γ_is included in the model, the mean $\hat{\boldsymbol{x}}$ and covariance $\boldsymbol{\Sigma}_x$ of $\boldsymbol{x}$ from (4.7) can be computed using fewer number of columns of $\boldsymbol{A}$ hence reducing the complexity.

5.2.3 Exploiting Additional Information with Fast SBL

This new perspective of the fast SBL algorithm provides opportunities to improve on the algorithm. As an example, we extend the algorithm to incorporate additional information about $\boldsymbol{x}$ by imposing priors that represent this information. To demonstrate the performance gains from such extension, we will impose some new priors on $\boldsymbol{x}$ and propose modifications to the fast-SBL algorithm to incorporate these priors.

To include the prior in the process of improving the MVDR, we propose a modified algorithm. The proposed algorithm keeps the steps of adding and deleting γ_is from the model unchanged. However, the values of $\hat{\boldsymbol{x}}$ and $\boldsymbol{\tau}_{\boldsymbol{x}}$ are computed using the two steps used in the previous section, where the MVDR receiver is first applied to x_i and then an MMSE estimate is computed based on the output of the MVDR and the chosen prior on x_i. In addition to that, whenever we need to re-estimate a value of γ_i the estimate is computed using the power estimate of x_i which is $\mathbb{E}[x_i^2] = \hat{x}_i^2 + \tau_{x_i}$. If the prior on $\boldsymbol{x}$ has unknown parameters, they are learned using the EM algorithm based on the approximate posterior $p(\hat{\boldsymbol{x}}|\boldsymbol{r})$ produced by the MMSE output. The algorithm's details are summarized in Table 5.1.

Previous work from [34] imposed a Laplacian prior on $\boldsymbol{x}$ by representing it using a GSM and a density on $\boldsymbol{\gamma}$. However, non GSM priors could not be previously incorporated into the fast-SBL algorithm. In the numerical analysis section we will compare the algorithm from [34] to some more flexible priors that can be handled by the proposed algorithm, and we will demonstrate how in some cases imposing non-GSM priors can enhance the performance of the algorithm. We will present results for a number of algorithms, the first set addresses the problem in (4.1), where

Table 5.1: Fast MVDR-SSR algorithm

Input: $\boldsymbol{A}, \boldsymbol{y}, \sigma^2$
Set all γ_is to 0 Initialize a single γ_i: $\gamma_i = \frac{\|\boldsymbol{a}_{.i}^\top \boldsymbol{y}\|^2 / \|\boldsymbol{a}_{.i}\|^2 - \sigma^2}{\|\boldsymbol{a}_{.i}\|^2}$

While convergence is not achieved
Select a candidate element γ_i and the corresponding basis $\boldsymbol{a}_{.i}$
update s_i and q_i using (5.3)
if $q_i^2 - s_i > 0$ AND $\gamma_i = 0$ then
add $\boldsymbol{a}_{.i}$ to the model and set $\gamma_i = q_i^2 - s_i$
compute $r_i = q_i \boldsymbol{y}/s_i, \tau_{r_i} = 1/s_i$
$\hat{x}_i = \mathbb{E}[x_i \vert r_i], \tau_{x_i} = var[x_i \vert r_i]$
else if $q_i^2 - s_i > 0$ AND $\gamma_i > 0$ then
compute $r_i = q_i \boldsymbol{y}/s_i, \tau_{r_i} = 1/s_i$
$\hat{x}_i = \mathbb{E}[x_i \vert r_i], \tau_{x_i} = var[x_i \vert r_i]$
re-estimate γ_i, $\gamma_i = \hat{x}_i^2 + \tau_{x_i}$
else if $q_i^2 - s_i < 0$ AND $\gamma_i = 0$ then
prune $\boldsymbol{a}_{.i}$ from the model and set $\gamma_i = 0$
end if
end while

we will impose an i.i.d. Laplace prior, a non-identical Laplace and a Bernoulli-Gaussian prior on $\boldsymbol{x}$, resulting in the algorithms, Fast-MVDR-$\ell 1$, Fast-MVDR-Laplace and Fast-MVDR-BG. The second set of algorithms address the case when a non-negative constraint is imposed on $\boldsymbol{x}$, therefore we impose a non-negative GSM prior and a known non-negative Bernoulli-Gaussian prior on $\boldsymbol{x}$ resulting in the algorithms Fast-NN-MVDR-SBL and Fast-NN-MVDR-BG where the MMSE step details can be found in [79, 80].

5.3 MMV SBL Using MPDR

In this section, we demonstrate the strength of the proposed MPDR framework by applying it to the Multiple Measurement Vector (row sparse signal vectors using a sequence of measurements that are acquired using the same sensing matrix $\boldsymbol{A}$). The row sparse signal vectors of interest share a common sparsity profile, and it is common for the amplitudes of non-zero

elements of the signals to be correlated. We refer to this correlation by temporal correlation, assuming the measurements are acquired at different points in time. It was shown in [57–60] that having multiple vectors can improve the recovery performance compared to the single measurement case. Moreover it was found that not accounting for the temporal correlation can result in serious degradation in the algorithm's performance. A number of Bayesian algorithms were proposed to address the MMV problem. Among the most successful is a Bayesian algorithm known as the TMSBL algorithm [61], which models the sparsity and correlation of the signals using a modified Gaussian scale mixture prior. Other Bayesian algorithms addressed the high complexity issues associated with the TMSBL algorithm by using an AMP based model with an AR(1) process to model the temporal correlation [65, 66]. Although AMP algorithms did offer significant complexity improvements, TMSBL still achieved superior successful recovery rates over them. In the following we summarize the TMSBL model and algorithm, we will then demonstrate how the MPDR SSR framework can be applied to the MMV problem. The strength of the approach lies in the fact that the MPDR decouples the measurements in space, allowing us to deal with the temporal correlation one row at a time simplifying the problem significantly. This approach enables the potential to deal with temporal correlation models that would be too complex to address without decoupling the measurements. As an example, we will apply the MPDR framework to the TMSBL model and show how the resulting algorithm is equivalent to the TMSBL. Based on this new MPDR derivation of the TMSBL, we will propose a low complexity algorithm based on the AMP MPDR receiver. In the numerical analysis section we will show how the proposed algorithm outperforms previous AMP based MMV algorithms, and how it has an advantage over them in complexity as the number of measurement grows.

5.3.1 TMSBL Algorithm

The MMV model can be stated as:

$$\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{X} + \boldsymbol{E}, \tag{5.6}$$

where $\boldsymbol{Y} = [\boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)} ..., \boldsymbol{y}^{(T)}]$ with $\boldsymbol{y}^{(t)} \in \mathbb{R}^M$. The objective is to recover $\boldsymbol{X} = [\boldsymbol{x}^{(t)}, \boldsymbol{x}^{(2)} ..., \boldsymbol{x}^{(T)}]$ with $\boldsymbol{x}^{(t)} \in \mathbb{R}^N$, where the index (t) indicates the measurement number, with T total measurements. $\boldsymbol{A} \in \mathbb{R}^{M \times N}$ is known, and $\boldsymbol{E} = [\boldsymbol{e}^{(1)}, \boldsymbol{e}^{(2)} ..., \boldsymbol{e}^{(T)}]$ is matrix with columns of i.i.d. noise vectors modeled as $\boldsymbol{e}^{(t)} \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I})$.

As mentioned before, TMSBL assumes a common sparsity profile. Furthermore, TMSBL assumes the sources and therefore the rows of $\boldsymbol{X}$ are independent. The common sparsity profile and the temporal correlation are modeled using the following Gaussian prior on each row of $\boldsymbol{X}$:

$$\boldsymbol{x}_{i.} = \mathcal{N}(0, \boldsymbol{B}\gamma_i), \tag{5.7}$$

where $\boldsymbol{x}_{i.}$ is the row i of the matrix $\boldsymbol{X}$, γ_i controls the row sparsity and $\boldsymbol{B} \in \mathbb{R}^{T \times T}$ models the temporal correlation within $\boldsymbol{x}_{i.}$. $\boldsymbol{B}$ is estimated by the algorithm, and to avoid over-fitting $\boldsymbol{B}$ is assumed to be the same across all rows.

Based on this prior and the system model (5.6), TMSBL attempts to use the EM algorithm to learn the hyperparamer vector $\boldsymbol{\gamma}$ and $\boldsymbol{B}$ from the data. TMSBL steps are derived in [61] by reducing the original problem's size and applying some approximations. We summarize the approximate E and M steps of the algorithm, while the The details of the derivation can be found

in [61].

$$\textbf{TMSBL E-step}: \boldsymbol{\Sigma}_x = (\boldsymbol{\Gamma}^{-1} + \sigma^{-2}\boldsymbol{A}^\top \boldsymbol{A})^{-1} \tag{5.8}$$

$$\hat{X} = \boldsymbol{\Gamma}\boldsymbol{A}^\top (\sigma^2 \boldsymbol{I} + \boldsymbol{A}\boldsymbol{\Gamma}\boldsymbol{A}^\top)^{-1}\boldsymbol{Y} \tag{5.9}$$

$$\textbf{TMSBL M-step}: \gamma_i = \frac{1}{T}\boldsymbol{x}_{i.}\boldsymbol{B}^{-1}\boldsymbol{x}_{i.}^\top + (\boldsymbol{\Sigma}_x)_{ii} \tag{5.10}$$

$$\tilde{\boldsymbol{B}} = \sum_{i=1}^{N} \frac{\boldsymbol{x}_{i.}^\top \boldsymbol{x}_{i.}}{\gamma_i}, \quad \boldsymbol{B} = \tilde{\boldsymbol{B}}/\|\boldsymbol{B}\|_{\mathcal{F}} \tag{5.11}$$

5.3.2 Iterative MPDR Applied to MMV Problems

The same three step technique proposed in section 4.2.4 is applied to the MMV problem: applying an MPDR receiver based on the current estimate of $\boldsymbol{X}$ to the measurement matrix $\boldsymbol{Y}$, an MMSE step based on the assumed prior on $\boldsymbol{X}$ to denoise the output of the MPDR and finally an update to the powers in the MPDR model based on the new estimate of $\boldsymbol{X}$.

MPDR Receiver Step: As before, the ith source $\boldsymbol{x}_{i.}$ is isolated using the MPDR BF:

$$\boldsymbol{w}_i = \frac{\boldsymbol{\Sigma}_y^{-1}\boldsymbol{a}_{.i}}{\boldsymbol{a}_{.i}^\top \boldsymbol{\Sigma}_y^{-1}\boldsymbol{a}_{.i}}, \;\; \boldsymbol{\Sigma}_y = (\boldsymbol{A}\boldsymbol{\Gamma}\boldsymbol{A}^\top + \sigma^2\boldsymbol{I}), \;\; \boldsymbol{\Gamma} = \text{Diag}(\boldsymbol{\gamma}) \tag{5.12}$$

This MPDR step is general, and it only uses the common sparsity profile assumption of $\boldsymbol{X}$. Therefore each row of $\boldsymbol{X}$ shares the power estimate parameter in the model which is represented by γ_i. Another insight, the framework provides into TMSBL (Equation 5.9) is that for the method to be successful, i.e. mitigate interference for each measurement, the power levels have to consistent in time. This means a B matrix with ones or near ones along the diagonal. When the MPDR receiver is applied to the measurement matrix, each row is decoupled into the original signal corrupted by noise:

$$\boldsymbol{r}_{i.} = \boldsymbol{w}_i^\top \boldsymbol{Y} = \boldsymbol{x}_{i.} + \boldsymbol{v}_{i.}, \tag{5.13}$$

where $\boldsymbol{v}_{i.}$ represents the interference plus noise, and we will refer to it by the output noise. Based on the results of linear receivers presented in section 4.4, the distribution of each element of the output noise v_{it} is Gaussian, and therefore each row of the noise matrix $\boldsymbol{v}_{i.}$ is jointly Gaussian.

Denoising (MMSE) Step: Decoupling reduces the complexity of the problem significantly, because the decoupled rows need to be only jointly denoised in time. This is as opposed to jointly processing the full matrix $\boldsymbol{X}$ when the MPDR receiver is not applied first. In the following, we will use the TMSBL prior on $\boldsymbol{x}_{i.}$ as an example on how to proceed with denoising based on an MMSE estimate of $\boldsymbol{x}_{i.}$. Other techniques can be applied for the denoising step, but we leave that for future work.

Given that each row of the noise matrix is jointly Gaussian, to carry out the MMSE step we will need to find noise covariance matrix of noise $\boldsymbol{v}_{i.}$. We start by finding the covariance matrix of the output of the MPDR receiver $\boldsymbol{r}_{i.}$ denoted by $\boldsymbol{\Sigma}_{r_i}$, to simplify the analysis we will consider one element of the cavariance matrix at a time. We use the subscripts a and b to denote measurement indexes:

$$
\begin{aligned}
(\boldsymbol{\Sigma}_{r_i})_{aa} &= \sum_{l=1}^{N} \boldsymbol{w}_i^\top (\boldsymbol{a}_{.l} B_{aa} \gamma_l \boldsymbol{a}_{.l}^\top + \sigma^2) \boldsymbol{w}_i \\
&= \boldsymbol{w}_i^\top (B_{aa} \boldsymbol{A} \boldsymbol{\Gamma} \boldsymbol{A}^\top + \sigma^2 \boldsymbol{I}) \boldsymbol{w}_i \approx \boldsymbol{w}_i^\top B_{aa} (\boldsymbol{A} \boldsymbol{\gamma} \boldsymbol{A}^\top + \sigma^2 \boldsymbol{I}) \boldsymbol{w}_i \qquad (5.14) \\
(\boldsymbol{\Sigma}_{r_i})_{ab} &= \sum_{l=1}^{N} \boldsymbol{w}_i^\top (\boldsymbol{a}_{.l} B_{ab} \gamma_l \boldsymbol{a}_{.l}^\top) \boldsymbol{w}_i \\
&= \boldsymbol{w}_i^\top (B_{ab} \boldsymbol{A} \boldsymbol{\Gamma} \boldsymbol{A}^\top) \boldsymbol{w}_i \approx \boldsymbol{w}_i^\top B_{ab} (\boldsymbol{A} \boldsymbol{\Gamma} \boldsymbol{A}^\top + \sigma^2 \boldsymbol{I}) \boldsymbol{w}_i \qquad (5.15)
\end{aligned}
$$

We argue that the approximations in (5.14) and (5.15) are valid not only at high SNR, but at lower SNR values as well. The iterative MPDR insight allows us to see that at initial iterations the term $\boldsymbol{w}_i^\top \boldsymbol{A} \boldsymbol{\Gamma} \boldsymbol{A}^\top \boldsymbol{w}_i$ will be more dominant than the noise term. This is due to the fact that the MPDR receiver is not providing sufficient interference cancellation when $\boldsymbol{\gamma}$ is not accurate. While the algorithm has not converged to the correct $\boldsymbol{\gamma}$ values, the first term will continue to be dominant.

However, if the algorithm does converge to the correct $\boldsymbol{\gamma}$ it means that the algorithm has converged to the correct solution and the approximation is no longer relevant. This explanation highlights why the MPDR interpretation is valuable in understanding TMSBL's iterations, where a similar approximation was made in [61], but it was only able to point out its validity at high SNR without providing further insight into why it still works for lower SNR values.

$$\boldsymbol{\Sigma}_{r_i} = \boldsymbol{B}\frac{1}{\boldsymbol{a}_{.i}^{\top}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{a}_{.i}} \tag{5.16}$$

$$\boldsymbol{\Sigma}_{v_i} = \boldsymbol{B}\frac{1}{\boldsymbol{a}_{.i}^{\top}\boldsymbol{\Sigma}_y^{-1}\boldsymbol{a}_{.i}} - \boldsymbol{B}\gamma_i = \boldsymbol{B}\tau_{r_i}. \tag{5.17}$$

Based on the noise covariance matrix and the prior on $\boldsymbol{X}$ the MMSE step can be carried out as follows:

$$\hat{\boldsymbol{x}}_{i.} = \boldsymbol{B}\gamma_i(\boldsymbol{B}\gamma_i + \boldsymbol{B}\tau_{r_i})^{-1}\boldsymbol{r}_{i.} = \gamma_i(\gamma_i + \tau_{r_i})^{-1}\boldsymbol{r}_{i.} \tag{5.18}$$

$$\boldsymbol{\Sigma}_{x_{i.}} = \boldsymbol{B}\gamma_i - \boldsymbol{B}\gamma_i(\boldsymbol{B}\gamma_i + \boldsymbol{B}\tau_{r_i})^{-1}\boldsymbol{B}\gamma_i = \boldsymbol{B}\tau_{x_i} \tag{5.19}$$

MPDR Model Update To update the MPDR receiver with the proper values $\boldsymbol{\gamma}$, we will use the fact that in the model $\mathbb{E}[\boldsymbol{x}_{i.}\boldsymbol{x}_{i.}^{\top}] = \boldsymbol{B}\gamma_i$. Given the posterior mean (5.18) and covariance (5.19) of $\boldsymbol{x}_{i.}$ we have:

$$\mathbb{E}[\boldsymbol{x}_{i.}\boldsymbol{x}_{i.}^{\top}] = \boldsymbol{B}\gamma_i = \boldsymbol{\Sigma}_{x_{i.}} + \hat{\boldsymbol{x}}_{.i}\hat{\boldsymbol{x}}_{.i}^{\top} \tag{5.20}$$

$$\gamma_i = \frac{\mathrm{Tr}\left[\boldsymbol{B}^{-1}(\boldsymbol{\Sigma}_{x_{i.}} + \hat{\boldsymbol{x}}_{.i}\hat{\boldsymbol{x}}_{.i}^{\top})\right]}{T} \tag{5.21}$$

To update $\boldsymbol{B}$ we use the same TMSBL update rule from (5.11), where in the context of estimating unknown parameters from the output, this update rule can be thought of as a sample covariance matrix of the elements in each row averaged over all the rows of $\boldsymbol{X}$. Putting these steps together we refer to the resulting algorithm by the MPDR-TSBL. It can be easily seen that the MPDR-TSBL algorithm derived is exactly equivalent to the TMSBL algorithm. This new perspective of the

TMSBL algorithm allows us to use the AMP based low complexity MPDR algorithm to propose a low complexity TMSBL algorithm. The low complexity TMSBL can be implemented simply by replacing the MPDR step from (5.13) by the GAMP MPDR from Table 4.1. We will show in the numerical analysis section how this proposed low complexity algorithm outperforms previously proposed AMP based MMV algorithms. Moreover, the complexity of the algorithm does not grow with increasing the number of time measurements, since we use a common MPDR receiver to process all columns of $\boldsymbol{X}$, in contrast to other AMP based methods that build a separate factor graph for each measurement and therefore their complexity grows as the number of measurements grows.

5.4 Numerical Analysis

5.4.1 Fast-SBL Numerical Results

In this section we present numerical studies to illustrate the performance improvement provided by using general priors with the fast-SBL algorithms as opposed to GSM priors. For the first experiment we study the performance of three algorithms proposed in section 5.2.3, namely Fast-MVDR-$\ell 1$, Fast-MVDR-Laplace and Fast-MVDR-BG. The algorithms will be compared against the original Fast-SBL algorithm from [71], and against the Fast-Laplace-SBL [34]. We set $N = 1000$, $M = 500$ and $SNR = 60$dB. We compute the NMSE for a range of K/N from 0.05 to 0.15 in Fig. 5.1.

From Fig. 5.1, we can see how the Bernoulli-Gaussian and Laplace priors provide performance enhancement compared to the GSM based fast-SBL and fast-Laplace-SBL algorithms.

In the second experiment we show how using general priors with the fast-SBL algorithm can be of value when some restrictions apply to $\boldsymbol{x}$. We enforce a non-negative constraint on $\boldsymbol{x}$, and based on that we propose a non-negative fast-SBL algorithm denoted by NN-Fast-SBL using a non-negative GSM prior. We also study the performance of the NN-Fast-SBL-BG algorithm

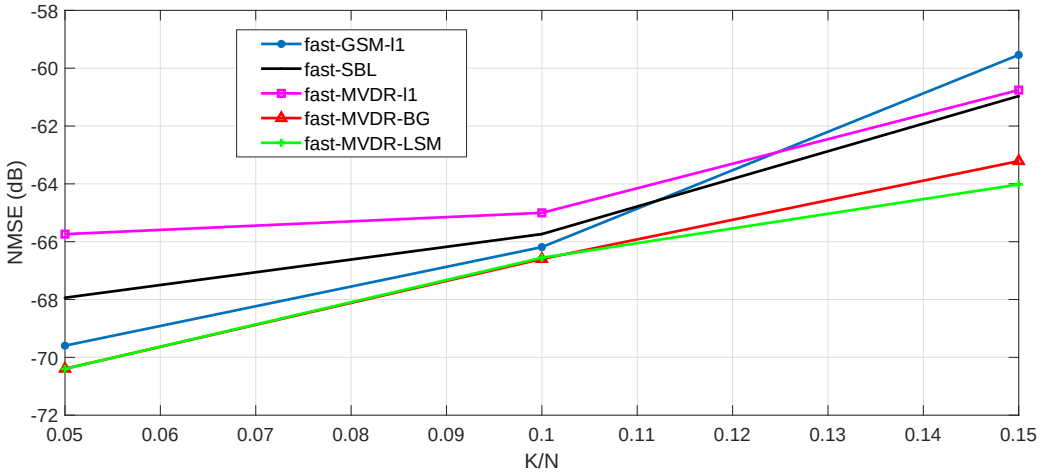


Figure 5.1: Performance of Fast SSR Algorithms

which uses a non-negative known Bernoulli-Gaussian prior. We compare both algorithms to the fast-SBL algorithm from [71]. We set $N = 1000$, $M = 500$ and $SNR = 60$dB. We compute the NMSE for a range of K/N from 0.05 to 0.2, we limit the comparison to this range since beyond this range all of the algorithms will fail.

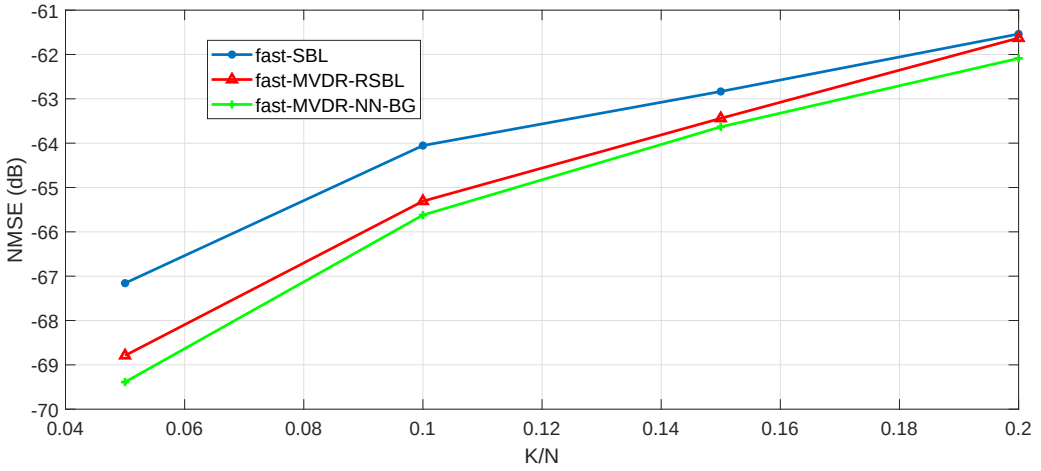


Figure 5.2: Performance of Non-Negative Fast SSR Algorithms

From the results in Fig. 5.2, we can see that imposing a non-negative GSM prior on $\boldsymbol{x}$ improves the recovery performance of the non-negative signal. Moreover, using the exact non-negative Bernoulli-Gaussian prior on $\boldsymbol{x}$ provides further performance improvement. These two algorithms serve as an example on how extra information about $\boldsymbol{x}$ can be incorporated into

the fast-SBL to achieve better results.

5.4.2 MMV MPDR-TSBL Numerical Results

In this subsection we will compare the performance and complexity of the low complexity MPDR-TSBL to the full complexity TMSBL algorithm. we will also compare MPDR-TSBL to two other AMP based MMV algorithms, namely the GGAMP-TSBL [66] and the AMP-MMV [65].

We will use the time-averaged normalized mean squared error (TNMSE) as a metric to evaluate the quality of the recovered MMV vectors, TNMSE $\triangleq \frac{1}{T}\sum_{t=1}^{T} \|\hat{\boldsymbol{x}}^{(t)} - \boldsymbol{x}^{(t)}\|^2 / \|\boldsymbol{x}^{(t)}\|^2$. We will also apply a complexity measure by tracking the time each algorithm requires to compute the final estimate, where we measure the time in seconds.

For the first experiment we study the performance of different algorithms versus the sparsity ratio K/N. The source matrix $\boldsymbol{x}$ was generated with K non-zero rows that are randomly chosen, the correlation between the non-zero elements in each row was modeled by an AR(1) process with a correlation coefficient $\rho = 0.95$. The problem dimension were set to $N = 1000$, $M = 500$ and $T = 4$. K was increased from 0 to M and the SNR set to SNR $= 60$dB. We choose to use an i.i.d. Gaussian $\boldsymbol{A}$. This choice is made to allow us to include the AMP-MMV algorithm in the comparison, because while the other algorithms can be shown to converge using non-i.i.d. Gaussian $\boldsymbol{A}$, AMP-MMV will diverge in such a case. Fig. 5.3 shows that as the number of non-zero elements increases, MPDR-TSBL experiences minimal performance degradation compared to the full complexity TMSBL algorithm. On the other hand, we can clearly see that the other two AMP based algorithms experience significant performance degradation compared to the TMSBL and MPDR-TSBL as K/N is increased. The MPDR-TSBL algorithm offers similar performance to the TMSBL algorithm with potentially significant complexity reduction, which will be shown next.

For the second experiment, we study the complexity advantage the MPDR-TSBL provides

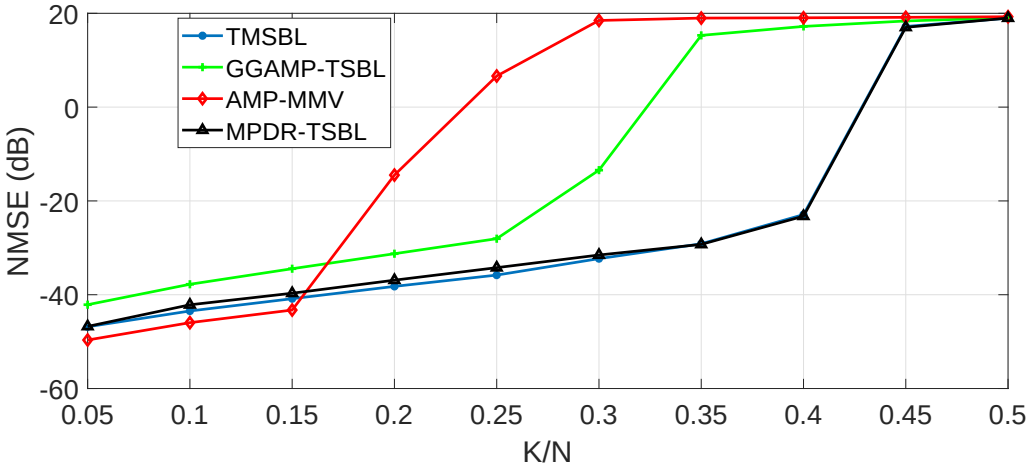


Figure 5.3: Performance of MMV Algorithms

over the TMSBL algorithm, and potentially over the other two AMP based algorithms. For this experiment we use the same generating model from the previous experiment, where we fix the ratio K/N at 0.2 and M/N at 0.5, and we increase N. To study the effect of the number of measurements on the complexity of different algorithms, we run two experiments, one with $T = 4$ time measurements, and another with $T = 20$.

In Fig. 5.4 we use solid lines for the results obtained using $T = 4$ and dashed lines for results obtained from $T = 20$. Fig. 5.4 shows the significant complexity reduction MPDR-TSBL provides over TMSBL. The TMSBL complexity was shown to be $O(NM^2)$ in [66] due to the matrix inverse computation in (5.9) and (5.8). While the complexity of the low complexity MPDR was found to be $O(NM)$ in [66] which explains the complexity difference. Another important observation is that the other two AMP algorithms have complexity orders of $O(TNM)$ because they were designed by building separate factor graphs for each time measurement. Both TMSBL and MPDR-TSBL apply a common MPDR step to all measurements, and therefore their complexities do not depend on T. This is clear in Fig. 5.4, where GGAMP-TSBL and AMP-MMV experienced increased runtimes as we increased the number of measurements, however the other two algorithms had minimal runtime increase with increased T.

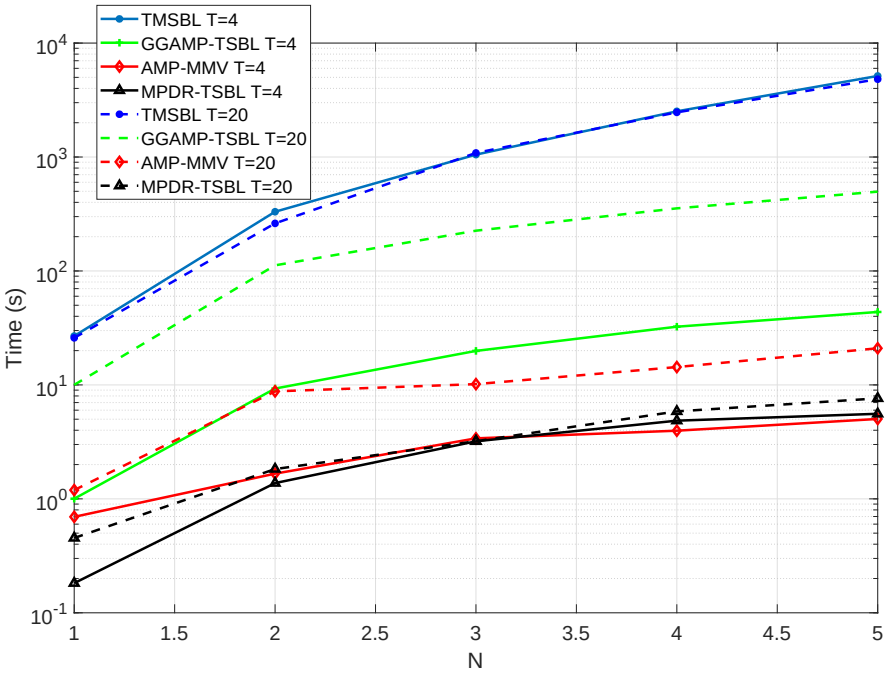


Figure 5.4: Complexity of MMV Algorithms

5.5 Conclusion

We showed how the SBL's MPDR interpretation from the previous results extends to other versions of the SBL algorithm, like the sequential fast-SBL and the MMV TMSBL algorithms. For the fast-SBL algorithm, we were able to enhance the algorithm's performance by incorporating extra information into the imposed priors. Where in the MMV case we demonstrated how the MPDR framework can decouple the problem in space transforming it into a much easier problem to solve. Based on this decoupling, we proposed a low complexity TMSBL algorithm that outperforms existing AMP based low complexity MMV algorithms.

5.6 Acknowledgment

This chapter, in full, is a reprint of material in Maher Al-Shoukairi, and Bhaskar D. Rao, "An MPDR Perspective and Extension of Sparse Bayesian Learning", under review at IEEE

Transactions on Signal Processing. I was the primary author and B. D. Rao supervised the research.

Chapter 6

Semi-Blind Channel Estimation in MIMO Systems with Discrete Priors on Data Symbols

6.1 Introduction

Obtaining accurate channel estimates is an important factor in achieving capacity gains in MIMO systems. Traditionally channel estimates were obtained using training pilots. However, as the number of users in a MIMO system increases, maintaining accurate channel estimates requires the length of the training pilots to increase, lowering the effective throughput. Semi-blind channel estimation uses data symbols in addition to the training symbols to improve channel estimates. This allows semi-blind techniques to have shorter training overhead and achieve better accuracy of channel estimates at the cost of increased processing at the receiver. Different techniques have been investigated for semi-blind channel estimation. In [93], the channel matrix was divided into a whitening matrix that is estimated using received data and a rotation matrix that is estimated using pilot symbols. In [94], a two level maximum likelihood (ML) optimization method is proposed

for channel and data estimation. In addition, a number of papers proposed to solve the problem using different formulations of the Expectation Maximization (EM) algorithm [70, 95–97]. In this letter we focus on the EM formulation that alternates between estimating the data symbols in the E-step and finding the best channel estimate in the M-step. To keep the algorithms tractable, in [97] a Gaussian prior on the data symbols is assumed, although the data symbols are drawn from a discrete distribution that represents the data constellation. To better represent the discrete distribution, in [70] a heuristic approach is proposed to demap the E-step's output to discrete symbols at each EM iteration. A Gaussian mixture model on data symbols is also considered to theoretically justify the approach in [70] while keeping the E-step tractable. The heuristic approach showed improvements in channel estimates at higher SNRs compared to the Gaussian prior case, while the Gaussian prior was superior in the low SNR region. This is due to the hard decisions made by method, which are much less accurate in low SNR. It was also shown in [97], how using the actual discrete prior on the data symbols results in exponential growth of complexity of the algorithm as the number of symbols in the data constellation increases deeming the approach not practical.

6.1.1 Chapter's Organization

The organization of the chapter is as follows. In Section 6.2, we describe the system model and discuss previous EM algorithms. In Section 6.3 we show how the dimentionality of the Gaussian EM algorithm can be reduced using the eigenvalue decomposition. In Section 6.4 we apply the MPDR decoupling principle to the E-step to incorporate the discrete prior that is based on the data constellation. Finally, in Section 6.5, we present numerical results to demonstrate the performance and complexity advantages achieved by the proposed algorithms.

6.2 System Model and Previous EM algorithms

We will generally follow the same system model and EM formulation from [70,97], where we will summarize it here for completeness. We consider the uplink transmission in a TDD multi-user MIMO system with a single serving base station (BS) equipped with M antennas. The BS serves K users with random locations within the cell, we also assume $M \geq K$. We consider an orthogonal division multiplexing system that is able to achieve flat fading over each of the carriers. Based on this model the channel matrix $\boldsymbol{G} \in \mathbb{C}^{M \times K}$ can be represented by:

$$\boldsymbol{G} = \boldsymbol{H}\boldsymbol{B}^{1/2}, \tag{6.1}$$

where $\boldsymbol{B}^{1/2} \in \mathbb{C}^{K \times K}$ is a diagonal matrix with diagonal elements β_k, that model large scale fading effects such as path loss and shadowing between each user and the BS. $\boldsymbol{H} \in \mathbb{C}^{M \times K}$ on the other hand models small scale fading effects. Those effects are modeled by circularly symmetric complex Gaussian columns of $\boldsymbol{H}$ with unit variance. For each user, an uplink transmission consists of N symbols, where the first L symbols are reserved for known pilot symbols, and the next $(N-L)$ symbols are unknown data symbols drawn randomly from a discrete constellation. At time n the received signal $\boldsymbol{y}^{(n)} \in \mathbb{C}^{M \times 1}$ at the BS is given by:

$$\boldsymbol{y}^{(n)} = \boldsymbol{G}\boldsymbol{s}^{(n)} + \boldsymbol{v}^{(n)}, \tag{6.2}$$

where $\boldsymbol{s}^{(n)} \in \mathbb{C}^{K \times 1}$ represents the transmit vector from K users at time n. When $n = [0, ..., L-1]$, $\boldsymbol{s}^{(n)}$ is a known pilot vector and when $n = [L, ..., N]$, $\boldsymbol{s}^{(n)}$ is an unknown data vector with $\mathbb{E}[\boldsymbol{s}^{(n)}\boldsymbol{s}^{(n)^H}] = \boldsymbol{I}_K$. $\boldsymbol{v}^{(n)}$ is the noise vector at time n with $\boldsymbol{v}^{(n)} \sim \mathcal{CN}(\boldsymbol{v}^{(n)}; 0, \sigma_v^2 \boldsymbol{I}_M)$. In the following we will refer to the group of pilot symbols $[\boldsymbol{s}^{(0)}, ..., \boldsymbol{s}^{(L-1)}]$ by the matrix $\boldsymbol{S}_p$. While data symbols are grouped in the matrix $\boldsymbol{S}_d = [\boldsymbol{s}^{(L)}, ..., \boldsymbol{s}^{(N)}]$. Similarly, at the receiver the first L received vectors corresponding to the pilot symbols are denoted by $\boldsymbol{Y}_p$ and the following $(N-L)$

vectors corresponding to the data symbols are denoted by $\boldsymbol{Y}_d$.

6.2.1 Pilot Based ML Receiver

The ML estimator of the channel matrix $\boldsymbol{G}$ based on the known pilot symbols $\boldsymbol{S}_p$ only, is given by [70, 97, 98]:

$$\hat{\boldsymbol{G}}_{ML}^{p} = (\boldsymbol{Y}_p \boldsymbol{S}_p^H)(\boldsymbol{S}_p \boldsymbol{S}_p^H)^{-1}, \tag{6.3}$$

where using orthogonal pilots that satisfy $\boldsymbol{S}_p \boldsymbol{S}_p^H = L\boldsymbol{I}_K$ minimizes the MSE of the channel estimate [98].

6.2.2 EM Semi-Blind Channel Estimation with Gaussian Prior

Previous work showed how the channel estimate can be improved by using the data symbols in addition to pilots to estimate the channel. A maximum likelihood estimate of the channel based on the full received symbols consisting of pilots and data, i.e. $\mathbf{Y} = [\mathbf{Y}_p, \mathbf{Y}_d]$, is given by:

$$\hat{\boldsymbol{G}}_{ML} = \underset{G}{\operatorname{argmax}} \log p(\boldsymbol{Y}|\boldsymbol{G}) \tag{6.4}$$

This ML estimate requires the maximization over all possible data symbol combinations. To avoid that, a number of iterative techniques were proposed to solve this problem, where we will focus on the EM based techniques in this letter. The EM algorithm can be used to address (6.4). An estimate of the channel can be obtained by:

$$\hat{\boldsymbol{G}}_{\ell+1} = \mathbb{E}_{p(\boldsymbol{S}_d|\boldsymbol{Y},\hat{\boldsymbol{G}}_\ell)}[\log p(\boldsymbol{Y}, \boldsymbol{S}_d|\boldsymbol{G})], \tag{6.5}$$

where ℓ is the EM iteration index. Based on (6.5), the EM algorithm iterates between an expectation step and a maximization step. The expectation step is given by:

$$\boldsymbol{\mu}_s^{(n)^\ell} = \mathbb{E}[\boldsymbol{s}^{(n)}|\hat{\boldsymbol{G}}_\ell, \boldsymbol{Y}], \quad \boldsymbol{\Sigma}_s^{(n)^\ell} = Cov[\boldsymbol{s}^{(n)}|\hat{\boldsymbol{G}}_\ell, \boldsymbol{Y}], \tag{6.6}$$

The maximization step is then given by [70]:

$$\begin{aligned}\hat{\boldsymbol{G}}_{\ell+1} = &\left(\boldsymbol{Y}_p \boldsymbol{S}_p^H + \sum_{n=L}^{N} \boldsymbol{y}^{(n)} \boldsymbol{\mu}_s^{(n)^{\ell H}}\right) \\ &\times \left(\boldsymbol{S}_p \boldsymbol{S}_p^H + \sum_{n=L}^{N} \left(\boldsymbol{\mu}_s^{(n)^\ell} \boldsymbol{\mu}_s^{(n)^{\ell H}}\right) + \boldsymbol{\Sigma}_s^{(n)^\ell}\right)^{-1}.\end{aligned} \tag{6.7}$$

Ideally the E-step should be executed based on a discrete prior that represents the constellation of the data symbols. It was shown in [97] that using the exact posterior directly in (6.6) grows the complexity exponentially with K and is computationally unfeasible. Therefore, approximate posteriors were previously proposed. The first approximation assumes a Gaussian distribution for the data symbols $\boldsymbol{s}^{(n)} \sim \mathcal{CN}(\boldsymbol{s}^{(n)}; 0, \boldsymbol{I}_K)$. This assumptions yields a closed form solution for the E-step as follows.

$$\begin{aligned}\boldsymbol{\mu}_s^{(n)^\ell} &= \left(\hat{\boldsymbol{G}}_\ell^H \hat{\boldsymbol{G}}_\ell + \sigma_v^2 \boldsymbol{I}_K\right)^{-1} \hat{\boldsymbol{G}}_\ell^H \boldsymbol{y}^{(n)} \\ \boldsymbol{\Sigma}_s^{(n)^\ell} &= \sigma_v^2 \left(\hat{\boldsymbol{G}}_\ell^H \hat{\boldsymbol{G}}_\ell + \sigma_v^2 \boldsymbol{I}_K\right)^{-1}\end{aligned} \tag{6.8}$$

While we will discuss other potential priors for the data symbols next, we point out that using different priors will only affect the E-step and will leave the M-step in (6.7) unchanged.

6.2.3 EM Semi-Blind Channel Estimation with Heuristic Demapping

To exploit the actual constellation of the data symbols, a heuristic approach to demap the output of the Gaussian E-step to the corresponding symbols was proposed in [70], resulting in the

following E-step:

$$\boldsymbol{\mu}_s^{(n)^\ell} = F\left(\left(\hat{\boldsymbol{G}}_\ell^H \hat{\boldsymbol{G}}_\ell + \sigma_v^2 \boldsymbol{I}_K\right)^{-1} \hat{\boldsymbol{G}}_\ell^H \boldsymbol{y}^{(n)}\right)$$
$$\boldsymbol{\Sigma}_s^{(n)^\ell} = \sigma_v^2 \left(\hat{\boldsymbol{G}}_\ell^H \hat{\boldsymbol{G}}_\ell + \sigma_v^2 \boldsymbol{I}_K\right)^{-1}, \tag{6.9}$$

where the function $F(.)$ performs an elementwise demapping of the estimates to their closest constellation points. In [70], an analytical justification was also provided for this approach by imposing a Gaussian mixture model to represent the data constellation, which provided similar results to the heuristic mapping approach. Therefore, in the numerical results section we will only consider the heuristic approach for comparison and not the Gaussian mixture algorithm.

6.3 Reduced dimensionality Gaussian EM algorithm

When the assumed prior on the data symbols is Gaussian, the ML optimization of the channel estimate (6.4) can be marginalized over the data symbols, resulting in the following ML optimization:

$$\hat{\boldsymbol{G}}_{ML} = \underset{G}{\operatorname{argmin}} \sum_{n=0}^{L-1} \frac{1}{\sigma_v^2} \|\boldsymbol{y}^{(n)} - \boldsymbol{G}\boldsymbol{s}^{(n)}\|^2$$
$$+ (N-L)\log|\Sigma_y| + \sum_{n=L}^{N} \boldsymbol{y}^{(n)^H} \Sigma_y^{-1} \boldsymbol{y}^{(n)}, \tag{6.10}$$

Where $\Sigma_y = (\boldsymbol{G}\boldsymbol{G}^H + \sigma_v^2 \boldsymbol{I}_M)$ is the estimate of the received data symbols covariance matrix. Using the fact that

$$\sum_{n=L}^{N} \boldsymbol{y}^{(n)^H} \Sigma_y^{-1} \boldsymbol{y}^{(n)} = \mathrm{Tr}(\Sigma_y^{-1} \boldsymbol{Y}_d \boldsymbol{Y}_d^H), \tag{6.11}$$

from (6.10) and (6.11) we can see how the channel estimate in the Gaussian prior case is a function of the covariance matrix of the received data symbols, rather than the actual received symbols themselves. Based on that, received data symbols can be replaced by a reduced dimensionality

representation with the same covariance matrix. The new representation of the received data symbols reduces the data matrix from a $M \times (N-L)$ to a $M \times M$ matrix. This can be achieved using the eigenvalue decomposition of the covariance matrix:

$$\tilde{\boldsymbol{Y}}_d = \boldsymbol{U}\boldsymbol{\Sigma}^{1/2}, \tag{6.12}$$

where $\boldsymbol{Y}_d\boldsymbol{Y}_d^H = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^H$ is the eigenvalue decomposition of $\boldsymbol{Y}_d\boldsymbol{Y}_d^H$. Based on this new representation $\boldsymbol{\mu}_s^{(n)^\ell}$ in (6.8) can be replaced by:

$$\boldsymbol{M}_d^\ell = \left(\hat{\boldsymbol{G}}_\ell^H \hat{\boldsymbol{G}}_\ell + \sigma_v^2 \boldsymbol{I}_K\right)^{-1} \hat{\boldsymbol{G}}_\ell^H \tilde{\boldsymbol{Y}}_d, \tag{6.13}$$

the sum $\sum_{n=L}^{N} \boldsymbol{y}^{(n)} \boldsymbol{\mu}_s^{(n)^{\ell H}}$ from (6.7) can be replaced by $\tilde{\boldsymbol{Y}}_d \boldsymbol{M}_d^{\ell H}$, and $\sum_{n=L}^{N} \boldsymbol{\mu}_s^{(n)^\ell} \boldsymbol{\mu}_s^{(n)^{\ell H}}$ can be replaced by $\boldsymbol{M}_d^\ell \boldsymbol{M}_d^{\ell H}$. This dimensionality reduction relaxes memory requirements to store the full data matrix, and speeds up the algorithm by reducing the complexity of the mean estimate in (6.8) for each EM iteration, leaving the final estimate unchanged.

6.4 MPDR Based Discrete Prior EM algorithm

In this section we show how the MPDR decoupling concept from [69] can be applied to the E-step of the semi-blind MIMO channel estimation. Because the MPDR decouples the measurements in space, we can compute a tractable E-step using a discrete prior for each $s_k^{(n)}$. The posterior mean $\boldsymbol{\mu}_s^{(n)^\ell}$ and the diagonal of the posterior covariance matrix $\boldsymbol{\Sigma}_s^{(n)^\ell}$ from (6.6), can be approximated using the actual discrete prior on the data symbols. The discrete prior on a transmitted data symbol from user k at time n is based on the data constellation. Given I complex

symbols in the data constellation, the prior on $s_k^{(n)}$ is:

$$p(s_k^{(n)}) = \sum_{i=1}^{I} \pi_i \delta\{s_k^{(n)} = a_i\}, \tag{6.14}$$

where a_is are the complex symbols of the constellation and π_i is the probability of each symbol. Based on the MPDR framework from [69], we first apply the MPDR receiver to the received data symbols:

$$r_k^{(n)^\ell} = \frac{\hat{\boldsymbol{g}}_{.k}^H \boldsymbol{\Sigma}_y^{-1} \boldsymbol{y}^{(n)}}{\hat{\boldsymbol{g}}_{.k}^H \boldsymbol{\Sigma}_y^{-1} \hat{\boldsymbol{g}}_{.k}} = s_k^{(n)} + n_k^{(n)} \tag{6.15}$$

$$\Sigma_{\tilde{r}_{kk}}^{\ell} = \frac{1}{\hat{\boldsymbol{g}}_{.k}^H \boldsymbol{\Sigma}_y^{-1} \hat{\boldsymbol{g}}_{.k}}, \quad \Sigma_{r_{kk}}^{\ell} = \frac{1}{\hat{\boldsymbol{g}}_{.k}^H \boldsymbol{\Sigma}_y^{-1} \hat{\boldsymbol{g}}_{.k}} - \mathbb{E}[{s_k^{(n)}}^2] \tag{6.16}$$

$$\boldsymbol{\Sigma}_y = (\hat{\boldsymbol{G}}_\ell {\hat{\boldsymbol{G}}_\ell}^H + \sigma_v^2 \boldsymbol{I}_M), \tag{6.17}$$

where $\hat{\boldsymbol{g}}_{.k}$ is the kth column of the matrix $\hat{\boldsymbol{G}}_\ell$. $r_k^{(n)^\ell}$ is the k^{th} element of the output of the MPDR receiver applied to $\boldsymbol{y}^{(n)}$ and $n_k^{(n)}$ is the associated output noise. $\Sigma_{\hat{r}_{kk}}^{\ell}$ represents the current estimate of the total signal and noise power out of the MPDR and $\Sigma_{r_{kk}}^{\ell}$ is the current estimate of the noise power. Based on the assumption that $\boldsymbol{r}^{(n)^\ell}$ decouples the measurements into the transmitted symbol vector $\boldsymbol{s}^{(n)}$ with AWGN noise [69], we can perform element wise MMSE estimation on each of the $s_k^{(n)}$ symbols as follows:

$$\mu_{s_k}^{(n)^\ell} \approx \mathbb{E}[s_k^{(n)} | r_k^{(n)^\ell}] = \frac{\sum_{i=1}^{I} \pi_i a_i \zeta_{ik}^{(n)^\ell}}{\sum_{i=1}^{I} \pi_i \zeta_{ik}^{(n)^\ell}} \tag{6.18}$$

$$\begin{aligned} \Sigma_{s_{kk}}^{(n)^\ell} \approx Cov[s_k^{(n)} | r_k^{(n)^\ell}] &= \frac{\sum_{i=1}^{I} \pi_i |a_i|^2 \zeta_{ik}^{(n)^\ell}}{\sum_{i=1}^{I} \pi_i \zeta_{ik}^{(n)^\ell}} \\ &\quad - \frac{(\sum_{i=1}^{I} \pi_i a_i \zeta_{ik}^{(n)^\ell})(\sum_{i=1}^{I} \pi_i a_i^H \zeta_{ik}^{(n)^\ell})}{(\sum_{i=1}^{I} \pi_i \zeta_{ik}^{(n)^\ell})^2}, \end{aligned} \tag{6.19}$$

where $\zeta_{ik}^{(n)^\ell} = \exp(-\frac{1}{\Sigma_{r_{kk}}^\ell}|r_k^{(n)^\ell} - a_i|^2)$. This E-step approximation with the discrete prior from (6.14) produces estimates of only the diagonal of the posterior covariance matrix $\mathbf{\Sigma}_s^{(n)^\ell}$. However, in the numerical analysis section it was found that we are still able to obtain good estimates of the channel by approximating $\mathbf{\Sigma}_s^{(n)^\ell}$ with a diagonal matrix. This E-step formulation has an advantage over the Gaussian prior E-step because it uses the actual discrete data constellation. Moreover, unlike (6.9) the E-step presented here produces soft outputs, which as we will see in the next section improves the performance for low SNR values. However, because the soft outputs do not make a quick decision on the symbols, using the proposed E-step can result in a larger number of iterations before convergence, which can potentially slow down the algorithm. To resolve this issue we propose using a combination of initializing the algorithm with a partial estimate obtained using the reduced dimensionality Gaussian prior from section 6.3 and then performing few iterations using the discrete prior based E-step.

6.5 Numerical Results

Similar to [70], we will setup our numerical model with a single cell equipped with a BS at its center. The radius of the cell is 500m with users uniformly distributed within the cell. We define the SNR to be $SNR = \frac{\mathbb{E}[\beta_k]}{\sigma_v^2}$. We use the large scale fading model from [97] and We use a QPSK constellation for both pilot and data symbols.

6.5.1 Performance of the Proposed Algorithm

We will examine the performance of the MPDR based proposed algorithm against the algorithms from [70], which are the EM algorithm with Gaussian prior, and the EM algorithm with heuristic demapping. In addition to that we include the ML estimator based on pilots only, and a genie ML estimator that is given the full transmitted data symbols. We will use the scaled mean squared error (MSE) given by $\mathbb{E}[\|\boldsymbol{G} - \hat{\boldsymbol{G}}\|_F^2]/\mathbb{E}[\beta_k]$ as a measure of the quality of the channel

estimate. For the first experiment we set $M = 8$, $K = 4$, $L = 16$ and $N = 512$. For the second experiment we set $M = 16$, $K = 8$, $L = 16$ and $N = 1024$.

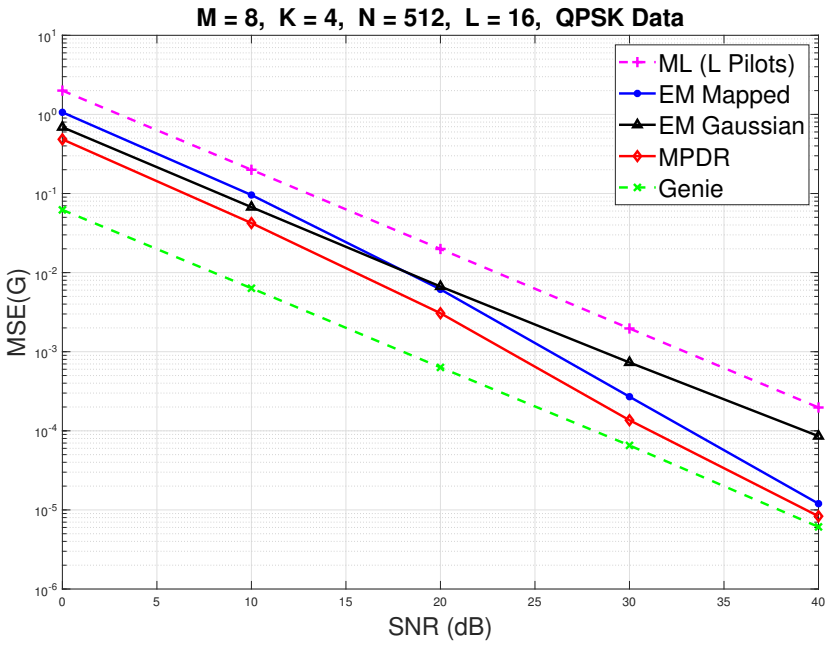


Figure 6.1: MSE vs SNR (First Experiment)

From Fig. 6.1 and Fig. 6.2 we can see that the proposed MPDR based algorithm that uses the actual data constellation provides significant improvement over the pilot only ML estimation. It is also clear that the algorithm outperforms the EM Gaussian and heuristic algorithms for both low and high SNR regions.

6.5.2 Runtime of the Proposed Algorithm

In this section we study the runtime of the reduced dimensionality Gaussian algorithm, and the runtime of the proposed MPDR based algorithm. We will use the same parameters from the experiments in the previous section, and we will track the time needed by each algorithm to converge in seconds.

From Fig. 6.3 and Fig. 6.4 we can see how reducing the dimensionality of the EM

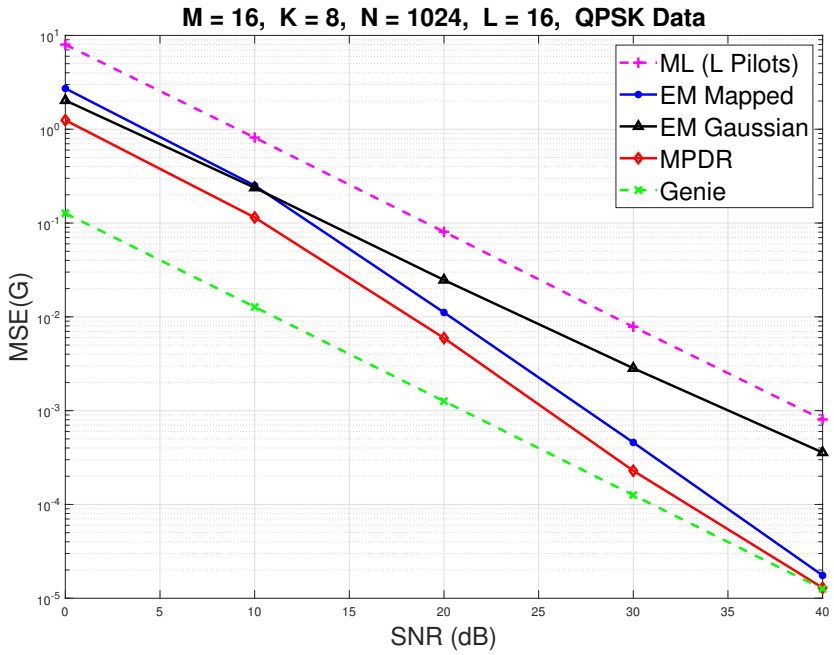


Figure 6.2: MSE vs SNR (Second Experiment)

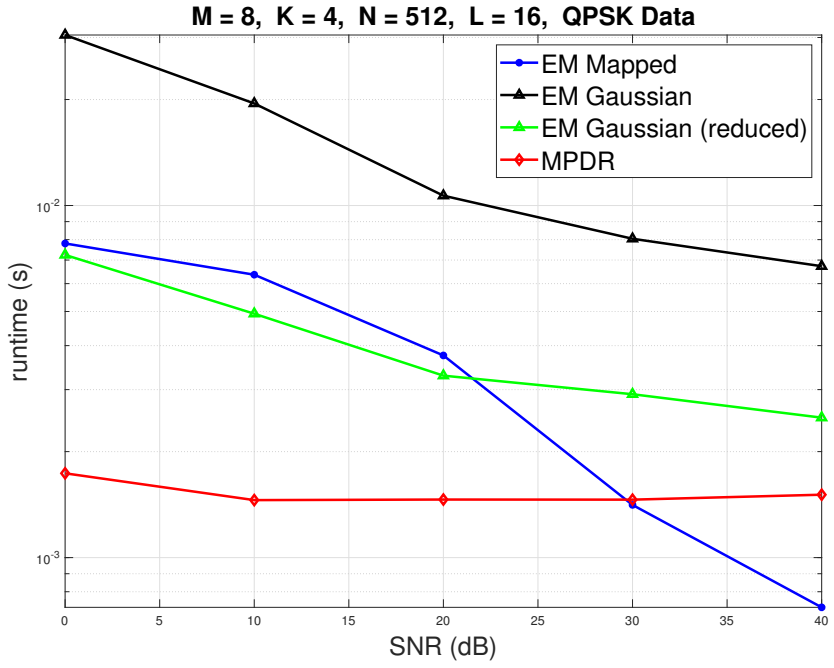


Figure 6.3: Runtime vs SNR (first Experiment)

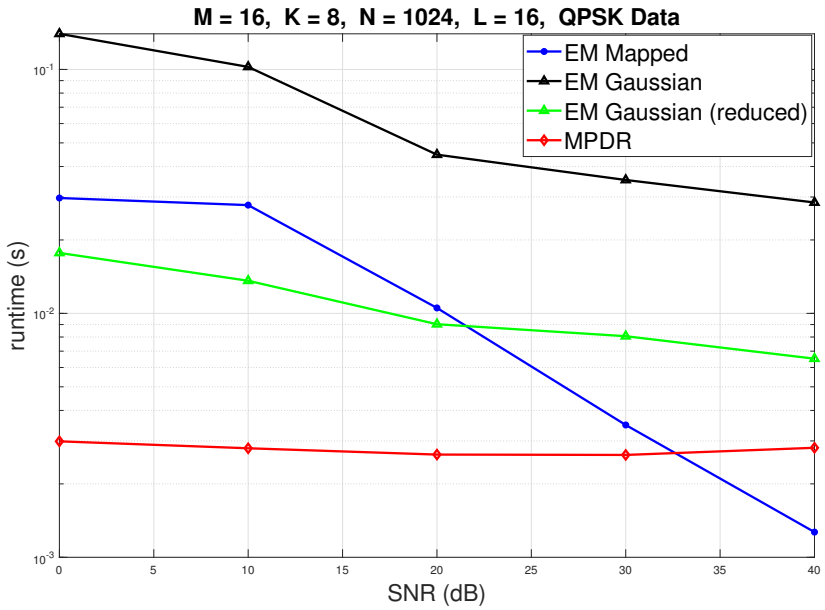


Figure 6.4: Runtime vs SNR (Second Experiment)

Gaussian algorithm can reduce the runtime in addition to reducing memory requirements for storing the data. The reduction in runtime is more significant when the number of data symbols is increased. We can also see in the figures that initializing the proposed MPDR algorithm with some reduced dimensionality EM Gaussian iterations and then performing few MPDR based iterations produces an algorithm with very efficient runtimes. The algorithm generally has faster runtimes than the other algorithms, except for the high SNR case, where the heuristic approach runs faster. This is due to the heuristic algorithm's ability to make good hard decisions on the data symbols at that SNR level.

6.6 Conclusion

We addressed the semi-blind channel estimation problem in MIMO systems, where we proposed an eigenvalue decomposition based technique to reduce the dimensionality of the EM algorithm with a Gaussian prior. The reduction is particularly beneficial when a large number of

data symbols is available. We also proposed another EM based algorithm with tractable MPDR based E-step and a discrete prior on the data symbols. The proposed MPDR based discrete prior algorithm outperforms previously used EM based algorithms in all SNR regions. We also showed how proper initialization of the MPDR based algorithm can reduce its runtime by reducing the number of iterations required for its convergence.

6.7 Acknowledgment

This chapter, in full, is a reprint of material in Maher Al-Shoukairi, and Bhaskar D. Rao, "Semi-Blind Channel Estimation In MIMO Systems with Discrete Priors On Data Symbols", under review at IEEE Transactions on Signal Processing. I was the primary author and B. D. Rao supervised

Bibliography

[1] D. L. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[2] E. J. Candes and T. Tao, "Near-optimal signal recovery from random projections: Universal encoding strategies?," *IEEE Transactions on Information Theory*, vol. 52, no. 12, pp. 5406–5425, 2006.

[3] R. G. Baraniuk, "Compressive sensing," *IEEE signal processing magazine*, vol. 24, no. 4, pp. 118–121, 2007.

[4] M. Elad, *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*. Springer, 2010.

[5] S. Foucart and H. Rauhut, *A Mathematical Introduction to Compressive Sensing (Applied and Numerical Harmonic Analysis)*. Birkhäuser, 2013.

[6] Y. C. Eldar and G. Kutyniok, eds., *Compressed Sensing: Theory and Applications*. Cambridge University Press, 2012.

[7] I. F. Gorodnitsky, J. S. George, and B. D. Rao, "Neuromagnetic source imaging with FOCUSS: a recursive weighted minimum norm algorithm," *Electroencephalography and clinical Neurophysiology*, vol. 95, no. 4, pp. 231–251, 1995.

[8] S. Makeig, C. Kothe, T. Mullen, N. Bigdely-Shamlo, Z. Zhang, and K. Kreutz-Delgado, "Evolving signal processing for brain–computer interfaces," *Proceedings of the IEEE*, vol. 100, no. Special Centennial Issue, pp. 1567–1584, 2012.

[9] D. Malioutov, M. Cetin, and A. S. Willsky, "A sparse signal reconstruction perspective for source localization with sensor arrays," *IEEE transactions on signal processing*, vol. 53, no. 8, pp. 3010–3022, 2005.

[10] J. M. Kim, O. K. Lee, and J. C. Ye, "Compressive MUSIC: Revisiting the link between compressive sensing and array signal processing," *IEEE Transactions on Information Theory*, vol. 58, no. 1, pp. 278–301, 2012.

[11] D. Wipf and S. Nagarajan, "Beamforming using the relevance vector machine," in *Proceedings of the 24th international conference on Machine learning*, pp. 1023–1030, 2007.

[12] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, no. 2, pp. 210–227, 2008.

[13] D. Giacobello, M. G. Christensen, M. N. Murthi, S. H. Jensen, and M. Moonen, "Sparse linear prediction and its applications to speech processing," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 5, pp. 1644–1657, 2012.

[14] J. Haupt, W. U. Bajwa, M. Rabbat, and R. Nowak, "Compressed sensing for networked data," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 92–101, 2008.

[15] M. F. Duarte, G. Shen, A. Ortega, and R. G. Baraniuk, "Signal compression in wireless sensor networks," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 370, no. 1958, pp. 118–135, 2012.

[16] J. Mo, P. Schniter, and R. W. Heath, "Channel estimation in broadband millimeter wave MIMO systems with few-bit adcs," *IEEE Transactions on Signal Processing*, vol. 66, no. 5, pp. 1141–1154, 2017.

[17] Y. Ding, S.-E. Chiu, and B. D. Rao, "Bayesian channel estimation algorithms for massive MIMO systems with hybrid analog-digital processing and low-resolution ADCs," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 3, pp. 499–513, 2018.

[18] B. K. Natarajan, "Sparse approximate solutions to linear systems," *SIAM Journal on Computing*, vol. 24, no. 2, pp. 227–234, 1995.

[19] B. D. Rao, K. Engan, S. F. Cotter, J. Palmer, and K. Kreutz-Delgado, "Subset selection in noise based on diversity measure minimization," *IEEE transactions on Signal processing*, vol. 51, no. 3, pp. 760–770, 2003.

[20] D. P. Wipf, B. D. Rao, and S. Nagarajan, "Latent variable Bayesian models for promoting sparsity," *IEEE Transactions on Information Theory*, vol. 57, no. 9, pp. 6236–6255, 2011.

[21] D. L. Donoho, "For most large underdetermined systems of linear equations the minimal ℓ_1-norm solution is also the sparsest solution," *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, vol. 59, no. 6, pp. 797–829, 2006.

[22] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[23] E. J. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Transactions on information theory*, vol. 52, no. 2, pp. 489–509, 2006.

[24] E. J. Candes and T. Tao, "Decoding by linear programming," *IEEE transactions on Information Theory*, vol. 51, no. 12, pp. 4203–4215, 2005.

[25] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Transactions on Information Theory*, vol. 53, no. 12, pp. 4655–4666, 2007.

[26] D. L. Donoho, Y. Tsaig, I. Drori, and J.-L. Starck, "Sparse solution of underdetermined systems of linear equations by stagewise orthogonal matching pursuit," *IEEE transactions on Information Theory*, vol. 58, no. 2, pp. 1094–1121, 2012.

[27] M. A. Figueiredo, J. M. Bioucas-Dias, and R. D. Nowak, "Majorization-minimization algorithms for wavelet-based image restoration," *IEEE Transactions on Image processing*, vol. 16, no. 12, pp. 2980–2991, 2007.

[28] E. J. Candes, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted ℓ_1 minimization," *Journal of Fourier Analysis and Applications*, vol. 14, no. 5-6, pp. 877–905, 2008.

[29] R. Chartrand and W. Yin, "Iteratively reweighted algorithms for compressive sensing," in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3869–3872, IEEE, 2008.

[30] M. E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *J. Mach. Learn. Res.*, vol. 1, pp. 211–244, Sept. 2001.

[31] D. P. Wipf and B. D. Rao, "Sparse Bayesian learning for basis selection," *IEEE Transactions on Signal Processing*, vol. 52, pp. 2153–2164, Aug 2004.

[32] S. Rangan, P. Schniter, and A. K. Fletcher, "Vector approximate message passing," *IEEE Transactions on Information Theory*, vol. 65, no. 10, pp. 6664–6684, 2019.

[33] J. P. Vila and P. Schniter, "Expectation-maximization Gaussian-mixture approximate message passing," *IEEE Transactions on Signal Processing*, vol. 61, no. 19, pp. 4658–4672, 2013.

[34] S. D. Babacan, R. Molina, and A. K. Katsaggelos, "Bayesian compressive sensing using Laplace priors," *IEEE Transactions on Image Processing*, vol. 19, no. 1, pp. 53–63, 2010.

[35] L. He and L. Carin, "Exploiting structure in wavelet-based Bayesian compressive sensing," *IEEE Transactions on Signal Processing*, vol. 57, no. 9, pp. 3488–3497, 2009.

[36] S. Ji, Y. Xue, and L. Carin, "Bayesian compressive sensing," *IEEE Transactions on Signal Processing*, vol. 56, no. 6, pp. 2346–2356, 2008.

[37] D. Baron, S. Sarvotham, and R. G. Baraniuk, "Bayesian compressive sensing via belief propagation," *IEEE Transactions on Signal Processing*, vol. 58, no. 1, pp. 269–280, 2010.

[38] D. Wipf and S. Nagarajan, "Iterative reweighted ℓ_1 and ℓ_2 methods for finding sparse solutions," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 4, no. 2, pp. 317–329, 2010.

[39] R. Giri and B. Rao, "Type I and type II Bayesian methods for sparse signal recovery using scale mixtures," *IEEE Transactions on Signal Processing*, vol. 64, no. 13, pp. 3418–3428, 2016.

[40] R. Giri, *Bayesian sparse signal recovery using scale mixtures with applications to speech*. PhD thesis, UC San Diego, 2016.

[41] D. P. Wipf, *Bayesian methods for finding sparse representations*. PhD thesis, UC San Diego, 2006.

[42] P. Gustafson and S. Bose, "Aspects of Bayesian robustness in hierarchical models [with discussion and rejoinder]," *Lecture Notes-Monograph Series*, pp. 63–80, 1996.

[43] P. K. Goel and M. H. Degroot, "Information about hyperparameters in hierarchical models," *Journal of the American Statistical Association*, vol. 76, no. 373, pp. 140–147, 1981.

[44] D. F. Andrews and C. L. Mallows, "Scale mixtures of normal distributions," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 99–102, 1974.

[45] J. Palmer, K. Kreutz-Delgado, B. D. Rao, and D. P. Wipf, "Variational EM algorithms for non-Gaussian latent variable models," in *Advances in Neural Information Processing Systems*, pp. 1059–1066, 2005.

[46] K. Lange and J. S. Sinsheimer, "Normal/independent distributions and their applications in robust regression," *Journal of Computational and Graphical Statistics*, vol. 2, no. 2, pp. 175–198, 1993.

[47] J. A. Palmer, *Variational and scale mixture representations of non-Gaussian densities for estimation in the Bayesian linear model: Sparse coding, independent component analysis, and minimum entropy segmentation*. PhD thesis, UC San Diego, 2006.

[48] N. L. Pedersen, C. N. Manchón, M.-A. Badiu, D. Shutin, and B. H. Fleury, "Sparse estimation using Bayesian hierarchical prior modeling for real and complex linear models," *Signal processing*, vol. 115, pp. 94–109, 2015.

[49] D. L. Donoho, A. Maleki, and A. Montanari, "Message passing algorithms for compressed sensing: I. motivation and construction," in *Information Theory (ITW 2010, Cairo), 2010 IEEE Information Theory Workshop on*, pp. 1–5, Jan 2010.

[50] S. Rangan, "Generalized approximate message passing for estimation with random linear mixing," in *Information Theory Proceedings (ISIT), 2011 IEEE International Symposium on*, pp. 2168–2172, July 2011.

[51] M. Bayati and A. Montanari, "The dynamics of message passing on dense graphs, with applications to compressed sensing," *IEEE Transactions on Information Theory*, vol. 57, no. 2, pp. 764–785, 2011.

[52] S. Rangan, P. Schniter, and A. Fletcher, "On the convergence of approximate message passing with arbitrary matrices," in *Information Theory (ISIT), 2014 IEEE International Symposium on*, pp. 236–240, June 2014.

[53] F. Caltagirone, L. Zdeborova, and F. Krzakala, "On convergence of approximate message passing," in *Information Theory (ISIT), 2014 IEEE International Symposium on*, pp. 1812–1816, June 2014.

[54] J. Vila, P. Schniter, S. Rangan, F. Krzakala, and L. Zdeborová, "Adaptive damping and mean removal for the generalized approximate message passing algorithm," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pp. 2021–2025, April 2015.

[55] A. Manoel, F. Krzakala, E. Tramel, and L. Zdeborovà, "Swept approximate message passing for sparse estimation," in *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pp. 1123–1132, 2015.

[56] S. Rangan, A. K. Fletcher, P. Schniter, and U. S. Kamilov, "Inference for generalized linear models via alternating directions and bethe free energy minimization," in *2015 IEEE International Symposium on Information Theory (ISIT)*, pp. 1640–1644, IEEE, 2015.

[57] S. F. Cotter, B. D. Rao, K. Engan, and K. Kreutz-Delgado, "Sparse solutions to linear inverse problems with multiple measurement vectors," *IEEE Transactions on Signal Processing*, vol. 53, no. 7, pp. 2477–2488, 2005.

[58] Y. C. Eldar and M. Mishali, "Robust recovery of signals from a structured union of subspaces," *IEEE Transactions on Information Theory*, vol. 55, no. 11, pp. 5302–5316, 2009.

[59] Y. Jin and B. D. Rao, "Insights into the stable recovery of sparse solutions in overcomplete representations using network information theory," in *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3921–3924, IEEE, 2008.

[60] D. P. Wipf and B. D. Rao, "An empirical Bayesian strategy for solving the simultaneous sparse approximation problem," *Signal Processing, IEEE Transactions on*, vol. 55, no. 7, pp. 3704–3716, 2007.

[61] Z. Zhang and B. D. Rao, "Sparse signal recovery with temporally correlated source vectors using sparse Bayesian learning," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 5, no. 5, pp. 912–926, 2011.

[62] Z. Zhang, *Sparse signal recovery exploiting spatiotemporal correlation*. PhD thesis, UC San Diego, 2012.

[63] R. Prasad, *Sparse Bayesian Learning For Joint Channel Estimation Data Detection In OFDM Systems*. PhD thesis, 2018.

[64] R. Prasad, C. R. Murthy, and B. D. Rao, "Joint approximately sparse channel estimation and data detection in OFDM systems using sparse Bayesian learning," *IEEE Transactions on Signal Processing*, vol. 62, no. 14, pp. 3591–3603, 2014.

[65] J. Ziniel and P. Schniter, "Efficient high-dimensional inference in the multiple measurement vector problem," *Signal Processing, IEEE Transactions on*, vol. 61, no. 2, pp. 340–354, 2013.

[66] M. Al-Shoukairi, P. Schniter, and B. D. Rao, "A GAMP based low complexity sparse Bayesian learning algorithm," *arXiv preprint arXiv:1703.03044*, 2017.

[67] M. Al-Shoukairi and B. Rao, "Sparse Bayesian learning using approximate message passing," in *Signals, Systems and Computers, 2014 48th Asilomar Conference on*, pp. 1957–1961, IEEE, 2014.

[68] H. L. Van Trees, *Optimum array processing: Part IV of detection, estimation, and modulation theory*. John Wiley & Sons, 2004.

[69] M. Al-Shoukairi and B. D. Rao, "Sparse signal recovery using MPDR estimation," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5047–5051, IEEE, 2019.

[70] E. Nayebi and B. D. Rao, "Semi-blind channel estimation in massive MIMO systems with different priors on data symbols," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3879–3883, IEEE, 2018.

[71] M. E. Tipping, A. Faul, *et al.*, "Fast marginal likelihood maximisation for sparse Bayesian models," in *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, 2003.

[72] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer, 2006.

[73] G. McLachlan and T. Krishnan, *The EM algorithm and extensions*, vol. 382. John Wiley & Sons, 2007.

[74] C. J. Wu, "On the convergence properties of the EM algorithm," *The Annals of statistics*, pp. 95–103, 1983.

[75] S. Rangan, P. Schniter, E. Riegler, A. Fletcher, and V. Cevher, "Fixed points of generalized approximate message passing with arbitrary matrices," in *Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on*, pp. 664–668, IEEE, 2013.

[76] Z. Zhang and B. D. Rao, "Sparse signal recovery in the presence of correlated multiple measurement vectors," in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3986–3989, IEEE, 2010.

[77] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Transactions on information theory*, vol. 47, no. 2, pp. 498–519, 2001.

[78] C.-J. Lin, "Projected gradient methods for nonnegative matrix factorization," *Neural computation*, vol. 19, no. 10, pp. 2756–2779, 2007.

[79] J. P. 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, 2014.

[80] A. Nalci, I. Fedorov, and B. D. Rao, "Rectified Gaussian scale mixtures and the sparse non-negative least squares problem," *arXiv preprint arXiv:1601.06207*, 2016.

[81] A. Javanmard and A. Montanari, "State evolution for general approximate message passing algorithms, with applications to spatial coupling," *Information and Inference*, p. iat004, 2013.

[82] D. L. Donoho and I. M. Johnstone, "Minimax risk over ℓ_p-balls forl ℓ_p-error," *Probability Theory and Related Fields*, vol. 99, no. 2, pp. 277–303, 1994.

[83] P. Stoica, D. Zachariah, and J. Li, "Weighted SPICE: A unifying approach for hyperparameter-free sparse estimation," *Digital Signal Processing*, vol. 33, pp. 1–12, 2014.

[84] P. Garrigues and B. A. Olshausen, "Group sparse coding with a Laplacian scale mixture prior," in *Advances in neural information processing systems*, pp. 676–684, 2010.

[85] P. Gerstoft, C. F. Mecklenbräuker, A. Xenaki, and S. Nannuru, "Multisnapshot sparse bayesian learning for doa," *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1469–1473, 2016.

[86] T. Yardibi, J. Li, P. Stoica, M. Xue, and A. B. Baggeroer, "Source localization and sensing: A nonparametric iterative adaptive approach based on weighted least squares," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 46, no. 1, pp. 425–443, 2010.

[87] F. Bellili, F. Sohrabi, and W. Yu, "Generalized approximate message passing for massive mimo mmwave channel estimation with laplacian prior," *IEEE Transactions on Communications*, vol. 67, no. 5, pp. 3205–3219, 2019.

[88] J. Vila and P. Schniter, "Expectation-maximization bernoulli-gaussian approximate message passing," in *2011 Conference Record of the Forty Fifth Asilomar Conference on Signals, Systems and Computers (ASILOMAR)*, pp. 799–803, IEEE, 2011.

[89] A. Kammoun, M. Kharouf, W. Hachem, and J. Najim, "A central limit theorem for the SINR at the LMMSE estimator output for large-dimensional signals," *IEEE Transactions on Information Theory*, vol. 55, no. 11, pp. 5048–5063, 2009.

[90] J. Zhang, E. K. Chong, and D. N. C. Tse, "Output MAI distributions of linear MMSE multiuser receivers in DS-CDMA systems," *IEEE Transactions on Information Theory*, vol. 47, no. 3, pp. 1128–1144, 2001.

[91] Y.-C. Liang, G. Pan, and Z. Bai, "Asymptotic performance of MMSE receivers for large systems using random matrix theory," *IEEE Transactions on Information Theory*, vol. 53, no. 11, pp. 4173–4190, 2007.

[92] D. Guo, S. Verdú, and L. K. Rasmussen, "Asymptotic normality of linear multiuser receiver outputs," *IEEE transactions on information theory*, vol. 48, no. 12, pp. 3080–3095, 2002.

[93] A. K. Jagannatham and B. D. Rao, "Whitening-rotation-based semi-blind MIMO channel estimation," *IEEE Transactions on Signal Processing*, vol. 54, no. 3, pp. 861–869, 2006.

[94] M. Abuthinien, S. Chen, and L. Hanzo, "Semi-blind joint maximum likelihood channel estimation and data detection for mimo systems," *IEEE Signal Processing Letters*, vol. 15, pp. 202–205, 2008.

[95] C. H. Aldana, E. de Carvalho, and J. M. Cioffi, "Channel estimation for multicarrier multiple input single output systems using the EM algorithm," *IEEE Transactions on Signal Processing*, vol. 51, no. 12, pp. 3280–3292, 2003.

[96] X. Wautelet, C. Herzet, A. Dejonghe, J. Louveaux, and L. Vandendorpe, "Comparison of em-based algorithms for mimo channel estimation," *IEEE Transactions on communications*, vol. 55, no. 1, pp. 216–226, 2007.

[97] E. Nayebi and B. D. Rao, "Semi-blind channel estimation for multiuser massive MIMO systems," *IEEE Transactions on Signal Processing*, vol. 66, no. 2, pp. 540–553, 2017.

[98] S. D. Silverstein, "Application of orthogonal codes to the calibration of active phased array antennas for communication satellites," *IEEE transactions on signal processing*, vol. 45, no. 1, pp. 206–218, 1997.