Signal Recovery for Quantized Compressed Sensing

Osman Musa

Institute of Telecommunications
Technische Universität Wien

This dissertation is submitted for the degree of
Doktor der technischen Wissenschaften

July 2019

Advisor

Univ.-Prof. Dipl.-Ing. Dr.-Ing. Norbert Görtz
Institute of Telecommunications
Technische Universität Wien
Austria

Examiners

Prof. Dr.-Ing. Volker Kühn
Institute of Communications Engineering
Universität Rostock
Germany

Ao. Univ.-Prof. Dipl.-Ing. Dr. techn. Gerald Matz
Institute of Telecommunications
Technische Universität Wien
Austria

Abstract

Many systems, including telecommunication systems, radar and imaging systems, biomedical systems, control and robotics systems, rely on powerful *digital signal processing* (DSP). DSP algorithms are hard pressed to provide accurate estimates of a signal from as few as possible noisy measurements. If the signal to be estimated is sparse and high dimensional, a novel DSP technique, called *compressed sensing* (CS), allows efficient recovery from (possibly noisy) low dimensional representation. Even though reconstruction guarantees of several CS recovery algorithms have been known for almost a decade, many nonlinear distortions introduced by a practical measurement system are often not considered in the analysis. Neglecting these distortions could, in turn, have a detrimental effect on the performance of a recovery algorithm in a practical application. In this thesis, I focus on algorithms for recovering sparse vectors from measurements tampered with some of the most common nonlinear distortions that appear in practice, namely quantization and modulo distortions, which are not treated with classical CS recovery algorithms.

To the present date, many reconstruction algorithms have been proposed to solve noisy CS problems. Among them, the class of *approximate message passing* (AMP) algorithms stands out for its low computational complexity, low reconstruction error, and the ability to predict the states of the algorithm across iterations (at least in the large system limit). Furthermore, the *Bayesian approximate message passing* (BAMP) algorithm has the ability to incorporate signal prior to additionally improve the estimate, while the *generalized approximate message passing* (GAMP) algorithm allows for the reconstruction of sparse signals from nonlinear measurements. These facts make the AMP algorithms particularly interesting for our problems involving quantization and modulo distortions with known prior.

BAMP follows the probabilistic estimation approach where a prior distribution is assumed for the unknown signal. A commonly used family of distributions that promotes sparsity of the solution is the *Bernoulli-Gauss* (BG) mixture. In this thesis, I show how a BG mixture can be thought of as a limiting case of mixture of two Gaussian distributions. I extend this to the case where the prior is modeled as a mixture of n

Gaussian distributions, which covers a much larger class of signals. For that case I derive explicit analytic update rules for the BAMP algorithm. Furthermore, I present a novel approach to the known *Analysis-by-Synthesis* (AbS) quantization scheme, where I use the BAMP algorithm to further reduce the end-to-end reconstruction MSE from quantized CS measurements.

In many practical applications, the computationally demanding AbS quantization scheme is not feasible, and CS measurements are simply scalar quantized. During the storage or communication over a noisy channel, the quantized measurements might be corrupted in different ways. Reconstruction by conventional algorithms on such highly distorted measurements will result in poor accuracy. To address these problems, I use the well established GAMP algorithm and tailor it for scalar quantized CS measurements corrupted with noise. I provide analytical expressions for the nonlinear updates assuming different noise models, and conduct numerical experiments to show that the GAMP algorithm outperforms conventional CS algorithms under the considered model assumptions.

Finally, I consider a problem that typically appears in the context of calibration of sensing devices: unknown dynamic range of the input signal. Traditionally, this problem was addressed by clipping or saturating the input, which results in a loss of information about the signal. Alternatively, by taking modulo measurements, the input samples exceeding the sensor's threshold are simply folded back to its dynamic range. Even though the sampling theory for recovering a sparse signal from its low-pass filtered version and modulo measurements already exists, in this thesis, I investigate the application of the GAMP algorithm for recovering a sparse signal from modulo samples of noisy randomized projections of the unknown signal.

Declaration

I hereby declare that, except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification at this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the references and acknowledgements.

Osman Musa
July 2019

Acknowledgements

First of all, I wish to express my sincere gratitude to my advisor Norbert Görtz, for giving me the chance to work in his research group, for his continuous support and encouragement over the past five years. I would also like to thank Norbert, for trusting in my abilities and allowing me to pursue my own research ideas.

Next, I would like to thank Volker Kühn and Gerald Matz for carefully reviewing the thesis and acting as the examiners at the defence.

I would like to thank Peter Jung and Giuseppe Caire for giving me the chance to collaborate during my several research visits at the Communications and Information Theory Group (CommIT), TU Berlin. I am also grateful to all members of CommIT, especially Jana Hantke, for making me feel welcome to the group.

My appreciation also goes to my professors and colleagues at the Institute of Telecommunications, TU Vienna. In particular, I would like to thank Martin Müller and Gabor Hannak, without whom it would have been impossible to finish this work. I also owe a big thank to Michael, Geetha, Wolfgang, Blanca and Ljiljana and all other colleagues and friends for supporting me over the years, reading my thesis, and being my right hand in Vienna while I was away.

For their unconditional love and support, I would like to express my deepest gratitude to my late father, my mother and my brother.

Table of Contents

Notation and Definitions

number of measurements	m
dimension of the source vector	N
sparsity of a vector	k
deterministic scalar, column vector, and matrix	$a, \mathbf{a}, \mathbf{A}$
random scalar, column vector, and matrix	$\mathsf{a}, \boldsymbol{\mathsf{a}}, \boldsymbol{\mathsf{A}}$
nth component of vector	a_n
(m, n)th entry of a matrix	$(\mathbf{A})_{m,N} = A_{m,N}$
vector and matrix transpose	$\mathbf{a}^T, \mathbf{A}^T$
matrix inverse	$\mathbf{A}^{-1}$
$N \times N$ identity matrix	$\mathbf{I}_N$
vector having all but i-th element	$\mathbf{a}_{\sim i}$
vector p-norm $(p \geq 1)$	$\|\mathbf{a}\|_p$
average value of the entries of a vector	$\langle \cdot \rangle$
natural numbers	$\mathbb{N}$
integer numbers	$\mathbb{Z}$
real numbers	$\mathbb{R}$
vector space of real valued N dimensional vectors	$\mathbb{R}^N$
sign of a number	$\mathrm{sgn}(\cdot)$
cardinality of a set	$\lvert\mathcal{S}\rvert$
set of positive integers up to N	$[N] = \{1, \ldots, N\}$
vector with components indexed by set	$\mathbf{a}_{\mathcal{S}}$
matrix with columns indexed by set	$\mathbf{A}_{\mathcal{S}}$
probability of an event	$\mathrm{P}\{\cdot\}$
expectation of a random quantity	$\mathbb{E}\{\cdot\}$
variance of a random quantity	$\mathrm{var}\{\cdot\}$
(multivariate) normal distribution with mean $\boldsymbol{\mu}$ and (co-)variance $\boldsymbol{\Sigma}$	$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$

iteration index (e.g., in an algorithm)	$(\cdot)^{(t)}$
Dirac delta function	$\delta(\cdot)$

Functionals

- Dirac delta (generalized) function: in the strict sense the Dirac delta is not a function, but defined by the integral

$$f(a) = \int_{\mathcal{R}(f)} f(x)\delta(x-a)dx$$

over any function $f : \mathcal{R}(f) \to \mathcal{I}(f)$ with $\mathcal{R}(f), \mathcal{I}(f) \subseteq \mathbb{R}$.

Miscellaneous

- *Mean squared error* (MSE): the MSE between two vectors of dimension N is defined as

$$\mathrm{MSE}(\mathbf{a}, \mathbf{b}) = \frac{1}{N}\|\mathbf{a}-\mathbf{b}\|_2^2 .$$

- Decibel notation: quantities $x \in \mathbb{R}$ in dB units are defined as

$$x\,\mathrm{dB} = 10^{\frac{x}{10}} .$$

Chapter 1

Introduction

1.1 Motivation

The recent advances in integrated semiconductor technology [73] has led to ubiquitous deployment of cheap digital devices equipped with sensors monitoring a multitude of natural phenomena (e.g., temperature, air pollution) [16]. In the context of Internet of things (IoT) many digital devices (e.g., sensors) communicate and interact with each other over the Internet while being monitored and controlled from a remote device [60]. These digital devices sense, save, process and transmit exponentially increasing volumes of sensor data, which is threatening to overwhelm classical sensing systems [3]. Therefore, the digital sensors of the future need to be supported with fast and robust yet flexible solutions for sensing, saving, processing, and transmission. In the signal processing world, the answers are to be found in the algorithms for sampling, quantization, compression, dequantization and recovery of the input signals. The task of designing these algorithms is tackled in this thesis.

To illustrate the issues mentioned above, consider the acquisition of a grey-scale image. In the classical approach, a binary number (i.e., pixel value) representing light intensity is taken from each light detector in a grid of hundreds-by-hundreds detectors. Storing this large matrix of binary numbers would require an extensive storage volume. The matrix, therefore, needs to be compressed before storage. For example, we can compress the matrix using JPEG compression. This compression is based on the fact that signals obtained from *2 dimensional discrete cosine transformation* (DCT-II) transformation of natural images show a specific structure; in particular, only a small portion of the transform coefficients have large absolute value, while the rest of the coefficients are close to zero. Therefore, only a few coefficients have a significant contribution to the signal. Saving the positions and the values of only

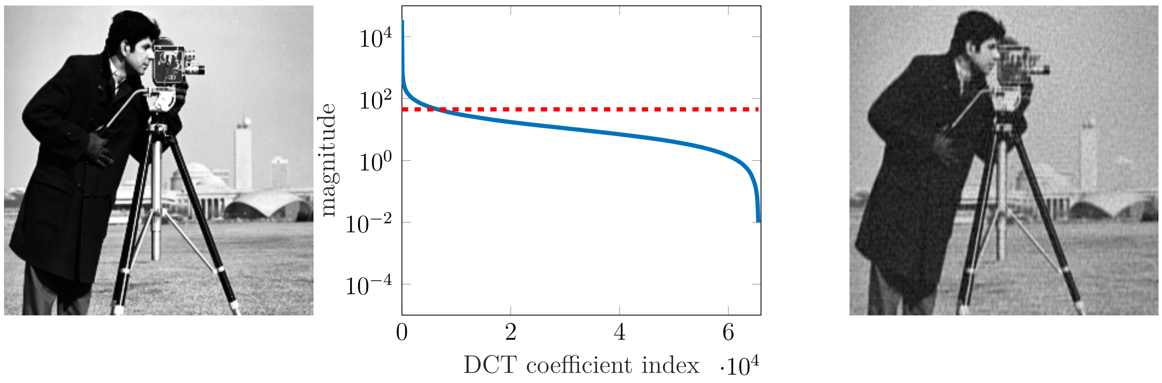


Fig. 1.1 An example of a classical image compression: the original image, distribution of the magnitudes, and the reconstruction from compressed representation (only 10 % of the coefficients are used for reproduction).

significant coefficients requires a fraction of storage space needed for the raw data. When recovering the image from the compressed representation, missing pixel values are replaced with zeros, and the image is obtained by performing the inverse DCT-II transform. An example of image compression with JPEG is shown in Figure 1.1 (taken from [43]). Even though there is a noticeable reduction in the image quality, the key elements, and most details of the image are contained in the compressed image.

Although a high compression ratio can be achieved using a JPEG compression, and even more so with more sophisticated compression algorithms (such as JPEG2000), one could argue that the whole approach is suboptimal. The reasoning is that the compression step is compensating for the redundant sampling step [43]. An alternative is offered by a modern *digital signal processing* (DSP) paradigm called *compressed sensing* (CS) [18, 19, 30], where previously separate tasks of sensing and compression are done in a single step. This is accomplished by taking randomized linear measurements of a low-dimensional signal that is embedded in a high-dimensional data space. Since the number of measurements is lower by orders of magnitude than the dimension of the ambient space, the compression is already achieved. On the other hand, the problem is ill-posed, and classical recovery algorithms break down. Nonetheless, a stable recovery is achieved using sophisticated iterative CS recovery algorithms that exploit the signal structure (e.g., sparse DCT-II transform) during the reconstruction. For example, the *single-pixel camera* [38] exploits these ideas to obtain an image in a poorly lit environment with the compression ration of 20, while using only a single light detector. Even though this has been a successful proof of concept imaging system, there are still many practical challenges in image acquisition with CS, that are often neglected during the development of the theoretical background. For example, the light detectors might not be well calibrated; the light intensities have to be quantized before any later processing; the image needs to be recovered from a finite-precision representation. Poor solutions to these problems typically degrade the user experience and are thus unacceptable. More detailed investigation of the practical aspects of CS, therefore, needs to be done.

In this thesis, I thus focus on two most prominent challenges of the practical application of CS in digital sensors of the future, namely quantization and calibration. More specifically, concerning quantization, I investigate the quantization of CS measurements, and recovery of sparse signals from quantized CS measurements. Furthermore, for the case of miscalibrated sensors, I investigate a modern sampling paradigm, called *unlimited sampling* (US), within the CS framework.

1.2 Contribution and Outline

Chapter 2: Preliminaries

This chapter aims to make the reader familiar with some general concepts that are relevant when discussing the main contributions of the thesis, namely, CS algorithms for quantization and sampling applications. I start by defining noiseless and noisy CS problem, as well as defining terms frequently used when discussing those problems. The list of the terms can be, therefore, used as a reference for subsequent chapters. Later, I discuss some of the most important results in CS, namely, *restricted isometry property* (RIP) and connection to *basis pursuit denoising* (BPDN).

In the subsequent chapters I will be referencing to algorithms that are either used in their original form or modified for the nonlinear CS problems I focus on. Hence, I provide a short overview and discussion on the classical CS algorithms. Finally, I present the problem of *quantized compressed sensing* (QCS) and discuss the most relevant results from literature.

Chapter 3: Approximate message passing

Regardless of the practical problem at consideration, i.e., quantization or miscalibration, the core task is, nonetheless, the recovery of a sparse signal from a noisy measurement. Since both problems introduce nonlinear effects, I am interested in the recovery of sparse signals from measurements corrupted with nonlinear distortions. Among many CS recovery algorithms operating with linear measurements in additive noise, the class of *approximate message passing* (AMP) algorithms stands out as the one with most potential for solving CS problems with nonlinear measurements. Even in this case, the AMP algorithm approximates the computationally intractable high-dimensional integration involved with calculating $\mathbb{E}\{\mathbf{x}|\mathbf{y}\}$ or $\arg\max_{\mathbf{x}} p(\mathbf{x}|\mathbf{y})$ with a highly efficient iterative procedure [62]. In Chapter 3, I, therefore, present an overview of the AMP algorithm, which assumes *independent and identically distributed* (i.i.d.) Laplacian prior for the entries of the unknown vector.

As there is nothing special about the Laplacian prior for AMP to be derived, one can also consider other prior distributions. If the source prior is known (or if it can be estimated) I show how to use the *Bayesian approximate message passing* (BAMP) algorithm to get even better estimates compared to the classical AMP algorithm. Furthermore, as my own contribution, I assume that the prior consists of a weighted average of n-Gaussian distributions, each with potentially different mean and different variance, and derive closed-form expressions for the denoiser functions of the BAMP algorithm. By choosing appropriate values for the means and the variances, one can model many practically interesting priors. Moreover, by picking a very small but still non-zero variance, one can even approximate discrete *probability mass function*s (pmfs).

Similar to the reasoning which led to the BAMP algorithm, one can argue that there is nothing special about the Gaussian noise model, where the linear mixtures $\mathbf{z} = \mathbf{A}\mathbf{x}$ are corrupted with i.i.d. Gaussian noise. Hence, I conclude Chapter 3 by briefly presenting the *generalized approximate message passing* (GAMP) algorithm, where one considers a general distribution describing the component-wise distortion of the linear mixtures, i.e., a general output channel given in terms of a conditional distribution $p(y_i|z_i)$.

Chapter 4: Analysis-by-Synthesis with Bayesian Approximate Message Passing Scalar Quantization for Compressed Sensing

In Chapter 4, I consider a scenario where sensors using CS observe a sparse signal, quantize the observations, and transmit the discrete valued data over a communication link with a low rate constraint. I start by stating the problem of quantization of CS measurements and providing a more intuitive derivation of the optimal quantizer, than offered in literature.

Major parts of this chapter were published in [64].

Chapter 5: Approximate Message Passing for Quantized and Noisy Compressed Sensing

In many practical applications, CS measurements are first scalar quantized and subsequently corrupted in different ways during storage or transmission over a noisy channel. Reconstruction by conventional CS algorithms on such highly distorted measurements results in poor accuracy. To address this problem, I use the well established GAMP algorithm and adapt it to our specific problem: recovery of sparse vectors from quantized CS measurements corrupted with noise. I consider different communication channels tampering with the quantized measurements, namely the *symmetric discrete memoryless* (SDM) channel and the *additive white Gaussian noise* (AWGN) channel. I provide analytical expressions for the necessary nonlinear updates of the GAMP algorithm for different channel models and different rates. I conduct numerical experiments and present performance results of the proposed scheme.

The contributions of this chapter were published in [66] and [65].

Chapter 6: GAMP for Unlimited Sampling of Compressed Sensing Measurements

In Chapter 6, I tackle the problem of sampling signals with miscalibrated sensors within the CS framework. More specifically, I consider the GAMP algorithm for recovering a sparse signal from modulo samples of its randomized projections. The modulo samples are obtained by a *self-reset analog to digital converter* (SR-ADC). In contrast to previous work on SR-ADC that considers sparse vectors either in time or frequency domain, I allow for sparse signals in any basis. Furthermore, I also consider a scenario where the randomized projections are sent through a communication channel before being digitizing by an SR-ADC. There, the channel is modeled as an AWGN channel. To show the effectiveness of the proposed approach, I conduct *Monte-Carlo* (MC) simulations for both noiseless and noisy case. The results show the ability of the proposed algorithm to fight the nonlinearity of the SR-ADC, as well as the possible additional distortion introduced by the AWGN channel.

Major parts of this chapter were published in [67].

Chapter 7: Conclusions

Conclusions of this thesis, as well as suggestion for future work are summarized in Chapter 7.

Chapter 2

Preliminaries

This chapter serves as a background for presenting the main contributions of the thesis. It introduces the reader to the mathematical foundations of *compressed sensing* (CS) in Section 2.2, and provides an overview of the classical CS recovery algorithms in Section 2.3. Furthermore, the problem of *quantized compressed sensing* (QCS), together with a discussion on the most relevant results from literature, are presented in Section 2.4.

2.1 Intoduction

Shannon's sampling theorem is a fundamental result in signal processing. It states that a continuous bandlimited signal can be perfectly reconstructed from a set of samples taken at a sampling rate proportional to the maximum frequency present in the signal (Nyquist rate) [75]. For many applications, the required number of samples is, however, disproportional to the information content of the signal, indicating redundancy in the acquisition [3, 37]. The ambition to develop a more efficient sampling framework led, more than a decade ago, to the emergence of CS [20, 30, 31].

2.2 Compressed Sensing

In CS we take randomized linear measurements of a low-dimensional signal that is embedded in a high-dimensional data space. More formally, we obtain m linear measurements $\{y_a\}_{a=1}^{m}$ of an N-dimensional vector $\mathbf{s}$ by multiplying it with a fat matrix Φ, i.e.,

$$\mathbf{y} = \Phi\mathbf{s}, \tag{2.1}$$

where $m < N$. In general, this is an underdetermined system of linear equations and the equality holds for infinitely many $\mathbf{s}$. However, in CS we take measurements of signals of a certain structure, i.e., signals that are generated by a model, and consequently, we restrict the set of possible solutions for $\mathbf{s}$ in (2.1) to those described by the model. In particular, we are interested in the case where $\mathbf{s}$ has a k-sparse[1] representation in some basis Ψ, i.e.,

$$\mathbf{s} = \Psi\mathbf{x}, \tag{2.2}$$

where $\mathbf{x} \in \sum_k$, and $\sum_k$ is the set of all k-sparse vectors in $\mathbb{R}^N$. Hence, we are interested in finding $\mathbf{x}$ which satisfies

$$\mathbf{y} = \mathbf{A}\mathbf{x}, \tag{2.3}$$

where $\mathbf{A} = \Phi\Psi$ is a so-called measurement matrix, and $\|\mathbf{x}\|_0 = k$. Before continuing, it is important to clarify the notation that I use throughout the thesis which is related to the problem above, namely

- N is the dimension of the unknown signal,
- m is the number of measurements,
- k is the sparsity of a signal,
- $\delta = m/N$ is the measurement ration,
- $\rho = k/m$ is the normalized measure of sparsity,
- $\epsilon = k/N$ is the fraction of nonzeros.

It can be show that for any $\mathbf{y} \in \mathbb{R}^m$, there exists at most one $\mathbf{x} \in \sum_k$ that satisfies the measurement constraint iff $\text{spark}(A) > 2k$, where $\text{spark}(A)$ is the smallest number of linearly independent columns of $\mathbf{A}$ [27]. Even though we know that for a particular problem, the solution is unique if it exists, the problem of finding that solution is still an NP-hard problem [68]. However, in [39, p. 48] it has been shown that $\mathbf{x}$ is a unique k-sparse solution of (6.6) iff it is the solution of

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \|\mathbf{x}\|_0, \qquad \text{s.t.} \qquad \mathbf{y} = \mathbf{A}\mathbf{x}, \tag{2.4}$$

which is till an NP hard problem. One way to find an approximate solution of the problem (2.4), is to replace $\|\cdot\|_0$ with $\|\cdot\|_1$, i.e.,

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \|\mathbf{x}\|_1, \qquad \text{s.t.} \qquad \mathbf{y} = \mathbf{A}\mathbf{x}, \tag{2.5}$$

[1] We say that a signal (i.e., vector) is k-sparse if it has exactly k non-zero entries, i.e., $\|\mathbf{x}\|_0 = k$.

with the intuition that, when searching for a solution in $\mathbb{R}^2$, the solution of an ℓ_1 minimization problem equals the solution of an ℓ_p minimization problem, for any $p < 1$ [27, p. 28]. This approach, called *basis pursuit* (BP), was proven to be surprisingly successful in many different applications, e.g., recovering bandlimited signals, and geological signals consisting of a train of spikes [27, 37]. Fortunately, the analysis of recovery properties of ℓ_1 minimization was shown to be possible using the *restricted isometry property* (RIP) [21].

2.2.1 Restricted isometry property

Definition 1 [21] *A matrix* $\mathbf{A} \in \mathbb{R}^{m \times N}$ *satisfies the RIP of order* k *with RIP constant* $\delta_K \in (0,1)$ *if for all* k*-sparse vectors* $\mathbf{x} \in \mathbb{R}^N$

$$(1-\delta_K)\|\mathbf{x}\|_2^2 \leq \|\mathbf{Ax}\|_2^2 \leq (1+\delta_K)\|\mathbf{x}\|_2^2 \,.$$

Matrices that satisfy the RIP of order k are approximate isometries for $\sum_k$, meaning that they approximately preserve the ℓ_2 norm of k-sparse signals under the transformation. Alternatively we can say that, if the measurement matrix satisfies the RIP of order $2k$, then it approximately preserves the distance between any two k-sparse vectors. This indicates that the recovery of sparse signals is possible from (6.6), if $\mathbf{A}$ satisfies RIP of order $2k$ with certain constant δ_{2k}. It is still unclear, however, how to construct a measurement matrix.

One approach is to use a deterministic procedure to construct a measurement matrix that satisfies the RIP of order k [15, 29, 44]. Even though this construction guarantees matrices with the desired properties, the matrix itself has a relatively large (at least $k^2 \log N$) number of rows. This means that the matrix requires too many measurements of the sparse signal for practical applications [39].

The second approach is to use random constructions of the measurement matrix. For example, one could construct a measurement matrix by independent sampling from a sub-Gaussian distribution and assign the samples to the entries of the matrix. In this setting, testing if the matrix satisfies RIP becomes an exponentially hard problem even for moderate size problems (i.e., m and N are not too small). Therefore, we can only aim to proove that the matrix satisfies RIP with certain probability, and that is precisely done by Theorem 1.

Definition 2 [39, p. 191] *A random variable* x *is called sub-Gaussian if there exist constants* $\beta, \kappa > 0$ *such that* $\forall t > 0$

$$\mathrm{P}\{|\mathsf{x}| > t\} \leq \beta e^{-\kappa t^2}.$$

Definition 3 [39, p. 193] *If* x *is a zero-mean sub-Gaussian random variable, then there exists a constant* c *(depending only on* β *and* κ*) such that*

$$\mathbb{E}\{\exp(\theta \mathsf{x})\} \leq \exp(c\,\theta^2) \qquad \textit{for all } \theta \in \mathbb{R}. \tag{2.6}$$

Any valid constant c *in (2.6) is called a sub-Gaussian parameter of* x*.*

Definition 4 [39, p. 309] *A matrix* $\mathbf{A}$ *is called sub-Gaussian if all the entries of the matrix are independent zero-mean sub-Gaussian random variables with variance 1 and the same sub-Gaussian parameter* c*. Alternatively,* $\mathbf{A}$ *is called sub-Gaussian if the entries are independent sub-Gaussian random variables with variance 1 and the same parameters* β *and* κ*.*

Theorem 1 [39, Theorem 9.2] *Let* $\mathbf{A}$ *be an* $m \times N$ *sub-Gaussian random matrix with normalized columns. Then there exists a constant* $C > 0$ *(independent of* m, N, δ_k*) such that the RIP constant of* $\mathbf{A}$ *satisfies* $\delta_k \leq \delta$ *with probability at least* $1 - \epsilon$ *provided* $m \geq 2C\delta^{-2}(k \ln(eN/k) - \ln(2\epsilon^{-1}))$*.*

Suppose now that we pick some small δ and take $m \geq 2C\delta^{-2}k \ln(eN/k)$ measurements. If we set $\epsilon = 2\exp(-\delta^2 m/(2C))$, according to Theorem 1 the RIP constant of the matrix satisfies $\delta_k \leq \delta$ with probability at least $1 - \epsilon$. This is a quite powerful result on the required number of rows of $\mathbf{A}$ for it to satisfy the RIP with certain δ and ϵ. Namely, if the matrix is constructed as described above, M scales linearly with k and logarithmically with N, which is by far superior than anything achieved by deterministic constructions.

2.2.2 RIP and ℓ_1 recovery

The significance of the RIP becomes apparent from the remarkable results of Candès, Tao, Romberg and others [18, 19, 30]. To get more insight into that work, let $\mathbf{x}_k$ denote the best sparse approximation one could obtain if one knew exactly the locations and amplitudes of the k-largest entries of $\mathbf{x}$, i.e., the vector $\mathbf{x}$ with all but the k-largest entries set to zero. Additionally, we assume that $\mathbf{A}$ satisfies the RIP of order $2k$ with a certain RIP constant δ_{2k}.

Theorem 2 [19, Theroem 1.1 (Noiseless recovery)] *Assume that* $\delta_{2k} < \sqrt{2} - 1$*. Then the solution* $\hat{\mathbf{x}}$ *to (2.5) obeys*

$$\|\hat{\mathbf{x}} - \mathbf{x}\|_1 \leq C_0 \|\mathbf{x} - \mathbf{x}_k\|_1, \tag{2.7}$$

and

$$\|\hat{\mathbf{x}} - \mathbf{x}\|_2 \leq C_0\, k^{-1/2} \|\mathbf{x} - \mathbf{x}_k\|_1, \tag{2.8}$$

for some constant[2] C_0..

It follows from Theorem 2 that both the ℓ_1 and the ℓ_2 norm of the reconstruction error are upper bounded by a term that depends on how well $\mathbf{x}$ can be approximated by a strictly k-sparse vector. Alternatively, if we assume that $\mathbf{x} \in \sum_k$, then the solution of (2.5) perfectly matches the unknown vector $\mathbf{x}$.

The problem of recovering a sparse vector becomes even more difficult when the linear mixtures $\mathbf{z} = \mathbf{A}\mathbf{x}$ are corrupted with noise, i.e.,

$$\mathbf{y} = \mathbf{z} + \mathbf{w} = \mathbf{A}\mathbf{x} + \mathbf{w}, \tag{2.9}$$

where $\mathbf{w}$ is the noise vector. Therefore, the linear mixtures $\mathbf{z}$ are often referred to as noiseless CS measurements. In this case we want to find

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \|\mathbf{x}\|_1, \quad \text{s.t.} \quad \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2 \leq \varepsilon, \tag{2.10}$$

where ε is an upper bound on the size of the noise contribution. This problem is also known as *basis pursuit denoising* (BPDN). Again, in the work of Candès [19], we find theoretical guarantees for recovering a sparse vector from noisy measurements.

Theorem 3 [19, Theroem 1.2 (Noisy recovery)] *Assume that* $\delta_{2k} < \sqrt{2} - 1$ *and* $\|\mathbf{w}\|_2 \leq \varepsilon$*. Then the solution* $\hat{\mathbf{x}}$ *to (2.10) obeys*

$$\|\hat{\mathbf{x}} - \mathbf{x}\|_2 \leq C_0\, k^{-1/2} \|\mathbf{x} - \mathbf{x}_k\|_1 + C_1\, \varepsilon, \tag{2.11}$$

with the same constant C_0 *as before and some constant*[3][4] C_1*.*

Similar to Theorem 2 for the noiseless case, Theorem 3 provides an upper bound for the ℓ_2 norm of the reconstruction error. The bound consists of two terms: one that

[2]The exact value for C_0 can be found in [19].
[3]The exact value for C_1 can be found in [19].
[4]This theorem can be specialised for *independent and identically distributed (i.i.d.)* Gaussian noise, or when $\|\mathbf{A}^T\mathbf{w}\|_\infty$ is small [19].

depends on *how sparse* $\mathbf{x}$ is, which vanishes if $\mathbf{x} \in \sum_k$, and one which scales linearly with ε. As a conclusion, the solution of (2.10), i.e., recovering a sparse vector from a noisy measurement vector using the ℓ_1 norm instead of the ℓ_0 norm is stable as it depends linearly on the size of the noise contribution. The remaining question is *"How to solve for the given problem?"*.

2.3 Compressed Sensing Recovery Algorithms

Since the early work of Candès on the equivalence of solutions of (2.4) and (2.5) for strictly sparse signals, many authors proposed different iterative recovery algorithms for solving (2.10). The list of available algorithms grows larger every year, and a comprehensive overview would go beyond the scope of this work. I think, however, that it is necessary to give a quick introduction for the algorithms that will be used later in the thesis. To do so, I make a slight modification to the concise classification of iterative CS recovery algorithms provided in [39]. Specifically, the algorithms are classified as:

- ℓ_1 minimization algorithms
- greedy algorithms
- thresholding-based algorithms
- *approximate message passing* (AMP) algorithms

I what follows, I give a short overview of the algorithm classes from above.

2.3.1 ℓ_1 minimization algorithms

The BPDN problem defined in (2.10), namely

$$\hat{\mathbf{x}}_{\text{BPDN}} = \arg\min_{\mathbf{x}} \|\mathbf{x}\|_1, \quad \text{s.t.} \quad \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2 \leq \varepsilon, \tag{2.12}$$

is already cast as a convex optimization problem. This means that one could apply numerical solvers known in convex optimization theory to solve both problems. Using this approach, we can solve two additional convex problems: *least absolute shrinkage and selection operator* (LASSO) and *constrained least squares* (CLS), defined in (2.13) and (2.14), respectively.

BPDN is a convex problem where we want to minimize a linear combination of the ℓ_1 norm of the solution and the term representing disagreement with the measurements, where the trade-off between the two is controlled by a parameter λ. Specifically, for some parameter $\lambda \geq 0$,

$$\hat{\mathbf{x}}_{\text{LASSO}} = \arg\min_{\mathbf{x}} \quad \lambda\|\mathbf{x}\|_1 + \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2. \tag{2.13}$$

In CLS, compared to (2.10), one switches the roles of the constraint and term to be minimized. Namely, for some τ,

$$\hat{\mathbf{x}}_{\text{CLS}} = \arg\min_{\mathbf{x}} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2, \quad \text{s.t.} \quad \|\mathbf{x}\|_1 \leq \tau. \tag{2.14}$$

In general, for some choice of ε, λ, and τ we obtain three different solutions for $\mathbf{x}$ in (2.10), (2.13), and (2.14), respectively. However, in [39, p. 64], it was shown that for any choice of $\varepsilon > 0$, the solution of (2.10) is identical to the solution of (2.13) for a specific $\lambda > 0$, and identical to the solution of (2.14) for a specific $\tau > 0$. The same conclusion is reached if we fix $\lambda > 0$ or $\tau > 0$ to any arbitrary value larger then zero. Choosing the appropriate algorithm mainly depends on prior information, e.g., if we have some estimate of the sparsity or noise level. For example, if we know or can estimate $\|\mathbf{x}\|_2$ or $\|\mathbf{x}\|_\infty$, we can upper bound $\|\mathbf{x}\|_1$ using

$$\|\mathbf{x}\|_1 \leq \sqrt{N}\|\mathbf{x}\|_2\,, \quad \text{or} \quad \|\mathbf{x}\|_1 \leq N\|\mathbf{x}\|_\infty\,. \tag{2.15}$$

Note that in literature the problems in (2.10), (2.13) and (2.14) interchange names (e.g., in [39, p. 64] one finds that the LASSO problem defined in (2.13) is referred to as the BPDN problem).

2.3.2 Greedy algorithms

With greedy algorithms, at each iteration n, we first calculate the residual estimation error, in short called *residual*, $\mathbf{r}^{(n)} = \mathbf{y} - \mathbf{A}\hat{\mathbf{x}}^{(n-1)}$. Next, one aims to find indices of columns of the measurement matrix with the largest contribution in minimizing the residual. For example, in *orthogonal matching pursuit* (OMP) whose steps are given in Algorithm 1, in each iteration we find exactly one index, whose corresponding column in $\mathbf{A}$ has the largest correlation with the residual. Once it is found, this index is added to the support $\mathcal{S}^{(t)}$ of $\hat{\mathbf{x}}^{(t)}$, followed by solving a *least squares* (LS) problem involving $\mathbf{A}_{\mathcal{S}^{(t)}}$. The classical solution of the LS problem involves computing the inverse $(\mathbf{A}_{\mathcal{S}^{(t)}}^T\mathbf{A}_{\mathcal{S}^{(t)}})^{-1}$, which has a complexity of $\mathcal{O}(|\mathcal{S}|^3)$. However, there exist faster iterative

Algorithm 1 Orthogonal matching pursuit (OMP)

```
Input: measurement matrix $\mathbf{A}$, measurement vector $\mathbf{y}$
Initialization: $t = 0, \ S^0 = \emptyset, \ \hat{\mathbf{x}}^{(0)} = \mathbf{0}_{N\times 1}$
 do:
  1: $t = t + 1$                                                            ▷ increment iteration counter
  2: $j^{(t)} = \arg\max_{j\in[N]} \ \{|\mathbf{A}^*(\mathbf{y} - \mathbf{A}\hat{\mathbf{x}}^{(t-1)})_j|\}$          ▷ index of the largest contribution
  3: $S^{(t)} = S^{(t-1)} \bigcup \{ j^{(t)} \}$                                  ▷ update support of the estimate
  4: $\hat{\mathbf{x}}^{(t)} = \arg\min_{\mathbf{z}\in\mathbb{R}^N}\{\|\mathbf{y} - \mathbf{A}\mathbf{z}\|_2, \ \mathrm{supp}(\mathbf{z}) \subset S^{(t)}\}$      ▷ estimate of sparse vector
 while $t \le k$
Output: $\hat{\mathbf{x}} = \hat{\mathbf{x}}^{(t)}$
```

methods for the projection step based on the QR-decomposition of $\mathbf{A}_{\mathcal{S}^{(t)}}$ that make a use of the decomposition at the previous step $t-1$, i.e., decomposition of $\mathbf{A}_{\mathcal{S}^{(t-1)}}$ [39].

In the noiseless setting, the OMP algorithm will provide an exact reconstruction of any $k-$sparse signal after k iterations, provided that $\mathbf{A}$ satisfies the RIP of order $k+1$ with isometry constant $\delta < \frac{1}{3\sqrt{k}}$ [28]. This result was further improved to $\delta < \frac{1}{\sqrt{2k}}$ in [53], and more recently to $\delta < \frac{1}{1+\sqrt{k}}$ in [56, 61]. For a matrix constructed using a sub-Gaussian distribution this corresponds to taking $m = \mathcal{O}(k\log(N/k)/\delta^2)$ measurements. In [27] the authors argue that, since the required constants are relatively small, we instead need to obtain $\mathcal{O}(k^2\log(N))$ measurements.

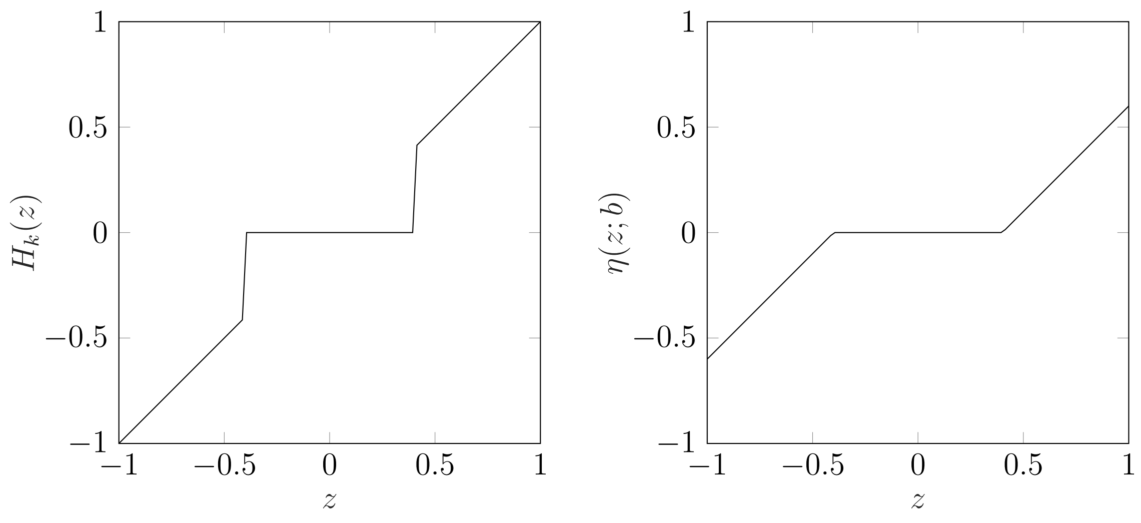
In the noisy case however, with OMP there is a certain probability that we will erroneously update the support set. Moreover, if this happens, the error will propagate across the subsequent iterations, and as a consequence this will introduce even more noise in those iterations. For this reason, in *compressive sampling matching pursuit* (CoSaMP) [69], the support is estimated in a more flexible way. Here, the support set of size k from the previous iteration is augmented with $L_{2k}(\mathbf{r}^{(t)})$, i.e., the set of $2k$ indices that correspond to the columns of $\mathbf{A}$ that have the largest correlation with the residual. As in OMP, this step is followed by a projection step. Since the cardinality of the candidate support set is anything between k and $3k$, one needs to apply the hard thresholding operator $H_k(\mathbf{z})$, which is the best k-term approximation of the input $\mathbf{z}$. The most important properties of the algorithm are summarized in Theorem 4.

Theorem 4 [69, Theroem A (CoSaMP)] *Suppose that $\mathbf{A}$ is an $m\times N$ sampling matrix with restricted isometry constant $\delta_{2k} \le c$. Let $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{w}$ be a vector of samples of an arbitrary signal, contaminated with arbitrary noise. For a given precision parameter η, the algorithm CoSaMP produces a k-sparse approximation $\hat{\mathbf{x}}$ that satisfies*

$$\|\mathbf{x} - \hat{\mathbf{x}}\|_2 \le C\max\left\{\eta, \frac{1}{\sqrt{k}}\|\mathbf{x} - \mathbf{x}_{k/2}\|_1 + \|\mathbf{e}\|_2\right\}, \tag{2.16}$$

Algorithm 2 Compressive sampling matching pursuit (CoSaMP)

```
Input: measurement matrix $\mathbf{A}$, measurement vector $\mathbf{y}$, sparsity level $k$, stopping
    threshold $t_{\max}$
Initialization: $t = 0$, $\hat{\mathbf{x}}^{(0)} = \mathbf{0}_{N\times 1}$
do:
  1: $t = t + 1$                                                                          ▷ increment iteration counter
  2: $U^{(t)} = \mathrm{supp}(\hat{\mathbf{x}}^{(t-1)}) \cup L_{2k}(\mathbf{A}^*(\mathbf{y} - \mathbf{A}\hat{\mathbf{x}}^{(t-1)}))$          ▷ update potential support
  3: $\mathbf{u}^{(t)} = \arg\min_{\mathbf{z}\in\mathbb{R}^N}\{\|\mathbf{y} - \mathbf{A}\mathbf{z}\|_2,\ \mathrm{supp}(\mathbf{z}) \subset U^{(t)}\}$     ▷ update candidate vector
  4: $\hat{\mathbf{x}}^{(t)} = H_k(\mathbf{u}^{(t)})$                                                    ▷ apply hard thresholding
while $t \leq t_{\max}$
Output: $\hat{\mathbf{x}} = \hat{\mathbf{x}}^{(t)}$
```



(a) Hard thresholding function. (b) Soft thresholding function.

Fig. 2.1 Hard thresholding function and soft thresholding function with thresholding parameters set to 0.4 in both cases.

where $\mathbf{x}_{k/2}$ *is a best* $k/2$*-sparse approximation of* $\mathbf{x}$[5]*. The running time is* $\mathcal{O}(\mathcal{L}\log(\|\mathbf{x}\|_2/\eta))$*, where* $\mathcal{L}$ *bounds the cost of a matrix-vector multiplication with* $\mathbf{A}$ *or* $\mathbf{A}^*$*. Working storage is* $\mathcal{O}(N)$*.*

It is worth noting that CoSaMP usually iterates for more steps than OMP, which might be a critical disadvantage for certain applications. The steps of the algorithm are given in Algorithm 2.

2.3.3 Thresholding algorithms

Two prominent algorithms based on thresholding operators are *iterative hard thresholding* (IHT) [12] and *iterative soft thresholding* (IST) [26, 58], which solve

$$\hat{\mathbf{x}}_{\text{IHT}} = \arg\min_{\mathbf{x}} \ \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \mu\|\mathbf{x}\|_0, \tag{2.17}$$

and

$$\hat{\mathbf{x}}_{\text{IST}} = \arg\min_{\mathbf{x}} \ \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \lambda\|\mathbf{x}\|_1, \tag{2.18}$$

respectively. Both algorithms iterate between moving in the direction of the gradient of the measurement fidelity term, i.e., gradient of $\|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2$, and applying a thresholding function $\mathcal{M}(\mathbf{z})$ that promotes the signal model. The thresholding function is applied component-wise and the specific functions for IHT and IST are shown in Figure 2.1. In the case of IHT, $\mathcal{M}(\mathbf{z})$ is the hard thresholding function $H_k(\mathbf{z})$ (Figure **??**). This function simply selects k largest terms of the current estimate. On the other hand, the soft thresholding function $\eta(x;b)$

$$\eta(x;b) \begin{cases} x - b, & \text{if } b \le x, \\ 0, & \text{if } -b \le x \le b, \\ x + b, & \text{if } x \le -b. \end{cases} \tag{2.19}$$

has a slightly different shape due to the minimization of the ℓ_1-term (Figure **??**). The question of how to select parameters of iterative thresholding algorithms (b, μ, λ), as well as the parameters of CoSaMP and subspace pursuit such that a given algorithm successfully reconstruct signals with as low as possible sparsity (i.e., largest number of nonzeros) was answered in [58].

Algorithm 3 Iterative thresholding algorithms

```
Input: measurement matrix $\mathbf{A}$, measurement vector $\mathbf{y}$, thresholding parameter $k$ or
    $\tau$, stopping threshold $t_{\max}$ or $\epsilon$
Initialization: $t = 0$,  $\hat{\mathbf{x}}^{(0)} = \mathbf{0}_{N\times 1}$
 do:
  1: $t = t + 1$                                                    ▷ increment iteration counter
  2: $\mathbf{z}^{(t)} = \hat{\mathbf{x}}^{(t-1)} + \mathbf{A}^*(\mathbf{y} - \mathbf{A}\hat{\mathbf{x}}^{(t-1)})$      ▷ move in the direction of the gradient
  3: $\hat{\mathbf{x}}^{(t)} = \mathcal{M}(\mathbf{z}^{(t)})$                                           ▷ thresholding
 while $t \le t_{\max}$ or $\|\hat{\mathbf{x}}^{(t)} - \hat{\mathbf{x}}^{(t-1)}\|_2 \ge \epsilon$
Output: $\hat{\mathbf{x}} = \hat{\mathbf{x}}^{(t)}$
```

[5] c and C are some universal positive constants. For details check [69].

2.3.4 Approximate message passing algorithms

The term AMP refers to a class of algorithms that are based on Gaussian and quadratic approximations of the loopy BP on dense graphs. In the corresponding graphical model, the entries of the unknown vector are represented by so-called variable nodes, the entries of the measurement vector are represented by variable nodes, and edges represent statistical dependencies between nodes. Since, in the general case, the measurement matrix is dense, and the entries of the unknown vector are i.i.d., the corresponding graph is an instance of a dense bipartite graph. Whether we are interested in an approximate *minimum mean squared error* (MMSE) estimate or a *maximum a posteriori* (MAP) estimate, we apply sum-product or max-sum belief propagation on a graph with loops. As a result, the AMP algorithm approximates the computationally intractable high-dimensional integration involved with calculating $\mathbb{E}\{\mathbf{x}|\mathbf{y}\}$ or $\arg\max_{\mathbf{x}} p(\mathbf{x}|\mathbf{y})$with a highly efficient iterative procedure [62]. Given that the AMP algorithms [33–35, 57, 71, 72] lie at the heart of this thesis, we will take a deeper look at these algorithms in Chapter 3.

2.4 Quantized Measurement Vectors

2.4.1 Quantization

In this thesis, I focus on scalar quantizers. A scalar quantizer $Q(\cdot)$ maps each component x_i of the source vector $\mathbf{x} \in \mathbb{R}^N$ to the closest point in the code alphabet $\mathcal{C} = \{c_0, c_1, \ldots, c_{2^R-1}\}$, i.e.,

$$x_i^Q = Q(x_i) = \arg\min_{c_j \in \mathcal{C}} \sqrt{(x_i - c_j)^2}, \quad \forall i \in \{1, 2, \ldots, N\}. \tag{2.20}$$

The length of binary vectors acting as labels for code symbols ($= \log_2 |\mathcal{C}|$) is called the *rate* (R) of the code. The set $\mathcal{P}$ of possible representations (i.e., codewords) p_i, is called codebook. For a scalar quantizer, $\mathcal{P} = \mathcal{C}^N$.

One can also consider a general (not a scalar) quantizer $Q(\cdot) : \mathbb{R}^N \to \mathcal{P}$, that operates on entire input vector $\mathbf{x}$. In order to compare different quantizers defined by the ensemble $\mathcal{A} = \{R, \mathcal{P}, Q(\cdot)\}$, a function $d(\mathbf{x}, \mathbf{p})$, called *distortion function*, satisfying

$$d(\mathbf{x}, \mathbf{p}) \geq 0$$
$$d(\mathbf{x}, \mathbf{p}) = 0 \iff \mathbf{x} = \mathbf{p}$$

needs to be introduced as the measure of distortion of the input vector. Furthermore, if we are given a distribution $p(\mathbf{x})$ of the input $\mathbf{x}$, we can use so-called average distortion of the quantizer D, defined as

$$D = D(p, R, \mathcal{P}, Q) = \mathbb{E}\{d(\mathbf{x}, \mathbf{p})\} = \int d(\mathbf{x}, Q(\mathbf{x}))p(\mathbf{x})d\mathbf{x}, \tag{2.21}$$

to characterizes the average performance of $\mathcal{A}$.

In general the case, the question "What is the lowest rate (i.e., number of codewords) for which a given distortion can be achieved?" or alternatively "What is the lowest distortion for which a given rate can be achieved?" was answered in Shannon's seminal paper from 1948 [74]. Both answers define the so-called *rate distortion function*. Assuming that the source alphabet is finite, the rate distortion function for an i.i.d. source $\mathbf{x}$ with distribution $p(\mathbf{x})$ and bounded distortion function $d(\mathbf{x}, \mathbf{p})$ is equal to the associated information rate distortion function, i.e.,

$$R(D) = R^{(I)}(D) = \min_{p(\mathbf{p}|\mathbf{x}): \sum_{(\mathbf{x},\mathbf{p})} p(\mathbf{x})p(\mathbf{p}|\mathbf{x})d(\mathbf{x},\mathbf{p}) \leq D} I(\mathbf{x}; \mathbf{p}), \tag{2.22}$$

where $I(\mathbf{x}; \mathbf{p})$ is the mutual information between $\mathbf{x}$ and $\mathbf{p}$ [22, Theorem 13.2.1][6]. However, obtaining a closed-form expression for the rate distortion function is possible only for a few special cases, e.g., for a Bernoulli discrete source or Gaussian continuous source.

2.4.2 Quantized Compressed Sensing

In many applications, including *magnetic resonance imaging* (MRI) [54, 55, 84, 85], sparse channel estimation [5, 70, 80], photography [38], a coarse quantization of the CS measurements is unavoidable for further *digital signal processing* (DSP). By quantization, i.e., representing continuous values of measurements with values from a finite discrete set, two different errors are produced, namely:

1. a saturation error, which occurs when there exists one or more measurements outside of the *codebook* range.

2. finite-size codebook error, which occurs as continuous source intervals are represented by discrete values.

[6]If the elements of $\mathbf{x}$ are i.i.d. continuous random variables, the sum in (2.22) is replaced by an integral.

These two errors, called *finite-rage* quantization errors, can be controlled by the size of the codebook, i.e., the rate R of the code. Using more bits per measurement to represent the unknown sparse vector in measurement domain would reduce the distortion. In a CS framework however, finding rate-distortion function for a specific recovery algorithm is even more difficult compared to the classical setting. This is because we quantize the measurement vector $\mathbf{y}$, and want to minimize the end-to-end distortion between $\mathbf{x}$ and $\hat{\mathbf{x}} = \hat{\mathbf{x}}(\mathbf{y}) = \hat{\mathbf{x}}(Q(\mathbf{z}))$, which involves a nonlinear measurement system as well as a nonlinear recovery method. Even though some bounds on the distortion are available for BP and *subspace pursuit* (SP) algorithms [25], much of research focused on finding suboptimal CS recovery algorithms from finite-rate quantized measurements - a problem called QCS [13, 14, 23, 41, 45, 46, 48, 49, 51, 76, 78, 83, 86, 87]. For example, counter-measures to mitigate signal distortions caused by saturation errors through measurement rejection or consistent reconstruction were investigated by Laska *et al.* in [50]. On the other hand, as pointed out in [76], a huge body of research on finite-size codebook effects in CS can be assigned to one of the two classes: design of the quantization scheme that works well for a specific CS algorithm; and modifying an existing classical CS reconstruction algorithm to include the quantization in the measurement model. Based on the rate of the quantizer, the algorithms from the last class can be further subdivided into algorithms for 1-bit CS [13, 14, 46, 49, 51, 83, 86] and algorithms for R-bit CS ($R > 1$, i.e., higher rate) [18, 23, 41, 45, 48, 69, 87].

Modified quantization schemes

In [78] the authors use high-rate functional scalar quantization and homotopy continuation to approximate the high rate optimal scalar quantizer under a specific CS reconstruction algorithm, namely LASSO. The authors demonstrate through a numerical experiment that the distoration of the proposed quantizer under LASSO recovery matches the estimated distortion and significantly outperforms a uniform quantizer under LASSO. However, the question of optimality of this approach is still unanswered and one could find a different recovery algorithm with another quantizer that outperforms the proposed one.

In [76] an end-to-end *mean squared error* (MSE) minimizing quantization scheme is proposed that uses the concept of *Analysis-by-Synthesis* (AbS). In AbS one investigates the neighbourhood of a scalar-quantized measurement vector in $\mathcal{C}$, with the aim of finding a representation that will, after applying a nonlinear recovery algorithm, give a lower end-to-end MSE. Results are presented where the OMP algorithm is used as the reconstruction algorithm in the scheme; the proposed scheme allows for use of any CS reconstruction algorithm.

1-bit CS $(R = 1)$

In 1-bit CS one considers the extreme case of scalar quantization of CS linear mixtures $\mathbf{z} = \mathbf{A}\mathbf{x}$, where each z_i $(i \in [m])$ is mapped to a single bit y_i. Bits $\{y_i\}_{i=1}^m$, arranged in the vector $\mathbf{y} \in \{\pm 1\}^m$, capture only the information about the signs of the inner products of the unknown vector and the corresponding measurement vectors, i.e.,

$$\mathbf{y} = Q(\mathbf{z}) = \operatorname{sign}(\mathbf{A}\mathbf{x}). \tag{2.23}$$

In other words, each bit y_i tells us on which side of the hyperplane orthogonal to the associated measurement vector the unquantized measurement vector lies on, i.e., in which half-space of $\mathbb{R}^N$ the unquantized measurement lives in. By taking m measurements, the intersection of m half-spaces creates the orthant[7] $\mathcal{O}_{\bar{z}}$, such that $\mathbf{z} \in \mathcal{O}_{\bar{z}}$. Since the measurement bits only capture the signs of the linear mixtures, any information about the ℓ_2 norm of the unknown vector is lost with 1-bit quantization. This property of the 1-bit quantizer can be shown analytically, by observing that

$$Q(a\mathbf{z}) = \operatorname{sign}(\mathbf{A}a\mathbf{x}) = \operatorname{sign}(\mathbf{A}\mathbf{x}) = Q(\mathbf{z}), \tag{2.24}$$

for any positive a. As a consequence, we aim to reconstruct the unknown sparse vector up to a normalisation constant, and restrict the solution to have unit ℓ_2 norm, i.e., $\|\mathbf{x}\|_2 = 1$.

In the last few years, many authors modified classical CS algorithms to estimate a sparse vector from 1-bit CS measurements in (2.26) [13, 14, 46, 49, 51, 83, 86]. However, as opposed to the classical problem where the measurements of the estimate $\hat{\mathbf{x}}$ should be close to $\mathbf{y}$ in the ℓ_2 sense, i.e., $\|\mathbf{A}\hat{\mathbf{x}} - \mathbf{y}\|_2$ should be small. Here one requires that the solution $\hat{\mathbf{x}}$ is consistent with the quantized measurements, i.e., $\mathbf{y} \in \mathbb{R}^m$ and $\mathbf{A}\hat{\mathbf{x}}$ should be close in some sense. For example, the authors of [14] consider the cost function of LASSO in (2.13), and modify it by using a term that promotes consistency with the 1-bit measurements, namely

$$\hat{\mathbf{x}}_{\text{FPC}} = \arg\min_{\mathbf{x}} \|\mathbf{x}\|_1 + \lambda \sum_{i=1}^m f(\mathbf{Y}\mathbf{A}\mathbf{x})_i, \quad \text{s.t.} \quad \|\mathbf{x}\|_2 = 1, \tag{2.25}$$

[7]An orthant in $\mathbb{R}^m$ is the set of all vectors that have the same sign pattern [46].

where $\mathbf{Y} = \operatorname{diag}(\mathbf{y})$, $f(x) = \frac{1}{2}x^2u(-x)$, and $u(x)$ is the unit step function. To solve (2.25), the authors propose an iterative algorithm based on the projected gradient descent method, and additionally normalize the solution at each step to make the solution lie on the unit sphere. Finally, the authors show results of numerical experiments to demonstrate that the recovery is possible in the oversampling regime. Note however, that in the 1-bit CS regime, oversampling is not a sign of bad design, but more an unavoidable consequence of such a coarse quantization.

Sunsequently, a new algorithm called *matching sign pursuit* (MSP) was presented in [13], that combines the principle of consistent reconstruction with greedy sparse reconstruction, while a faster algorithm, called *restricted-step shrinkage* (RSS), with convergence guarantees was presented in [51] which shows significantly better performance in terms of average *signal-to-noise ratio* (SNR).

In [46] the authors provide probably the first theoretical results on QCS, namely they give a lower bound on the best achievable reconstruction error from noiseless 1-bit measurements as a function of m and k, and characterize the ability of a measurement system to fight the noise in the measurements by introducing a property called ϵ-stable embedding. Furthermore, the authors complement the work with two algorithms called *binary iterative hard thresholding* (BIHT) algorithm, which draw some similarities with the IHT algorithm for classical CS. In [83], the authors propose the *adaptive outlier pursuit* (AOP) algorithm based on the BIHT algorithms, which iteratively detects sign flips (i.e., errors) in the measurements and recover a sparse vector from the estimated “correct” measurements.

Finally, the recovery of sparse vectors from noiseless 1-bit CS measurements based on the *generalized approximate message passing* (GAMP) algorithm was introduced in [49]. It was further exploited in [86] where the authors consider different ways corrupting the 1-bit CS measurements by noise: *additive white Gaussian noise* (AWGN) before quantization; flipping each bit with probability p_e after quantization, etc. The latter corresponds to the case where one sends 1-bit CS measurements through a *binary symmetric channel* (BSC) with bit error probability p_e.

R-bit CS ($R > 1$)

In R-bit CS ($R > 1$) one considers the scalar quantization of CS linear mixtures $\mathbf{z} = \mathbf{A}\mathbf{x}$, where each z_i ($i \in [m]$) is mapped to a symbol $c_k \in \mathcal{C}$ according to (2.20) and placed at the i-th position of vector $\mathbf{y}$. Hence, we want to estimate a sparse $\mathbf{x}$ from

$$\mathbf{y} = Q(\mathbf{z}) = Q(\mathbf{A}\mathbf{x}), \tag{2.26}$$

where the quantizer function $Q(\cdot)$ is applied component-wise.

To solve this problem, the authors in [18] consider quantization effects as bounded additive noise and consider the solution of BPDN problem, defined in (2.10). They prove that, assuming bounded noise power ($\|\mathbf{w}\|_{\ell_2} \leq \epsilon$), the measurement matrix $\mathbf{A}$ obeys a uniform uncertainty principle[8] and the vector is sufficiently sparse, the recovery with (2.10) is stable, i.e., within the noise level.

In [87] the authors assume that the $\mathbf{w}$ is zero-mean i.i.d. Gaussian norm vector and present two algorithms which use convex optimization methods to minimize two convex cost functions, each having the standard l_1 term that promotes sparsity and a convex term that promotes fidelity to the quantized measurements, namely a convex *maximum likelihood* (ML) term or a convex quasi-LS term. Interestingly, this is one of the few works that includes a noise term before quantizing the measurements.

A sigma delta quantizer for CS was proposed in [41], which lacks the ability to fight distortions introduced by other noise sources.

Modifications of the BP algorithm from [18] that solves (2.5) and SP from [23, 69], that now exploit information about quantization of the linear mixtures are presented in [23].

Another modification of the BPDN defined in (2.10) is done in [45]. Here the authors consider different norms ($p > 2$) of the data fidelity term, i.e., $\|\mathbf{y} - \mathbf{Ax}\|_p$, to obtain an algorithm for quantized CS called BPDN$_p$ algorithm. The idea is that for $p > 2$, the data fidelity term promotes quantization consistency, i.e., measurements should be close to the requantized estimated noiseless CS measurements.

In [48] the authors propose a GAMP based algorithm called *message-passing de-quantization* (MPDQ) algorithm, which provides an approximation of the MMSE estimate. Furthermore, if the measurement matrix is i.i.d. Gaussian, the asymptotic error performance of the algorithm can be predicted using *state evolution* (SE) for the GAMP algorithm, which can then be used to optimize the cells $\{s_i\}_{i=1}^{2R}$ when considering a scalar quantizer. However, this paper does not offer the closed-form expressions for the nonlinear updates of the algorithm as well as a deeper analysis of the GAMP algorithm for noisy QCS.

[8]Uniform uncertainty principle essentially states that the measurement matrix obeys the RIP [18].

Chapter 3

Approximate Message Passing Algorithms

Regardless of the practical problem at consideration (i.e., quantization or miscalibration), the core task is, nonetheless, the recovery of a sparse signal from noisy measurements. Since both studied problems introduce nonlinear artefacts, I am interested in the recovery of sparse signals from measurements corrupted with nonlinear distortions. Among many *compressed sensing* (CS) recovery algorithms operating with linear measurements in additive noise, the class of *approximate message passing* (AMP) algorithms stands out as the one with the most potential for solving CS problems with nonlinear measurements. Therefore, in this chapter, I discuss some important aspects of the AMP algorithm. Moreover, as my own contribution, I derive closed-form expressions for the denoiser functions of the *Bayesian approximate message passing* (BAMP) algorithm for a particularly interesting prior: a weighted average of n-Gaussian distributions, each with potentially different mean and different variance.

3.1 Approximate Message Passing

The term AMP refers to a class of algorithms that are based on Gaussian and quadratic approximations of the loopy belief propagation on dense graphs. In the corresponding graphical model, the entries of the unknown vector are represented by so-called variable nodes, the entries of the measurement vector are represented by variable nodes, and the edges represent statistical dependencies between nodes. Since, in the general case, the measurement matrix is dense, and the entries of the unknown vector are *independent and identically distributed* (i.i.d.), the corresponding graph is an instance of a dense bipartite graph. Whether we are interested in an approximate *minimum* *mean squared error* (MMSE) estimate or a *maximum a posteriori* (MAP) estimate, we apply sum-product or max-sum belief propagation on a graph with loops. As a result, the AMP algorithm approximates the computationally intractable high-dimensional integration involved with calculating $\mathbb{E}\{\mathbf{x}|\mathbf{y}\}$ or $\arg\max_{\mathbf{x}} p(\mathbf{x}|\mathbf{y})$with a highly efficient iterative procedure [62].

Given that the AMP algorithm [33–35, 57] lies at the heart of this thesis, in this chapter, I will take a deeper look at the algorithm. The rest of this chapter is organized as follows.

In the first section, I provide an overview of the derivation of the algorithm. The overview is supplemented with some comments on the steps of the derivation, that might have been skipped or unintuitive in the original derivation. Furthermore, I discuss computational complexity of the AMP algorithm, and show why it is in the order of the other iterative thresholding algorithms. I conclude the first section with a discussion on the *state evolution* (SE) of the AMP algorithm. As the name indicates, with SE one can predict the evolution of some parameter of the AMP algorithm. Put differently, one can predict the value of a parameter of the AMP algorithm across iterations t.

As I will show in the first section, the AMP algorithm assumes i.i.d. Laplacian prior for the entries of the unknown vector. As there is nothing special about the Laplacian prior for AMP to be derived, one can also consider other prior distributions. If the source prior is known, or can be estimated from measurements, in Section 3.2, I show how to use the BAMP algorithm to get more accurate estimates compared to the classical AMP algorithm. Subsequently, I consider a prior that consists of a weighted average of n-Gaussian distributions, each with potentially different mean and different variance. By choosing appropriate values for the means and the variances, one can model many practically interesting priors. Moreover, by picking a very small but still non-zero variance, one can even approximate discrete *probability mass functions* (pmfs). As my own contribution, for the prior consisting of weighted average of n-Gaussians, I derive closed-form expressions for the denoiser functions of the BAMP algorithm.

Similar to the reasoning which led to the BAMP algorithm, one can also argue that there is also nothing special about the Gaussian noise model for AMP to be derived. Considering a general distribution describing the component-wise distortion of the CS linear mixtures, one can once again approximate the message passing algorithm. This approach leads to the *generalized approximate message passing* (GAMP) algorithm, which allows us to incorporate the knowledge of nonlinear acquisition of the CS measurements in the estimation. Since in the following chapters I frequently refer to the GAMP algorithm, I conclude this chapter by briefly presenting the GAMP algorithm.

3.1.1 Message passing algorithm for basis pursuit

For simplicity, it is assumed that $a_{i,j} \in \{-1/\sqrt{m}, +1/\sqrt{m}\}$, the measurements of a sparse vector $\mathbf{x}$ are obtained according to (6.6), and we are interested in solving the *basis pursuit* (BP) problem defined in (2.5). Instead of assuming that a fixed number of entries of $\mathbf{x}$ are nonzero, in AMP we take a probabilistic approach. In particular, we assume that the entries of $\mathbf{x}$ are i.i.d. random variables, distributed according to the same sparsity promoting distribution. For example, the Laplace distribution is a good candidate since we can promote sparsity by reducing the variance of the distribution. Faced with an observation vector $\mathbf{y}$ and using the Bayes' theorem, we can write the conditional distribution $p_{\mathsf{x}|\mathsf{y}}(\mathbf{x}|\mathbf{y})$ as

$$p_{\mathsf{x}|\mathsf{y}}(\mathbf{x}|\mathbf{y}) = \frac{1}{p(\mathbf{y})} p(\mathbf{x}) p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z} \prod_{i=1}^{N} \exp(-\beta |x_i|) \prod_{a=1}^{m} \delta(y_a = (\mathbf{A}\mathbf{x})_a). \tag{3.1}$$

The authors of [33, 34] argue that when $\beta \to \infty$, the mass of this distribution concentrates around the solution of the basis pursuit problem. Therefore, provided that the maximizer is unique, the BP problem becomes the problem of finding marginal distributions of $p_{\mathsf{x}|\mathsf{y}}(\mathbf{x}|\mathbf{y})$. Since finding exact marginals $p_{\mathsf{x}_i|\mathsf{y}}(x_i|\mathbf{y})$ for all $i \in [N]$ requires multidimensional integration over $\mathbf{x}_{\sim i}$[1], the problem is infeasible even for small problem sizes.

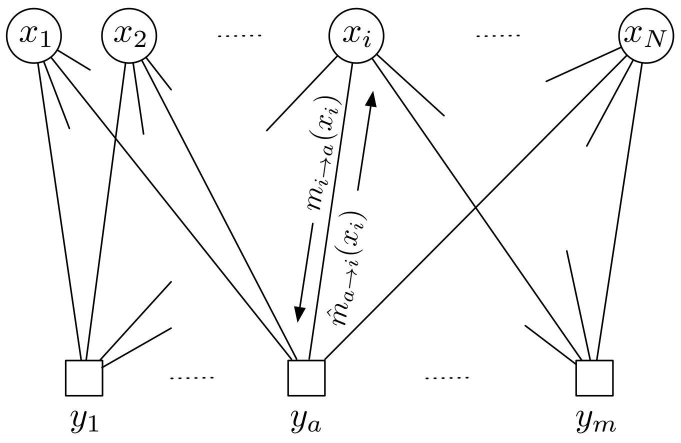
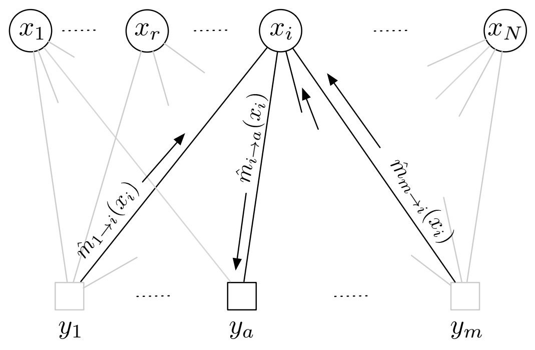
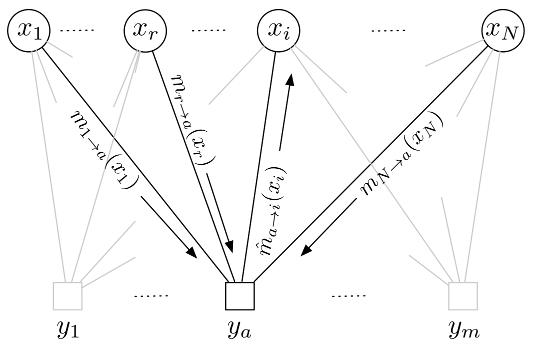


Fig. 3.1 Factor graph corresponding to the BP problem.

[1]$\mathbf{a}_{\sim i}$ is a vector containing all but i-th element of $\mathbf{a}$



(a) Messages involved with calculation of messages from variable nodes to factor nodes.



(b) Messages involved with calculation of messages from factor nodes to variable nodes.

Fig. 3.2 Factor graph corresponding to the BP problem.

To approximate MMSE estimate $\mathbb{E}\{\mathbf{x}|\mathbf{y}\}$ or *maximum likelihood* (ML) estimate $\arg\max_{\mathbf{x}} p(\mathbf{x}|\mathbf{y})$, we turn to efficient optimization algorithms based on belief propagation on factor graphs, namely the sum-product and the max-sum algorithm [62]. Here, one starts by constructing a graphical model (i.e., factor graph) $\mathcal{G} = (\mathcal{V}, \mathcal{F}, \mathcal{E})$, consisting of a set of variable nodes $\mathcal{V}$ corresponding to the entries of the unknown $\mathbf{x}$ (denoted by circles), a set of factor nodes $\mathcal{F}$ corresponding to the observations $\mathbf{y}$ (denoted by a square), and a set of edges $\mathcal{E}$, where each edge indicates statistical dependency between connected nodes. In Figure 3.1, an example of a complete bipartite graph is shown, where every factor node is connected to all variable nodes, and vice versa, every variable node is connected to all factor nodes. Belief propagation iteratively updates a set of functions of the optimization variables (messages) associated to directed edges in $\mathcal{E}$ [62]. In particular, for each direction of the edge $e \in \mathcal{E}$, a message corresponding to a scaled distribution is sent in that direction. Those scaled distributions are updated according to the update rules of the sum-product algorithm, namely:

(1) (Figure 3.2a) At each variable node x_i, multiply prior distribution (local belief) of x_i with all incoming messages about x_i except the one from destination factor node y_a, and send the resulting message to y_a. Repeat the process for all $y_a \in \mathcal{F}$.

(2) (Figure 3.2b) At each factor node y_a multiply the local constraint $f(y_a)$ with all incoming messages except from x_i, marginalise over $\mathbf{x}_{\sim i}$, and send the resulting scaled distribution to x_i. Repeat the process for all $i \in \mathcal{V}$. .

3.1.2 Derivation of the algorithm

We start by drawing the associated bipartite graph $\mathcal{G} = (\mathcal{V}, \mathcal{F}, \mathcal{E})$ given in Figure 3.1. Here, $\mathcal{V}$ is the set of the N components of $\mathbf{x}$, $\mathcal{F}$ is the set of m measurements in $\mathbf{y}$, and the edge set $\mathcal{E}$ is given by $\mathcal{E} = \mathcal{V} \times \mathcal{F} = \{(i, a) : i \in [N], a \in [m]\}$.

The belief propagation algorithm iteratively updates the $2mN$ variables of the algorithm

$$\begin{aligned} m^{(t+1)}_{i\to a}(x_i) &\cong p(x_i) \prod_{b\neq a} \hat{m}^{(t)}_{b\to i}(x_i) \\ &= e^{-\beta|x_i|} \prod_{b\neq a} \hat{m}^{(t)}_{b\to i}(x_i), \end{aligned} \tag{3.2}$$

$$\begin{aligned} \hat{m}^{(t)}_{a\to i}(x_i) &\cong \int \prod_{j\neq i} m^{(t)}_{j\to a}(x_j)\, p(y_a|z_a; x_i)\, d\mathbf{x}_{\sim i} \\ &= \mathbb{E}_{\mathsf{x}_{\sim i}} \Big\{ p(y_a|z_a; x_i) \Big\} \\ &= \mathbb{E}_{\mathsf{x}_{\sim i}} \Big\{ \delta(y_a - z_a; x_i) \Big\}, \end{aligned} \tag{3.3}$$

where in (3.2) we implicitly assumed an i.i.d. Laplacian prior for $\mathbf{x}$, $\mathbf{z}$ is the vector of linear mixtures (i.e., $\mathbf{z} = \mathbf{A}\mathbf{x}$), and $\cong$ denotes equality up to a normalization constant.

Message passing algorithm

Keeping track of the terms in (3.2) and (3.3), i.e., functions over $\mathbb{R}$, leads to prohibitive complexity. To simplify the updates from above, the authors in [33, 34, 57] utilize the Berry-Esseen theorem[2] [77] to approximate $m^{(t+1)}_{i\to a}(x_i)$, and $\hat{m}^{(t)}_{a\to i}(x_i)$ with functions that belong to a family of functions characterized with a small set of parameters. This way, instead of sending functions over $\mathbb{R}$, one needs to send a vector of parameters to perform the updates of the sum product algorithm.

Approximation of $\hat{m}^{(t)}_{a\to i}(x_i)$:

Theorem 5 [6, The Berry-Esseen theorem] *Let* $\mathsf{x}_1, \mathsf{x}_2, \ldots, \mathsf{x}_n$ *be independent random variables with respective means* μ_i *and variances* σ_i^2. *We call*

$$\lambda(\mathsf{x}_i) = \begin{cases} \mathbb{E}\{|\mathsf{x}_i|^3\}/\sigma_k^2, & \text{if } \sigma_k^2 \neq 0, \\ 0, & \text{if } \sigma_k^2 = 0, \end{cases} \tag{3.4}$$

[2]The Berry-Esseen theorem is a more general version of the central limit theorem.

the moment-ratio of x_i*, and set* $\Lambda = \max\{\lambda(\mathsf{x}_1), \lambda(\mathsf{x}_2), \ldots, \lambda(\mathsf{x}_n)\}$*. Consider the sum* $\mathsf{x} = \sum_{i=1}^n \mathsf{x}_i$*, which has mean* $\mu = \sum_{i=1}^n \mu_i$*, and variance* $\sigma^2 = \sum_{i=1}^n \sigma_i^2$*, and denote with* $F(x)$ *the cumulative distribution function (cdf) of* x*. The least upper bound of the modulus of the difference between* $F(x)$ *and the cdf of the associated normal distribution is upper bounded by a constant that depends only on the ratio* Λ/σ*, i.e.,*

$$\sup_{x\in\mathbb{R}} \Big| F(x) - G((x-\mu)/\sigma) \Big| \leq 1.88\,\frac{\Lambda}{\sigma}. \tag{3.5}$$

Corollary 1 *Suppose* $\mu_i = 0$*,* $\sigma_i^2 = \sigma_0^2$*, and* $\mathbb{E}\{|\mathsf{x}_i|^3\} \leq \rho$ *for* $\forall i \in [n]$*. It follows that, in the limit as* $n \to \infty$*,*

$$\lim_{n\to\infty} \sup_{x\in\mathbb{R}} \Big| F(x) - \Phi((x)) \Big| \leq \lim_{n\to\infty} 1.88\,\frac{C\rho}{\sigma_0^3\sqrt{n}} = 0. \tag{3.6}$$

According to the Berry-Esseen theorem (Theorem 5), distribution of the sum of n random variables (with bounded absolute third moment) converges to a normal distribution as the sample size increases to infinity. It follows that the distribution of $\sum_{j\neq i} a_{aj}\mathsf{x}_j$ can be well approximated with a Gaussian distribution with mean $\sum_{j\neq i} a_{aj} x_{j\to a}^{(t)}$ and variance $\sum_{j\neq i} a_{ar}^2 \tau_{j\to a}^{(t)}$, i.e.,

$$\sum_{j\neq i} a_{aj}\mathsf{x}_j \mathrel{\dot\sim} \mathcal{N}\Big(\sum_{j\neq i} a_{aj} x_{j\to a}^{(t)}, \sum_{j\neq i} a_{aj}^2 \tau_{j\to a}^{(t)}\Big), \tag{3.7}$$

where $\dot\sim$ is interpreted as "is approximately distributed as", and $x_{j\to a}^{(t)}$ and $\tau_{j\to a}^{(t)}$ denote mean and variance of the message $m_{j\to a}^{(t)}(x_j)$, i.e.,

$$\begin{aligned} x_{j\to a}^{(t)} &= \mathbb{E}\Big\{x_j \,\Big|\, m_{j\to a}^{(t)}(x_j)\Big\}, \\ \tau_{j\to a}^{(t)} &= \operatorname{var}\Big\{x_j \,\Big|\, m_{j\to a}^{(t)}(x_j)\Big\}. \end{aligned} \tag{3.8}$$

Therefore,

$$\mathsf{z}_a|\mathsf{x}_i = a_{ai}x_i + \sum_{j\neq i} a_{aj}\mathsf{x}_j \mathrel{\dot\sim} \mathcal{N}\Big(a_{ai}x_i + \sum_{j\neq i} a_{aj} x_{j\to a}^{(t)}, \sum_{j\neq i} a_{aj}^2 \tau_{j\to a}^{(t)}\Big), \tag{3.9}$$

and

$$\begin{aligned}\hat{m}_{a\to i}^{(t)}(x_i) &= \mathbb{E}_{\mathbf{x}_{\sim i}}\Big\{p(y_a|z_a;x_i)\Big\}\\ &\approx \int p(y_a|z_a;x_i)\,\mathcal{N}\Big(z_a;\,a_{ai}x_i+\sum_{j\neq i}a_{aj}x_{j\to a}^{(t)},\,\sum_{j\neq i}a_{aj}^2\tau_{j\to a}^{(t)}\Big)\,dz_a\\ &= \int \delta(z_a-y_a)\,\mathcal{N}\Big(z_a;\,a_{ai}x_i+\sum_{j\neq i}a_{aj}x_{j\to a}^{(t)},\,\sum_{j\neq i}a_{aj}^2\tau_{j\to a}^{(t)}\Big)\,dz_a\\ &= \mathcal{N}\Big(y_a;\,a_{ai}x_i+\sum_{j\neq i}a_{aj}x_{j\to a}^{(t)},\,\sum_{j\neq i}a_{aj}^2\tau_{j\to a}^{(t)}\Big)\\ &= \mathcal{N}\Big(a_{ai}x_i;\,y_a-\sum_{j\neq i}a_{aj}x_{j\to a}^{(t)},\,\sum_{j\neq i}a_{aj}^2\tau_{j\to a}^{(t)}\Big).\end{aligned} \tag{3.10}$$

It follows from (3.10), that $2(N-1)$ scalar parameters are sufficient to approximate the message $\hat{m}_{a\to i}^{(t)}(x_i)$.

Approximation of $m_{i\to a}^{(t+1)}$: Since the messages $\hat{m}_{a\to i}^{(t)}(x_i)$ can be well approximated by Gaussian distributions, going back to (3.2), we can conclude that the messages from the variable nodes to the factor nodes $m_{i\to a}^{(t+1)}(x_i)$ can be well approximated by the product of a Gaussian distribution and the prior (Laplace distribution). It turns out that the mean $x_{j\to a}$ and the variance $\tau_{j\to a}$ of $m_{i\to a}^{(t+1)}(x_i)$, as $\beta\to\infty$, can be approximated as

$$x_{i\to a}^{(t+1)} = \eta\Big(\sum_{b\neq a}a_{bi}z_{b\to i}^t;\,\hat{\tau}^t\Big),\qquad \tau_{i\to a}^{(t+1)}\approx\hat{\tau}^t\,\eta'\Big(\sum_{b\neq a}a_{bi}z_{b\to i}^t;\,\hat{\tau}^t\Big), \tag{3.11}$$

where $\eta(x;b)$ is the soft thresholding function defined in (2.19), and $\hat{\tau}^t$ is an edge-independent approximation of $\hat{\tau}_{a\to i}^t$. To calculate $\hat{\tau}^t$, it is observed in [57] that

$$\hat{\tau}^{(t+1)} = \sum_i^N a_{ai}^2\tau_{i\to a}^{(t+1)} = \frac{1}{m}\sum_i^N \tau_{i\to a}^{(t+1)} \approx \frac{1}{m}\sum_i^N \hat{\tau}^t\,\eta'\Big(\sum_b a_{bi}z_{b\to i}^t;\,\hat{\tau}^t\Big). \tag{3.12}$$

The above message passing algorithm iteratively computes $2mN$ messages, namely mN variances $z_{a\to i}^t$, and mN means $x_{i\to a}^{(t)}$. Even though these messages (formerly functions over $\mathbb{R}$) simplify to real numbers, the number of the messages still introduces prohibitive computational complexity, and further approximations are needed.

Approximation of the message passing algorithm

To further simplify the message passing algorithm, since $a_{i,j}\in\{-1/\sqrt{m},+1/\sqrt{m}\}$, it is assumed that $\mu_{i\to a}^{(t)}$ as well as $z_{a\to i}^t$ differ in terms that are of order $\mathcal{O}(\frac{1}{\sqrt{N}})$ for different a and i, respectively, i.e.,

$$x_{i\to a}^{(t)} = x_i^{(t)} + \delta x_{i\to a}^{(t)}, \qquad z_{a\to i}^{(t)} = z_a^{(t)} + \delta z_{a\to i}^{(t)}, \tag{3.13}$$

where $\delta x_{i\to a}^{(t)}$ and $\delta z_{a\to i}^{(t)}$ are $\mathcal{O}(\frac{1}{\sqrt{N}})$. These assumptions, although not strictly proven to be correct, lead to the following approximations in the large system limit (i.e., $m, N \to \infty$, $\delta = const$) for the messages $z_{a\to i}^t$ and $x_{i\to a}^{(t)}$

$$z_{a\to i}^{(t)} \approx z_a^{(t)} \quad = y_a - \sum_{j\in[N]} a_{aj} x_j^{(t)} + \frac{1}{\delta} z_a^{(t)} \Big\langle \eta'\Big(\mathbf{A z}^{(t)} + \mathbf{x}^t, \hat{\tau}^t\Big)\Big\rangle, \tag{3.14}$$

$$x_{i\to a}^{(t)} \approx x_i^{(t+1)} = \eta\Big(\sum_{b\in[n]} a_{bi} z_b^{(t)} + x_i^t, \hat{\tau}^t\Big), \tag{3.15}$$

where $\langle \cdot \rangle$ denotes the average value of the entries of its input vector. Equations (3.12), (3.14) and (3.15)) give final expressions for the tuning free AMP algorithm which can be written in vector notation as

$$\begin{aligned} \mathbf{x}^{(t+1)} &= \eta\Big(\mathbf{A}^{(t)}\mathbf{z}^{(t)} + \mathbf{x}^t, \hat{\tau}^t\Big), \qquad \hat{\tau}^t = \frac{\hat{\tau}^{(t-1)}}{\delta}\Big\langle \eta'\Big(\mathbf{A}^{(t)}\mathbf{z}^{(t-1)} + \mathbf{x}^t, \hat{\tau}^{(t-1)}\Big)\Big\rangle, \\ \mathbf{z}^{(t)} &= \mathbf{y} - \mathbf{A x}^{(t)} + \frac{1}{\delta}\mathbf{z}^{(t-1)}\Big\langle \eta'\Big(\mathbf{A}^{(t)}\mathbf{z}^{(t)} + \mathbf{x}^t, \hat{\tau}^{(t-1)}\Big)\Big\rangle. \end{aligned} \tag{3.16}$$

The details of above approximations can be found in [57].

AWGN output channel (AMP for soft constraints)

Here, we assume that the CS measurements are corrupted with *additive white Gaussian noise* (AWGN), i.e.,

$$\mathbf{y} = \mathbf{A x} + \mathbf{w}, \tag{3.17}$$

where $\mathbf{w}$ is an i.i.d. zero-mean AWGN noise vector with the covariance matrix $\sigma_w^2 I$. Now suppose that we want to recover $\mathbf{x}$ by solving the *basis pursuit denoising* (BPDN) problem given in (2.10). As before, one needs to write down the conditional distribution $p_{\mathsf{x}|\mathsf{y}}(\mathbf{x}|\mathbf{y})$, which, respecting the measurement model in (3.17), can be written as

$$p_{\mathsf{x}|\mathsf{y}}(\mathbf{x}|\mathbf{y}) = \frac{1}{Z}\prod_{i=1}^{N} \exp(-\beta\lambda|x_i|) \prod_{a=1}^{m} \exp\Big(-\frac{\beta}{2}(y_a - (\mathbf{A x})_a^2\Big). \tag{3.18}$$

Note that, compared to the conditional distribution $p_{\mathsf{x}|\mathsf{y}}(\mathbf{x}|\mathbf{y})$ of the the BP problem given in (3.1), the distribution in (3.18) differs in the term that describes the measurement process, and in the weighting term λ. Just as in the case of the AMP algorithm for hard constraints, one proceeds with applying the Berry-Esseen theorem to approximate messages from factor nodes to variable nodes by Gaussian distributions with certain means and variances, and Taylor expansions to approximate means of the messages in the opposite direction with edge-independent terms. Finally, we end up with the following AMP algorithm for soft constraints:

$$\begin{aligned}
\mathbf{u}^{(t)} &= \mathbf{A}^{(t)}\mathbf{z}^{(t-1)} + \mathbf{x}^{(t-1)}, \\
\mathbf{x}^{(t)} &= \eta\Big(\mathbf{u}^{(t)}, \lambda + \gamma^{(t)}\Big), \\
\mathbf{z}^{(t)} &= \mathbf{y} - \mathbf{A}\mathbf{x}^{(t)} + \frac{1}{\delta}\mathbf{z}^{(t)}\Big\langle \eta'\Big(\mathbf{u}^{(t)}, \lambda + \gamma^{(t)}\Big)\Big\rangle, \\
\gamma^{(t+1)} &= \frac{\lambda + \gamma^{(t)}}{\delta}\Big\langle \eta'\Big(\mathbf{u}^{(t)}, \lambda + \gamma^{(t)}\Big)\Big\rangle.
\end{aligned} \tag{3.19}$$

Comparing the algorithm with the one given in (3.16), we can see that the only difference is the way the soft thresholding parameter is calculated. Additionally, in (3.19) one needs to tune the λ parameter which trades off between l_1 and l_2 penalties in (2.10). Poor selection of this parameter can lead to a high reconstruction error as well as slow convergence of the algorithm [57, p. 68]. In [57], Maleki gives a recipe how to select λ for certain classes of sparse signals, using the maximin approach, i.e., selecting λ which gives the lowest *mean squared error* (MSE) for the worst case signal. Arguing that this approach is too pessimistic, the authors in [63] propose a parameter free optimal AMP. This algorithm asymptotically, as $N \to \infty$, at the same time achieves the MMSE and the highest convergence rate. Assuming that the argument $\mathbf{w}^{(t)}$ of the soft thresholding function $\eta(\mathbf{w}^{(t)}; \tau)$ is given by[3]

$$\mathbf{w}^{(t)} = \mathbf{A}^{(t)}\mathbf{z}^{(t)} + \mathbf{x}^{(t)} = \mathbf{x} + \sigma^{(t)}\mathbf{v}^{(t)}, \tag{3.20}$$

where $\mathbf{v}^{(t)}$ is an i.i.d. standard Gaussian noise vector at iteration t, the authors apply the approximate gradient descent algorithm to find the soft thresholding parameter τ which minimizes Stein's unbiased risk estimate of the risk function $r(\tau;\sigma) = \frac{1}{N}\mathbb{E}_{\mathbf{x},\mathbf{u}}\|\eta(\mathbf{x} + \sigma\mathbf{u};\tau) - \mathbf{x}\|_2^2$, where $\mathbf{u} \sim \mathcal{N}(0, I)$.

Using a very simple approach, in (3.19) one can approximate $\lambda + \gamma^{(t)}$ by the mean empirical power of the residual, i.e., $b = \sqrt{\sigma_w^2 + \frac{1}{m}\|\mathbf{z}^{(t)}\|^2}$, which is correct in the large system limit and showed good empirical results for moderate-size problems [62]. If the noise variance σ_w^2 is unknown we can simply use $b = \sqrt{\frac{1}{m}\|\mathbf{z}^{(t)}\|^2}$ as it has shown good empirical performance [17, Chapter 9.5.1]. These choices of the threshold parameter

[3]Correctness of this assumption will be addressed in Subsection 3.1.4.

avoid determining the optimal λ for each pair (δ, ρ), and to my experience give a more stable implementation of the AMP algorithm for a large set of pairs (δ, ρ).

3.1.3 Computational complexity

The AMP algorithm, given in Algorithm 4, is a highly efficient algorithm whose computational complexity is similar to other iterative thresholding algorithms. The most complex operation of the algorithm is matrix vector multiplication, whose cost is $\mathcal{O}(N^2)$. Additionally, even tough the computation of the thresholding function $\eta(\cdot;\cdot)$ might be costly, it needs to be applied component-wise. Hence, the cost of this step is $\mathcal{O}(N)$ per iteration of the algorithm.

The AMP algorithm, just as other iterative thresholding algorithms, terminates when some stopping criterion is met. For example, one can run the algorithm for $t_{\max}$ iterations, where $t_{\max}$ is typically in the order of N or less. Another common choice is to run the algorithm as long as the l_2 norm of the relative difference of the solutions between two successive iterations is greater than some predefined small number $\varepsilon_{\text{stop}}$ (e.g., 10^{-3}), i.e., $\|\hat{\mathbf{x}}^{(t)} - \hat{\mathbf{x}}^{(t-1)}\|_2 \geq \varepsilon_{\text{stop}} \|\hat{\mathbf{x}}^{(t)}\|_2$. Having larger $t_{\max}$ or smaller $\varepsilon_{\text{stop}}$ increases the reconstruction accuracy at the cost of higher computational complexity.

Algorithm 4 Approximate message passing algorithm

```
Input: measurement matrix $\mathbf{A}$, measurement vector $\mathbf{y}$, noise level $\sigma_w^2$ (optional), soft
    threshold parameter $\lambda$, stopping threshold $\varepsilon_{\text{stop}}$ or $t_{\max}$
Initialization: $t = 0$, $\hat{\mathbf{x}}^{(0)} = \mathbf{0}_{N\times 1}$, $\mathbf{z}^{(0)} = \mathbf{y}$, $\gamma^{(1)} = \frac{1}{\delta}\,\epsilon\,\sigma_x^2$
 do:
  1: $t = t + 1$                                          ▷ increment iteration counter
  2: $\mathbf{u}^{(t)} = \mathbf{A}^{(t)}\mathbf{z}^{(t-1)} + \mathbf{x}^{(t-1)}$          ▷ decouple measurements
  3: $\mathbf{x}^{(t)} = \eta(\mathbf{u}^{(t)}, \lambda + \gamma^{(t)})$          ▷ denoise, i.e., apply thresholding
  4: $\mathbf{z}^{(t)} = \mathbf{y} - \mathbf{A}\mathbf{x}^{(t)} + \frac{1}{\delta}\mathbf{z}^{(t-1)}\langle\eta'(\mathbf{u}^{(t)}, \lambda + \gamma^{(t)})\rangle$          ▷ calculate residual
  5: $\gamma^{(t+1)} = \frac{\lambda+\gamma^{(t)}}{\delta}\langle\eta'(\mathbf{u}^{(t)}, \tau^{(t)})\rangle$          ▷ calculate soft threshold parameter
 while $t \leq t_{\max}$ or $\|\hat{\mathbf{x}}^{(t)} - \hat{\mathbf{x}}^{(t-1)}\|_2 \geq \varepsilon_{\text{stop}}\,\|\hat{\mathbf{x}}^{(t)}\|_2$
Output: $\hat{\mathbf{x}} = \mathbf{x}^{(t)}$
```

3.1.4 State evolution

The goal of the *state evolution* (SE) is to analyze the AMP algorithm in the large system limit, i.e., $N, m \to \infty$ while $\delta = m/N = const$, at any iteration t. As the name indicates, with SE we predict the value (i.e., evolution) of some parameter (i.e., state)

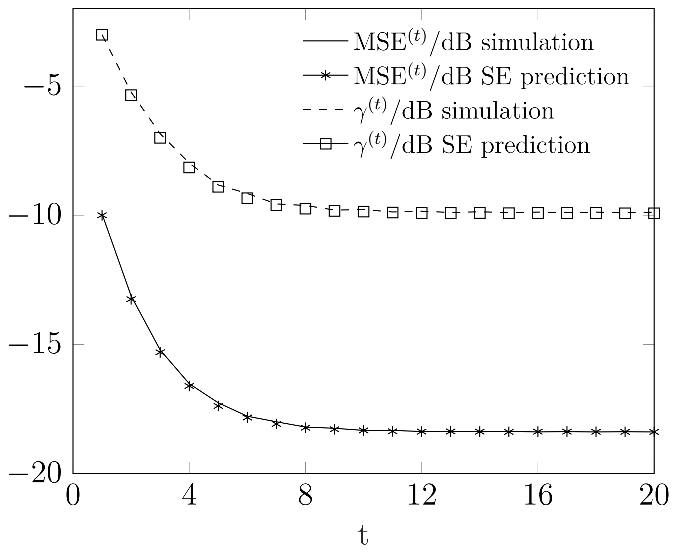


Fig. 3.3 SE prediction of MSE$^{(t)}$ and $\gamma^{(t)}$ of the AMP algorithm in (3.19) against iteration counter t. Parameters: $N = 1000$, $\lambda = 0.3$, $\delta = 0.4$, $\epsilon = 0.1$, $\sigma^2_{\mathsf{w}} = -35$dB. The results of 100 MC simulations are averaged and presented with the blue curves.

of the algorithm across iterations t. For example, it would be an insightful result to investigate the evolution of the MSE of the AMP algorithm.

Replacing the soft thresholding function $\eta(\,\cdot;\cdot\,)$ in (3.19) with a general denoising function $\eta_t(\,\cdot;\cdot\,)$ that may change from iteration to iteration, the authors in [4] consider the recursion

$$\begin{aligned} \mathbf{x}^{(t+1)} &= \eta_t\Big(\mathbf{A}^{(t)}\mathbf{z}^{(t)} + \mathbf{x}^{(t)}\Big), \\ \mathbf{z}^{(t+1)} &= \mathbf{y} - \mathbf{A}\mathbf{x}^{(t)} + \frac{1}{\delta}\mathbf{z}^{(t)}\Big\langle \eta_t'\Big(\mathbf{A}^{(t)}\mathbf{z}^{(t)} + \mathbf{x}^{(t)}\Big)\Big\rangle, \end{aligned} \tag{3.21}$$

where $\eta_t(\,\cdot\,)$ is applied component-wise.

For a set $\Big\{\{\eta_t\}_{t=1}^{N}, \rho, \delta, \lambda, p_{\mathsf{x}}\Big\}$ the SE is a recursive map $\sigma^{2(t)} \mapsto \Psi(\sigma^{2(t)})$ that describes the evolution (change) of the state $\sigma^{2(t)}$, also called *effective noise variance*, of the AMP algorithm across iterations t. Starting with $\sigma^{2(0)} = \sigma_w^2 + \frac{1}{\delta}\mathbb{E}_{\mathsf{x}}\{\mathsf{x}^2\}$, the SE is given by

$$\begin{aligned} \sigma^{2(t+1)} &= \Psi\Big(\sigma^{2(t)}\Big), \\ \Psi(\sigma^{2(t)}) &= \sigma_w^2 + \frac{1}{\delta}\underbrace{\mathbb{E}_{\mathsf{x},\mathsf{z}}\left\{\Big[\eta_t(\mathsf{x} + \sigma^{(t)}\mathsf{z}) - \mathsf{x}\Big]^2\right\}}_{\widehat{\mathrm{MSE}}(\hat{\mathsf{x}}^{(t)},\mathsf{x})}, \end{aligned} \tag{3.22}$$

where $\mathsf{z} \sim \mathcal{N}(0,1)$.The following theorem is fundamental for characterisation of the AMP algorithm.

Definition 5 [4] *For $k > 1$ we say a function $\phi : \mathbb{R}^m \to \mathbb{R}$ is pseudo-Lipschitz of order k and denote it by $\phi \in PL(k)$ if there exists a constant $L > 0$ such that, for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^m$*

$$|\phi(\mathbf{x}) - \phi(\mathbf{y})| \leq L\Big(1 + \|\mathbf{x}\|^{k-1} + \|\mathbf{y}\|^{k-1}\Big)\|\mathbf{x} - \mathbf{y}\| \,.$$

Theorem 6 [4, Theroem 1] *Let $\{\mathbf{A}(N)\}_{N\geq 0}$ be a sequence of sensing matrices $\mathbf{A} \in \mathbb{R}^{m\times N}$ indexed by N, with i.i.d. entries $a_{ij} \sim \mathcal{N}(0, 1/m)$, and assume $m/N \to \delta \in (0,\infty)$. Consider further a sequence of signals $\{\mathbf{x}(N)\}_{N\geq 0}$ whose empirical distribution converge weakly*[4] *to a probability measure p_{x} on $\mathbb{R}$ with bounded $(2k-2)^{th}$ moment, and assume $\mathbb{E}_{\hat{p}_{\mathsf{x}}(N)}(\mathsf{x}^{2k-2}) \to \mathbb{E}_{p_{\mathsf{x}}(N)}(\mathsf{x}^{2k-2})$ as $N \to \infty$ for some $k \geq 2$. Also assume the noise $\mathbf{w}$ has i.i.d. entries with distribution p_{w} that has bounded $(2k-2)^{th}$ moment. Then, for any pseudo-Lipschitz function $\psi : \mathbb{R}^2 \to \mathbb{R}$ of order k and all $t \geq 0$, almost surely*

$$\lim_{N\to\infty} \frac{1}{N}\sum_{i=1}^{N} \psi(x_i^{(t+1)}, x_i) = \mathbb{E}_{\mathsf{x},\mathsf{z}}\left\{\psi\Big(\eta_t(\mathsf{x} + \sigma^{(t)}\mathsf{z}), \mathsf{x}\Big)\right\}, \tag{3.23}$$

with $\mathsf{x} \sim p_{\mathsf{x}}$ and $\mathsf{z} \sim \mathcal{N}(0,1)$ independent.

This theorem, which was proven in the large system limit for any choice of $(\rho, \delta, \lambda, p(x))$, has a few important consequences for asymptotical prediction of the AMP algorithm. In particular, the evolution of parameters of the algorithm; prediction of the MSE of the algorithm; phase transition; and the decoupling principle will be explained in the following subsection.

Evolution of AMP parameters

If we choose $\psi(x, y) = (x-y)^2$, we can estimate the true MSE of the AMP algorithm at iteration t, given by $\mathrm{MSE}^{(t)}(\hat{\mathbf{x}}^{(t)}, \mathbf{x}) = \frac{1}{N}\|\hat{\mathbf{x}}^{(t)} - \mathbf{x}\|_2^2$, by

$$\mathrm{MSE}_{\mathrm{SE}}^{(t)} = \mathbb{E}_{\mathsf{x},\mathsf{z}}\left\{(\hat{\mathsf{x}} - \mathsf{x})^2\right\} = \mathbb{E}_{\mathsf{x},\mathsf{z}}\left\{\Big(\eta_t(\mathsf{x} + \sigma^{(t)}\mathsf{z}) - \mathsf{x}\Big)^2\right\}, \tag{3.24}$$

for any choice of $\eta_t(\cdot\,;\cdot)$ (under certain technical conditions). Compared to all iterative thresholding algorithms, this property is unique to the AMP algorithm. Additionally, if we use the denoiser function given in (3.19), one could track the evolution of the soft threshold parameter $\gamma^{(t)}$. Figure 3.3 shows an example of both SE prediction as

[4]Details about weak convergence can be found in [10].

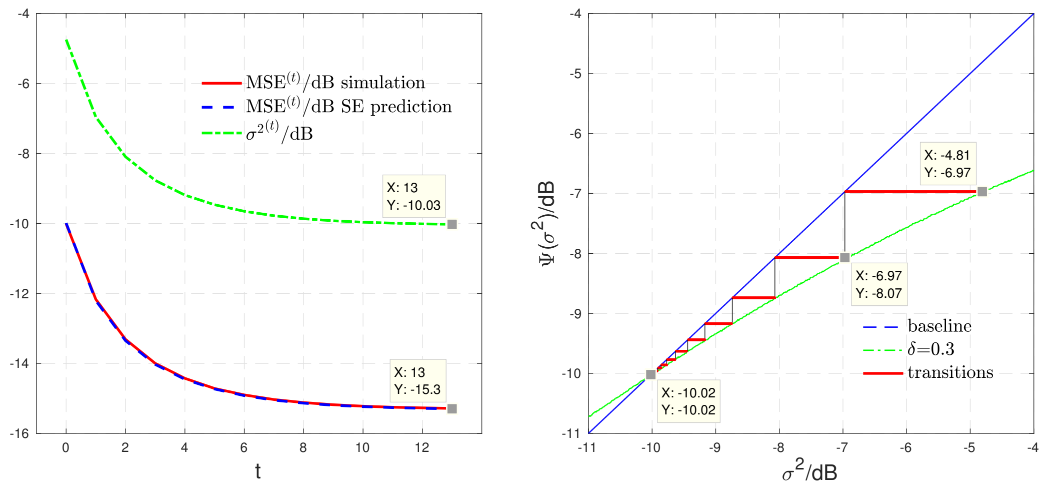


Fig. 3.4 $\mathrm{MSE}^{(t)}$ and $\sigma^{2(t)}$ against iteration t of the AMP algorithm (left) and the prediction of the effective noise transitions (right). Parameters: $N = 1000$, $\lambda = 2.0$, $\delta = 0.3$, $\epsilon = 0.1$, $\sigma^2_{\mathrm{w}} = -30$dB.

well simulation results (averaged over 100 simulations) of $\mathrm{MSE}^{(t)}$ and $\gamma^{(t)}$ across t for a specific set of parameters given below. Even for not so large N we can observe an excellent match between simulation results and SE prediction of the corresponding terms.

MSE and fixed points

Fixed points of the state evolution are states for which $\sigma^{2(t+1)} = \sigma^{2(t)}$, and consequently $\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)}$. We can differentiate between two types of fixed points of the SE iteration, namely: stable fixed points and unstable fixed points. A stable fixed point is a fixed point to which the system converges after an arbitrary small perturbation. Otherwise, the fixed point is unstable. Let us take a better look at Figure 3.4, where we show $\mathrm{MSE}^{(t)}$ and $\sigma^{2(t)}$ against iteration t of the AMP algorithm on the left, and the prediction of the effective noise transitions on the right. The first stable fixed point that appears from the right side (higher MSE) is the one that defines $\mathrm{MSE}^{(\infty)}$. According to (3.22), at $t = 0$ we start with $\sigma^{2(0)} = \sigma^2_w + \frac{1}{\delta}\,\mathbb{E}\{\mathsf{x}^2\} \approx -4.78$ dB, and at this point the value of the SE prediction is $\sigma^{2(1)} \approx -6.97$ dB. This point is highlighted in Figure 3.4 (right), by the far right marker. Moving horizontally from this point to the baseline and then vertically to the SE curve we find the SE prediction of $\sigma^{2(1)}$. Continuing this process, we will end up in the point where function Ψ and the baseline cross. The crossing point corresponds to the fixed point of the algorithm, and asymptotically the algorithm is producing $\mathrm{MSE}^{(\infty)} = \delta(\sigma^{2(\infty)} - \sigma^2_w)$. In this specific case, $\mathrm{MSE}^{(\infty)} \approx -15.3$ dB and

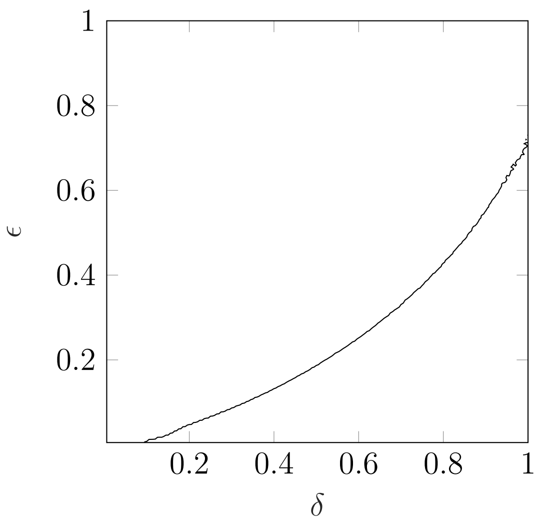


Fig. 3.5 PT curve of the AMP algorithm $\epsilon_{\text{AMP}}(\delta)$ for hard constraints predicted by the SE. Maximum number of iterations is $t_{\max} = 200$. Additionally, the recursion is terminated if $\text{MSE}_{\text{SE}}^{(t)} \le -90\,\text{dB}$.

$\sigma^{2(\infty)} \approx -10.02$ dB, where the difference of -5.28 dB between the two is mainly due to the $10\log_{10}\delta \approx -5.22$ dB.

Phase transition

All pairs of measurement ratio and non-zero probability $(\delta, \epsilon) \in [0,1]^2$ constitute the so-called phase space[5]. For BP and BPDN problems given in (2.5) and (2.10), provided that the elements of $\mathbf{A}$ are i.i.d. Gaussian random variables, asymptotically the phase space partitions in the two regions [36]. The curved line separating those two regions $\epsilon_{l1}(\delta)$ (alternatively $\rho_{l1}(\delta)$) is called PT curve (or simply phase transition), and the regions indicate if the reconstruction is successful or not: below the PT curve *with high probability* (whp) the sparsity ratio is sufficient to allow for successful reconstruction, while above the PT curve whp the reconstruction is unsuccessful.

For the AMP algorithm on the other hand, for any pair (δ, ϵ), in the large system limit, we can use fixed points of the SE to determine if the algorithm converges to the true solution or not. If $\sigma^{2(t)} = 0$ is the only stable fixed point, then $\hat{\mathbf{x}}^{(t)} \to \mathbf{x}$ as $t \to \infty$ since

$$\text{MSE}^{(t)} = \delta\sigma^{2(t)} \to 0 \quad \text{as} \quad t \to \infty.$$

[5]Alternatively, one can consider phase space as all pairs of measurement ratio and normalized sparsity $(\delta, \rho) \in [0,1]^2$. In this case the phase transition curve is denoted by $\rho(\delta)$.

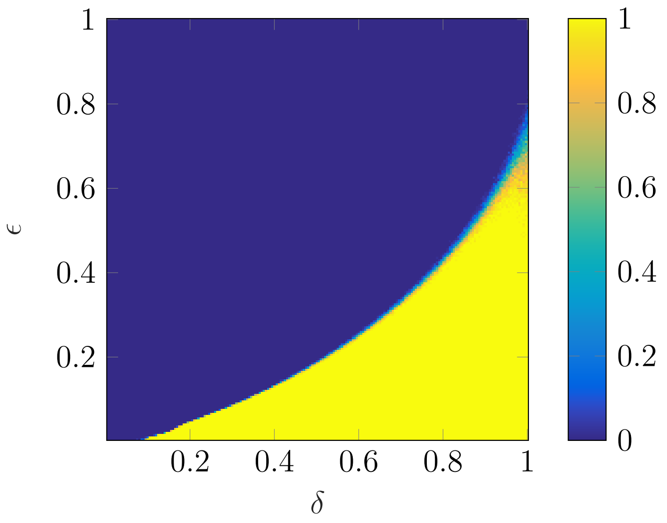


Fig. 3.6 Success rate of the AMP algorithm for hard constraints predicted by the SE. Maximum number of iterations is $t_{\max} = 200$. Additionally, the recursion is terminated if $\mathrm{MSE}_{\mathrm{SE}}^{(t)} \le -90\,\mathrm{dB}$.

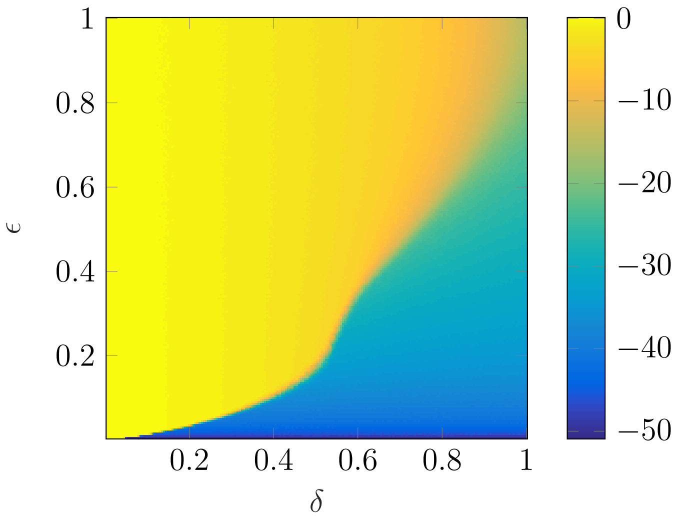


Fig. 3.7 MSE of the BAMP algorithm predicted by the SE for the problem described in Examples 1 in 3.2.2. Maximum number of iterations is $t_{\max} = 100$. Additionally, the recursion is terminated if $\mathrm{MSE}_{\mathrm{SE}}^{(t)} \le -60\,\mathrm{dB}$.

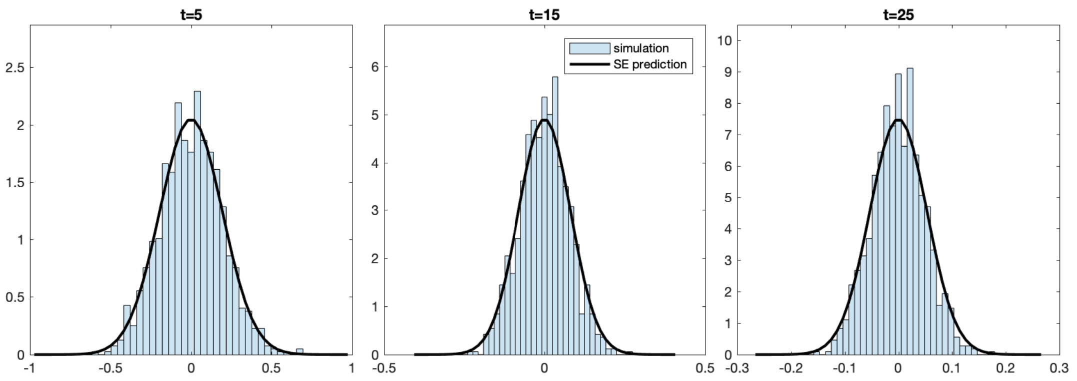
Figure 3.5 shows the PT curve $\epsilon_{\text{AMP}}(\delta)$ and Figure 3.6 shows success rate of the AMP algorithm for hard constraints given in (3.16). To get the results shown in the figures, we run the SE recursion given in (3.22) for every pair $(\delta, \epsilon) \in [0,1]^2$ 50 times, and average the end $\text{MSE}_{\text{SE}}^{(t)}$. The recursion is stopped when the maximum number of iterations $t_{\max}$ is reached, or if $\text{MSE}_{\text{SE}}^{(t)}$ is less than -90 dB. We say that a reconstruction $\hat{\mathbf{x}}$ in Algorithm 4 is successful (i.e., the algorithm converges to the true solution) if $\text{MSE}_{\text{SE}}^{(t)} \leq -60$. In Figure 3.6 we can se a clear transition between the region of the (δ, ϵ) space where the reconstruction is successful and the region where it is not. The line separating those regions, i.e., the phase transition curve, is shown in Figure 3.5.

Alternatively, one could find empirical PT curve [32, 36] by running the AMP algorithm a number of times for every pair (δ, ϵ), each time with an independent realization of **x** and **A**, and calculating the average number of successful recoveries. However, this approach is much slower than running the SE recursion.

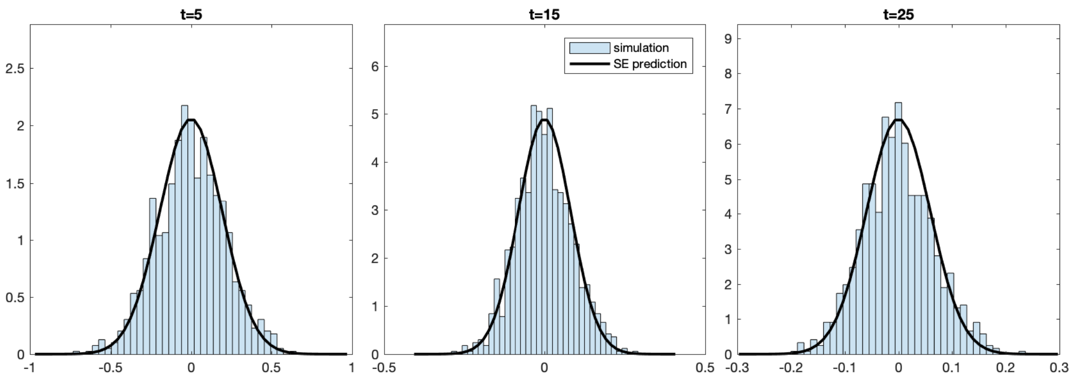
In Figure 3.7, one can show MSE of the AMP based algorithms as a function of (δ, ϵ) for problem defined. Here, we use Bayesian-optimal denoiser function of the BAMP algorithm, as will be later described in 3.2.

Decoupling principle

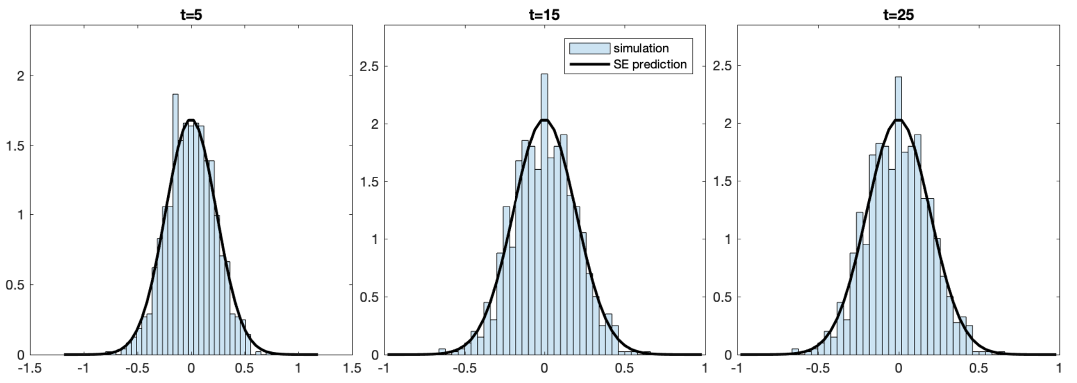
As a consequence of Theorem 6, any typical subsets of the entries of $\mathbf{x}^{(t)}$ are asymptotically independent [4]. Additionally, the argument of the soft threshold function $\mathbf{A}^{(t)}\mathbf{z}^{(t)}+\mathbf{x}^{(t)}$ "behaves like" an observation of true $\mathbf{x}$ corrupted with zero mean Gaussian noise with variance $\sigma^{2(t)}$, i.e., $\mathbf{A}^{(t)}\mathbf{z}^{(t)} + \mathbf{x}^{(t)} = \mathbf{x} + \mathbf{v}^{(t)}$, where $\mathbf{v}^{(t)} \sim \mathcal{N}(0, \sigma^{2(t)}\mathbf{I}_m)$. These properties of the AMP algorithm are demonstrated in Figure 3.8, where we show the empirical distribution of the effective noise as well as zero-mean Gaussian distribution whose variance is predicted by SE, for different levels of AWGN. We can observe a very good match between the two distributions. Since the presence of the Onsager term in (3.22) is the only difference between thresholding algorithms and AMP, one can argue that this term is responsible for the decoupling principle.



(a)



(b)



(c)

Fig. 3.8 Normalized histogram (blue) and SE prediction of the distribution (black) of the effective noise at different iterations of the AMP algorithm: a) $\sigma_{\mathrm{w}}^2 = -\infty\,\mathrm{dB}$, b) $\sigma_{\mathrm{w}}^2 = -40\,\mathrm{dB}$, c) $\sigma_{\mathrm{w}}^2 = -20\,\mathrm{dB}$ (bottom). Parameters: $N = 1000$, $\lambda = 2.0$, $\delta = 0.5$, $\epsilon = 0.1$.

3.2 Bayesian Approximate Message Passing

Going back to (3.18) we see that by choosing λ, we implicitly assume a certain prior from the family of Laplacian distributions for $\mathbf{x}$. There is nothing stopping us from choosing some other prior $p_{\mathsf{x}}(\mathbf{x};\alpha)$, parametrised by a set of parameters α, that promotes prior knowledge about $\mathbf{x}$. For example, if we want to promote sparsity, we could use i.i.d. zero-mean *Bernoulli-Gauss* (BG) mixture prior given by

$$p_{\mathsf{x}_i}(x;\epsilon,\sigma_x^2) = (1-\epsilon)\delta(x) + \epsilon\mathcal{N}(x;0,\sigma_x^2), \tag{3.25}$$

where ϵ is probability of nonzero value, and σ_x^2 is the power of the Gaussian component. In this case, the corresponding conditional distribution of x given the observation vector $\mathbf{y}$ is

$$p_{\mathsf{x}|\mathsf{y}}(\mathbf{x}|\mathbf{y};\alpha) = \frac{1}{Z}\prod_{i=1}^{N}\Big((1-\epsilon)\delta(x_i) + \epsilon\mathcal{N}(x_i;0,\sigma_x^2)\Big)\prod_{a=1}^{m}\exp\Big(-\frac{\beta}{2}(y_a-(\mathbf{A}\mathbf{x})_a^2\Big). \tag{3.26}$$

Repeating the steps for deriving the AMP algorithm, one can see that nothing changes, except that now the messages from the variable nodes to the factor nodes $m_{i\to a}^{(t+1)}(x_i)$ can be approximated by the product of a Gaussian and here assumed prior. As a consequence, the mean and the variance of $m_{i\to a}^{(t+1)}(x_i)$, given in (3.11), take different form, namely the mean and the variance of the above mentioned product of distributions. The resulting algorithm, called BAMP is given by Algorithm 5. It is the implementation of the following iteration

$$\begin{aligned}
\mathbf{u}^{(t)} &= \mathbf{A}^{(t)}\mathbf{z}^{(t-1)} + \mathbf{x}^{(t-1)},\\
\mathbf{x}^{(t)} &= F\Big(\mathbf{u}^{(t)};c^{(t)},\alpha\Big),\\
\mathbf{z}^{(t)} &= \mathbf{y} - \mathbf{A}\mathbf{x}^{(t)} + \frac{1}{\delta}\mathbf{z}^{(t)}\Big\langle F'\Big(\mathbf{u}^{(t)};c^{(t)},\alpha\Big)\Big\rangle,\\
c^{(t+1)} &= \sigma_w^2 + \frac{1}{\delta}\Big\langle G\Big(\mathbf{u}^{(t)};c^{(t)},\alpha\Big)\Big\rangle,
\end{aligned} \tag{3.27}$$

where functions $F(\cdot)$, $F'(\cdot)$, and $G(\cdot)$ are applied component-wise and are given by

$$\begin{aligned}
F(u_i;c,\alpha) &= \mathbb{E}_{\mathsf{x}}\{\mathsf{x}|\mathsf{u}=u_i\,;\,c,\alpha\},\\
F'(u_i;c,\alpha) &= \frac{d}{du_i}F(u_i\,;\,c,\alpha),\\
G(u_i;c,\alpha) &= \mathrm{var}_{\mathsf{x}}\{\mathsf{x}|\mathsf{u}=u_i\,;\,c,\alpha\}.
\end{aligned} \tag{3.28}$$

Comparing the AMP algorithm given by (3.19), and BAMP given by (3.27), we can see that the only difference is in the denoiser function, that uses different prior knowledge about x. In (3.28), the argument u_i of the denoiser function $F(\cdot)$ acts as an observation of x_i corrupted with zero-mean Gaussian noise with variance c, i.e.,

Algorithm 5 Bayesian Approximate Message Passing Algorithm

```
Input: measurement matrix $\mathbf{A}$, measurement vector $\mathbf{y}$, noise level $\sigma_w^2$ (optional), prior
    distribution $p(x_i|\alpha)$, stopping threshold $\varepsilon_{\text{stop}}$ or $t_{\max}$
Initialization: $t = 0$, $\hat{\mathbf{x}}^{(0)} = \mathbf{0}_{N\times 1}$, $\mathbf{z}^{(0)} = \mathbf{y}$, $c^{(1)} = \sigma_w^2 + \frac{1}{\delta}\epsilon\sigma_x^2$
do:
    $t = t + 1$                                                        ▷ increment iteration counter
    $\mathbf{u}^{(t)} = \mathbf{A}^{(t)}\mathbf{z}^{(t-1)} + \mathbf{x}^{(t-1)}$                                  ▷ decouple measurements
    $\mathbf{x}^{(t)} = F\left(\mathbf{u}^{(t)}; c^{(t)}, \sigma_x^2\right)$                     ▷ denoise, i.e., apply thresholding
    $\mathbf{z}^{(t)} = \mathbf{y} - \mathbf{A}\mathbf{x}^{(t)} + \frac{1}{\delta}\mathbf{z}^{(t-1)}\langle F'\left(\mathbf{u}^{(t)}; c^{(t)}, \sigma_x^2\right)\rangle$    ▷ calculate residual
    $c^{(t+1)} = \sigma_w^2 + \frac{1}{\delta}\langle G\left(\mathbf{u}^{(t)}; c^{(t)}, \sigma_x^2\right)\rangle$            ▷ estimate effective noise variance
while $t \le t_{\max}$ or $\|\hat{\mathbf{x}}^{(t)} - \hat{\mathbf{x}}^{(t-1)}\|_2 \ge \varepsilon_{\text{stop}} \|\hat{\mathbf{x}}^{(t)}\|_2$
Output: $\hat{\mathbf{x}} = \mathbf{x}^{(t)}$
```

$u_i \sim \mathcal{N}(x_i, c)$. Therefore, the conditional distribution $p(x_i|u_i\,;\,c,\alpha)$ can be written as

$$p(x_i|u_i\,;\,c,\alpha) = \frac{1}{p(u_i)}\,p(u_i|x_i\,;c)\,p(x_i\,;\alpha) = \frac{1}{\kappa(u_i;c)}\,g(u_i - x_i; 0, c)\,p(x_i\,;\alpha), \tag{3.29}$$

where $\kappa(u_i;c)$ is a normalizing constant, and $g(\cdot;0,c)$ is zero-mean Gaussian distribution with variance c. Depending on the specific prior $p(x_i)$, one could get closed-form expressions for the functions in (3.19), or use numerical methods to get approximations of those functions.

As indicated in the input part of the Algorithm 5, we need to provide the prior distribution $p(x_i\,;\alpha)$. Therefore, one can argue that this is a disadvantage of the BAMP algorithm, since the prior is not usually known. One way out of this problem is to model the signal as a general Gaussian-Bernoulli mixture, and learn the parameters of the distribution using an empirical-Bayesian technique while producing ever better MSE estimate in each iteration of the algorithm [82]. Alternatively, if the prior can be parametrised, one can use low complexity scheme based on Method-of-Moments to estimate the parameters of the prior distribution during the iterations of the BAMP algorithm [40].

Before giving some results for specific priors of interest, we show alternative formulation of the denoiser functions $F(\cdot)$, $F'(\cdot)$, and $G(\cdot)$.

Alternative formulation of the denoiser functions

Using integration by parts and a few mathematical tricks, one can rewrite (3.28) as

$$\begin{aligned} F(u_j; c, \alpha) &= u_j - \sqrt{\frac{2c}{\pi}}\,\frac{I_0(u_j; c, \alpha)}{I_2(u_j; c, \alpha)}, \\ G(u_j; c, \alpha) &= c + F(u_j; c, \alpha)\Big(u_j - F(u_j; c, \alpha)\Big) - \sqrt{\frac{2c}{\pi}}\,\frac{I_1(u_j; c, \alpha)}{I_2(u_j; c, \alpha)}, \\ F'(u_j; c, \alpha) &= \frac{1}{c}\, G(u_j; c, \alpha), \end{aligned} \tag{3.30}$$

where the integrals $I_0(u_j; c, \alpha)$, $I_1(u_j; c, \alpha)$, and $I_2(u_j; c, \alpha)$ are defined as

$$\begin{aligned} I_0(u_j; c, \alpha) &= \int_{-\infty}^{+\infty} e^{-\frac{(x_j - u_j)^2}{2c}}\, p'_{\mathsf{x}_j}(x_j; \alpha)\, dx_j, \\ I_1(u_j; c, \alpha) &= \int_{-\infty}^{+\infty} x_j\, e^{-\frac{(x_j - u_j)^2}{2c}}\, p'_{\mathsf{x}_j}(x_j; \alpha)\, dx_j, \\ I_2(u_j; c, \alpha) &= \int_{-\infty}^{+\infty} \operatorname{erf}\left(\frac{x_j - u_j}{\sqrt{2c}}\right) p'_{\mathsf{x}_j}(x_j; \alpha)\, dx_j. \end{aligned} \tag{3.31}$$

3.2.1 Mix of n non-zero mean Gaussians

I assume that the prior consists of a weighted (weighting factors γ_k) average of n-Gaussian distributions, each with possibly different mean μ_k and different variance σ_k^2. By choosing appropriate values for the means and variances, many practically interesting priors can be modeled or approximated, even including discrete distributions (by picking very small but still non-zero variance).

Hence, the prior considered is given as

$$p(x_j\,;\,\alpha) = \sum_{k=1}^{n} \gamma_k\, p_k(x_j) = \sum_{k=1}^{n} \gamma_k\, g(x_j; \mu_k, \sigma_k^2), \tag{3.32}$$

where $\sum_{k=1}^{n} \gamma_k = 1$, and $0 \leq \gamma_k \leq 1 \; \forall k$. Inserting the prior from (3.32) in to (3.31), we have

$$\begin{aligned}
I_0(u_j; c, \alpha) &= \sum_{k=1}^{n} \gamma_k \underbrace{\int_{-\infty}^{+\infty} e^{-\frac{\left(x_j - u_j\right)^2}{2c}} \; p_k'(x_j)\, dx_j}_{I_{0k}(u_j;c)}, \\
I_1(u_j; c, \alpha) &= \sum_{k=1}^{n} \gamma_k \underbrace{\int_{-\infty}^{+\infty} x_j \; e^{-\frac{\left(x_j - u_j\right)^2}{2c}} \; p_k'(x_j)\, dx_j}_{I_{1k}(u_j;c)}, \\
I_2(u_j; c, \alpha) &= \sum_{k=1}^{n} \gamma_k \underbrace{\int_{-\infty}^{+\infty} \operatorname{erf}\left(\frac{x_j - u_j}{\sqrt{2c}}\right) p_k'(x_j)\, dx_j}_{I_{2k}(u_j;c)} .
\end{aligned} \tag{3.33}$$

Here, I omit a detailed calculation of the integrals $I_{0k}(u_j; c)$, $I_{1k}(u_j; c)$, and $I_{2k}(u_j; c)$ which is given in the Appendix A.2, and instead give final expressions for the integrals, namely

$$\begin{aligned}
I_0(u_j; c, \alpha) &= -\sqrt{2\pi c} \sum_{k=1}^{n} \gamma_k \, g(u_j; \mu_k, \sigma_{k,c}^2) \frac{u_j - \mu_k}{\sigma_{k,c}^2}, \\
I_1(u_j; c, \alpha) &= -\sqrt{2\pi c} \sum_{k=1}^{n} \gamma_k \, g(u_j; \mu_k, \sigma_{k,c}^2) \left[\frac{\sigma_k^2 (u_j - \mu_k)^2}{\sigma_{k,c}^4} + \frac{c + \mu_k (u_j - \mu_k)}{\sigma_{k,c}^2} \right], \\
I_2(u_j; c, \alpha) &= -\sqrt{2\pi c} \sum_{k=1}^{n} \gamma_k \, g(u_j; \mu_k, \sigma_{k,c}^2) \sqrt{\frac{2}{\pi c}},
\end{aligned} \tag{3.34}$$

where $\sigma_{k,c}^2 = \sigma_k^2 + c$.

3.2.2 General discrete prior mixed with a zero-mean Gaussian prior

We assume that the source takes a discrete value with probability $1 - \epsilon$ and a continuous value with probability ϵ, i.e.,

$$p_{\mathsf{x}_j}(x_j, \alpha) = (1 - \epsilon)\, p_{\mathsf{x}_j}^d(x_j, \alpha) + \epsilon\, p_{\mathsf{x}_j}^c(x_j, \alpha). \tag{3.35}$$

Moreover, the discrete values come from the set $\{b_m\}_{m=1}^{M}$, where the probability of the value b_m is ϵ_m^d, i.e.,

$$p_{\mathsf{x}_j}^d(x_j, \alpha) = \sum_{m=1}^{M} \epsilon_m^d \delta(x_j - b_m) \quad \text{with} \quad \sum_{m=1}^{M} \epsilon_m^d = 1, \tag{3.36}$$

while the continuous part $p^c_{\mathsf{x}_j}(x_j,\alpha)$ of the distribution $p_{\mathsf{x}_j}(x_j,\alpha)$ is a zero-mean Gaussian distribution with variance σ_x^2. Hence, the prior becomes

$$p_{\mathsf{x}_j}(x_j,\alpha) = (1-\epsilon)\sum_{m=1}^{M}\epsilon^d_m\delta(x_j-b_m) + \epsilon\, g(x_j;0,\sigma_x^2). \tag{3.37}$$

Plugging in the mixture prior given in (3.37) to (3.28) we can obtain closed-form expressions for the denoiser functions of the BAMP algorithm. Alternatively, since we model occurrence of zero value with delta distribution $\delta(x)$, and $\delta(x)$ can be considered as a limit $\lim_{\sigma\to 0} g(x;0,\sigma^2)$, we can use the results given in (3.34) to obtain closed-form expressions for the denoiser functions of the BAMP algorithm. We omit this derivation, which is given in Appendix A.1 and instead give final expressions for the integrals $I_{0k}(u_j;c)$, $I_{1k}(u_j;c)$, and $I_{2k}(u_j;c)$, namely

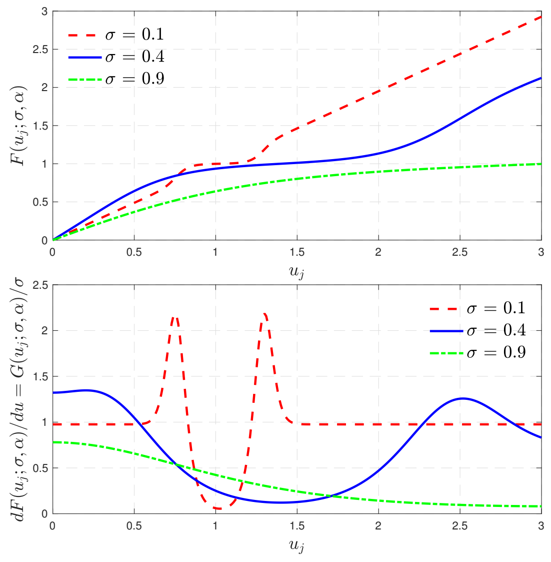
$$\begin{aligned}
I_0(u_j;c) &= -\frac{1-\epsilon}{c}\sum_{m=1}^{M}\epsilon_m(u_j-b_m)\,e^{-\frac{(u_j-b_m)^2}{2c}} - \epsilon\frac{\sqrt{c}}{\sigma^3_{x,c}}\,u_j\,e^{-\frac{u_j^2}{2\sigma^2_{x,c}}},\\
I_1(u_j;c) &= -\frac{1-\epsilon}{c}\sum_{m=1}^{M}\epsilon_m\,(c+b_mu_j-b_m^2)\,e^{-\frac{(u_j-b_m)^2}{2c}} - \epsilon\frac{\sqrt{c}}{\sigma^3_{x,c}}\Big(c+\frac{\sigma^2}{\sigma^2_{x,c}}u_j^2\Big)e^{-\frac{u_j^2}{2\sigma^2_{x,c}}},\\
I_2(u_j;c) &= -(1-\epsilon)\sqrt{\frac{2}{\pi c}}\sum_{m=1}^{M}\epsilon_m\,e^{-\frac{(u_j-b_m)^2}{2c}} - \epsilon\sqrt{\frac{2}{\pi\sigma^2_{x,c}}}e^{-\frac{u_j^2}{2\sigma^2_{x,c}}},
\end{aligned} \tag{3.38}$$

where $\sigma^2_{x,c}=\sigma^2_x+c$.

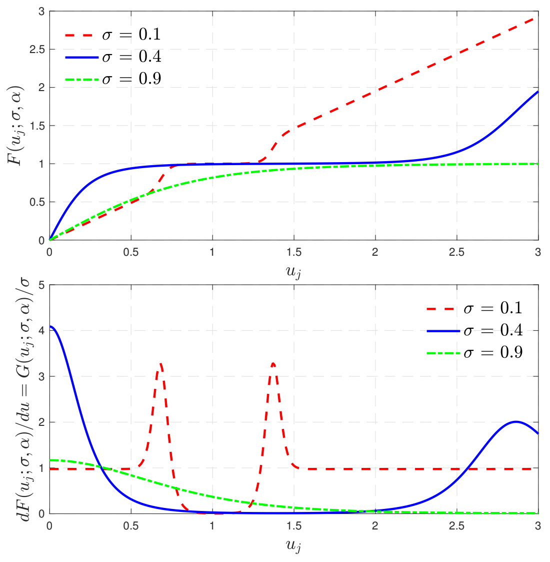
Numerical examples

1. *Example 1: Discrete point at* ± 1. Here in (3.37) we choose $M=2$, $\epsilon^d_1=\epsilon^d_2=\frac{1}{2}$, and $b_1=1=-b_2$, while the variance of the zero-mean Gaussian continuous component is $\sigma^2_x=0.4$. The probabilities for the continuous components are $\epsilon=0.4$ and $\epsilon=0.05$.

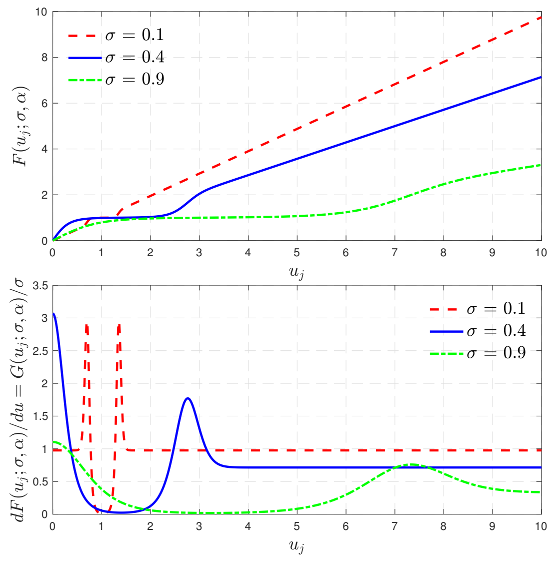
   The results for the estimator functions $F(u_j;c,\alpha)$ and its derivative are given in Figs. 3.9a and 3.9b. Since the results are symmetric with respect to the y-axis, we only show results for positive u_j. In Fig. 3.9c a Gaussian prior with larger variance than in Figs. 3.9a and 3.9b is used. In all three figures we can observe a plateau forming around the discrete points, where the plateau becomes wider with increasing effective noise standard deviation σ and probability of discrete points $1-\epsilon$. For the inputs close to the discrete points, the denoiser



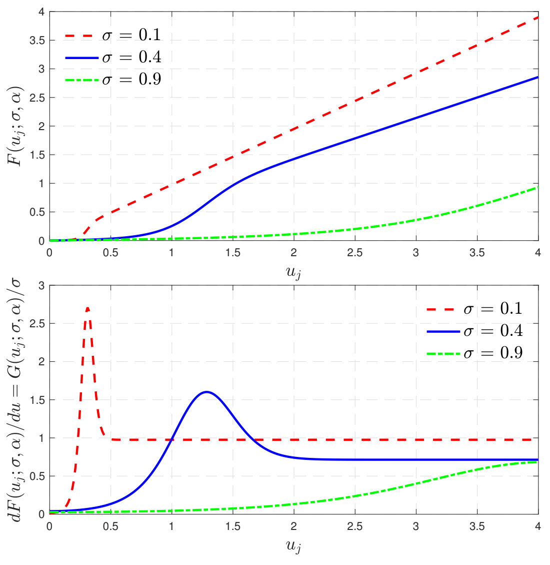
(a) Discrete points at $\{-1,+1\}$; probability for a continuous point $\epsilon = 0.4$, with $\mu_x = 0$, and $\sigma_x^2 = 0.4$.



(b) Discrete points at $\{-1,+1\}$; probability for a continuous point $\epsilon = 0.05$, with $\mu_x = 0$, and $\sigma_x^2 = 0.4$.



(c) Discrete points at $\{-1,+1\}$; probability for a continuous point $\epsilon = 0.1$, with $\mu_x = 0$, and $\sigma_x^2 = 1.2$.



(d) Discrete point at 0; probability for a continuous point $\epsilon = 0.1$, with $\mu_x = 0$, and $\sigma_x^2 = 1.2$.

Fig. 3.9 Function plots of $F(u_j;c)$ and $F'(u_j;c) = G(u_j;c)/c$ for different Gauss-Bernoulli mixtures and different effective noise standard-deviations σ.

behaves similar to a quantizer, since those inputs are mapped to values almost indistinguishable from the discrete points.

2. *Example 2: Sparse case, i.e., discrete point at 0.* In Fig. 3.9d a continuous Gaussian prior with variance $\sigma_x^2 = 1.2$ is used, but now the discrete part is located at zero with probability $1 - \epsilon = 0.9$.

   Note that as long as a reasonably large variance for the continuous Gaussian prior is chosen, there are no numerical problems, as there are no gaps in the support and the prior distribution has significant non-zero values in a reasonable range of input values u to the estimator.

   Also note that the curve for $c = 0.1$ (low noise case) in Fig. 3.9d looks rather similar to a hard thresholding function. However, as the noise increases, the typical effect of MMSE estimators appears: the output magnitudes tend to smaller values, due to the ambiguity caused by the noise.

3.3 Generalized Approximate Message Passing

In Section 3.2 we saw that there is nothing special about the Laplacian prior for AMP to be derived, and later showed BAMP algorithm for a general prior distribution (also referred to as the input channel). Similarly, one can argue that there is nothing special about the Gaussian noise model that describes a distortion of the vector of linear mixtures $\mathbf{z} = \mathbf{A}\mathbf{x}$. This approach leads to the GAMP algorithm, where one considers a general distribution describing the component-wise distortion of the linear mixtures, i.e., a general output channel given in terms of a conditional distribution $p(y_i|z_i)$. This includes for example, component-wise nonlinear distortions of the vector $\mathbf{z}$, that appears when we quantize CS measurements. By approximating sum-product loopy belief propagation, the sum-product GAMP algorithm approximates the computationally intractable high-dimensional integration involved with calculating $\mathbb{E}\{\mathbf{x}|\mathbf{y}\}$ with a highly efficient iterative procedure. Similarly, the max-sum version of GAMP approximates the computationally intractable calculation of $\arg\max_{\mathbf{x}} p(\mathbf{x}|\mathbf{y})$ when one seeks for the MAP estimate of $\mathbf{x}$ [71]. In the rest of the text we will focus on the sum-product GAMP for the MMSE estimate, and refer to it simply as the GAMP algorithm.

The generalized estimation problem with linear mixing is shown in Figure 3.10. Here, as in BAMP, vector $\mathbf{x}$ is distributed according to $p(\mathbf{x}; \alpha)$, where α contains the distribution parameters. This vector is multiplied (i.e., sampled) with measurement

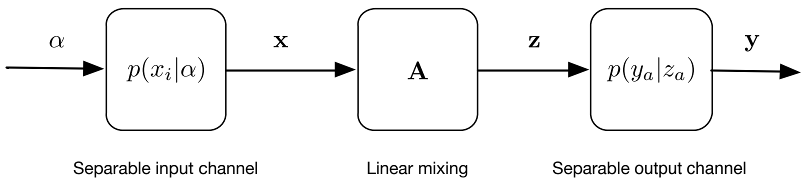


Fig. 3.10 General estimation problem: Vector $\mathbf{x}$, distributed according to $p(\mathbf{x};\alpha)$ is multiplied with measurement matrix $\mathbf{A}$, resulting in $\mathbf{z}$. Each measurement y_i is generated randomly from z_i according to $p(y_i|z_i)$. Find $\mathbf{x}$ given $\mathbf{y}$.

matrix $\mathbf{A}$, and the resulting vector is denoted by $\mathbf{z}$. Finally, each measurement y_i is generated randomly from z_i according to $p(y_i|z_i)$.

The problem is to get MMSE or MAP estimate $\mathbf{x}$ given the measurement vector $\mathbf{y}$, where the corresponding conditional distribution can be written as

$$p_{\mathsf{x}|\mathsf{y}}(\mathbf{x}|\mathbf{y};\alpha) = \frac{1}{Z}\prod_{i=1}^{N} p(x_i;\alpha)\prod_{a=1}^{m} p(y_a|z_a). \tag{3.39}$$

3.3.1 Overview of the derivation of the GAMP algorithm

Approximation of the message passing algorithm for the conditional distribution given in (3.39) is similar to derivation of AMP and BAMP, but somewhat more mathematically involved. A detailed derivation of the GAMP algorithm is provided in [71], with more insights available in [79]. Compared to the derivation of the AMP algorithm, one difference is that now the messages of the sum-product algorithm are log-pdfs. Following the update rules for variable nodes (1) and for factor nodes (2) on page 26, the log-pdf messages $m_{i\to a}^{(t+1)}(x_i)$ and $\hat{m}_{a\to i}^{(t)}(x_i)$ can be written as

$$\begin{aligned} m_{i\to a}^{(t+1)}(x_i) &= \log \frac{1}{Z_1} p(x_i;\alpha)\prod_{b\neq a}\exp\left(\hat{m}_{b\to i}^{(t)}(x_i)\right) \\ &= \text{const} + \log p(x_i;\alpha) + \sum_{b\neq a}\hat{m}_{b\to i}^{(t)}(x_i), \end{aligned} \tag{3.40}$$

$$\begin{aligned} \hat{m}_{a\to i}^{(t)}(x_i) &= \log \frac{1}{Z_2}\int p(y_a|z_a)\prod_{j\neq i}\exp\left(m_{j\to a}^{(t)}(x_j)\right) d\mathbf{x}_{\sim i} \\ &= \text{const} + \log \int p(y_a|z_a)\prod_{j\neq i}\exp\left(m_{j\to a}^{(t)}(x_j)\right) d\mathbf{x}_{\sim i}, \end{aligned} \tag{3.41}$$

where Z_1 and Z_2 are normalizing constants.

Approximation of $\hat{m}^{(t)}_{a\to i}(x_i)$: To approximate the message $\hat{m}^{(t)}_{a\to i}(x_i)$, given in (3.41), we follow the same reasoning as in (3.10) to obtain

$$\begin{aligned}
\hat{m}^{(t)}_{a\to i}(x_i) &= \text{const} + \log \int p(y_a|z_a;x_i) \prod_{j\neq i} \exp\left(m^{(t)}_{j\to a}(x_j)\right) d\mathbf{x}_{\sim i} \\
&= \text{const} + \log \mathbb{E}_{\mathbf{x}_{\sim i}} \left\{ p(y_a|z_a;x_i) \right\} \\
&\approx \text{const} + \underbrace{\log \int p(y_a|z_a;x_i)\, \mathcal{N}\Big(a_{ai}x_i + \sum_{j\neq i} a_{aj}\hat{x}^{(t)}_{ja},\, \sum_{j\neq i} a^2_{aj} v^{x\,(t)}_{ja}\Big) dz_a}_{H(y_a,\, a_{ai}x_i + \sum_{j\neq i} a_{aj}\hat{x}^{(t)}_{ja},\, \sum_{j\neq i} a^2_{aj} v^{x\,(t)}_{ja})} \qquad (3.42) \\
&= \text{const} + H\Big(y_a,\, a_{ai}x_i + \sum_{j\neq i} a_{aj}\hat{x}^{(t)}_{ja},\, \sum_{j\neq i} a^2_{aj} v^{x\,(t)}_{ja}\Big) \\
&= \text{const} + H\Big(y_a,\, a_{ai}(x_i - \hat{x}^{(t)}_{ia}) + \sum_{j} a_{aj}\hat{x}^{(t)}_{ja},\, \sum_{j} a^2_{aj} v^{x\,(t)}_{ja} - a^2_{ai} v^{x\,(t)}_{ia}\Big).
\end{aligned}$$

Let us denote the sums over j in (3.42) with

$$\begin{aligned}
\hat{p}^{(t)}_a &= \sum_j a_{aj}\hat{x}^{(t)}_{ja}, \\
v^{p(t)}_a &= \sum_j a^2_{aj} v^{x\,(t)}_{ja},
\end{aligned} \qquad (3.43)$$

and the posterior mean and variance of variable x_j at iteration t with

$$\begin{aligned}
\hat{x}^{(t)}_i &= \mathbb{E}\{x_i \,|\, m^{(t)}_i(\cdot)\}, \\
v^{x(t)}_i &= \operatorname{var}\{x_i \,|\, m^{(t)}_i(\cdot)\},
\end{aligned} \qquad (3.44)$$

where

$$m^{(t)}_i(x_i) = \text{const} + \log p(x_i;\alpha) + \sum_b \hat{m}^{(t)}_{b\to i}(x_i). \qquad (3.45)$$

Neglecting terms that are $O(1/N)$ in (3.46), i.e.,

$$\begin{aligned}
H(\cdot,\cdot,\cdot) &= H\Big(y_a,\, a_{ai}(x_i - \hat{x}^{(t)}_{ia}) + \sum_j a_{aj}\hat{x}^{(t)}_{ja},\, \sum_j a^2_{aj} v^{x\,(t)}_{ja} - a^2_{ai} v^{x\,(t)}_{ia}\Big) \\
&= H\Big(y_a,\, a_{ai}(x_i - \hat{x}^{(t)}_{i}) + \underbrace{a_{ai}(\hat{x}^{(t)}_{i} - \hat{x}^{(t)}_{ia})}_{O(1/N)} + \hat{p}^{(t)}_a,\, v^{p(t)}_a - \underbrace{a^2_{ai} v^{x\,(t)}_{ia}}_{O(1/N)}\Big) \qquad (3.46) \\
&\approx H\Big(y_a,\, a_{ai}(x_i - \hat{x}^{(t)}_{i}) + \hat{p}^{(t)}_a,\, v^{p(t)}_a\Big),
\end{aligned}$$

and using Taylor's expansion of $H(\cdot,\cdot,\cdot)$, message $\hat{m}^{(t)}_{a\to i}(x_i)$ can be approximated with

$$\begin{aligned}\hat{m}^{(t)}_{a\to i}(x_i) \approx \text{const} + H\Big(y_a,\, \hat{p}^{(t)}_a,\, v^{p\,(t)}_a\Big) + a_{ai}(x_i - \hat{x}^{(t)}_i)H'\Big(y_a,\, \hat{p}^{(t)}_a,\, v^{p\,(t)}_a\Big)\\ + \frac{1}{2}a^2_{ai}(x_i - \hat{x}^{(t)}_i)^2 H''\Big(y_a,\, \hat{p}^{(t)}_a,\, v^{p\,(t)}_a\Big).\end{aligned} \tag{3.47}$$

Following the derivation in [79, p. 14], it can be shown that the first derivative and negative of the second derivative of $H(\cdot,\cdot,\cdot)$ in (3.47) are equal to

$$\begin{aligned}\hat{s}^{(t)}_a = H'\Big(y_a,\, \hat{p}^{(t)}_a,\, v^{p\,(t)}_a\Big) &= \frac{1}{v^{p\,(t)}_a}\Big(\mathbb{E}\Big\{z_a \,\Big|\, y_a;\, \hat{p}^{(t)}_a,\, v^{p\,(t)}_a\Big\} - \hat{p}^{(t)}_a\Big),\\ v^{s\,(t)}_a = -H''\Big(y_a,\, \hat{p}^{(t)}_a,\, v^{p\,(t)}_a\Big) &= \frac{1}{v^{p\,(t)}_a}\left(1 - \frac{\operatorname{var}\Big\{z_a \,\Big|\, y_a;\, \hat{p}^{(t)}_a,\, v^{p\,(t)}_a\Big\}}{v^{p\,(t)}_a}\right),\end{aligned} \tag{3.48}$$

where $p(z_a \,|\, y_a;\, \hat{p}^{(t)}_a,\, v^{p\,(t)}_a) \propto p(y_a \,|\, z_a)\,\mathcal{N}(\hat{p}^{(t)}_a,\, v^{p\,(t)}_a)$. Therefore, we can finally write (3.47) as

$$\begin{aligned}\hat{m}^{(t)}_{a\to i}(x_i) &\approx \text{const} + a_{ai}(x_i - \hat{x}^{(t)}_i)\,\hat{s}^{(t)}_a - \frac{1}{2}a^2_{ai}\,(x_i - \hat{x}^{(t)}_i)^2 v^{s\,(t)}_a\\ &= \text{const} + (a_{ai}\hat{s}^{(t)}_a + a^2_{ai}v^{s\,(t)}_a\,\hat{x}^{(t)}_i)x_i - \frac{1}{2}a^2_{ai}\,v^{s\,(t)}_a x^2_i.\end{aligned} \tag{3.49}$$

Approximation of $m^{(t+1)}_{i\to a}(x_i)$: Inserting (3.49) into (3.40) we get

$$\begin{aligned}m^{(t+1)}_{i\to a}(x_i) &= \text{const} + \log p(x_i;\alpha) + \sum_{b\neq a}\hat{m}^{(t)}_{b\to i}(x_i)\\ &\approx \text{const} + \log p(x_i;\alpha) + \sum_{b\neq a}(a_{bi}\,\hat{s}^{(t)}_b + a^2_{bi}\,v^{s\,(t)}_b\hat{x}^{(t)}_i)x_i - \frac{1}{2}a^2_{bi}\,v^{s\,(t)}_b x^2_i\\ &= \text{const} + \log p(x_i;\alpha) - \frac{1}{2}\Big(\sum_{b\neq a}a^2_{bi}\,v^{s\,(t)}_b\Big)\left(x_i - \frac{\sum_{b\neq a}a_{bi}\,\hat{s}^{(t)}_b + a^2_{bi}\,v^{s\,(t)}_b\hat{x}^{(t)}_i}{\sum_{b\neq a}a^2_{bi}\,v^{s\,(t)}_b}\right)^2\\ &= \text{const} + \log p(x_i;\alpha) - \frac{1}{2v^{r\,(t)}_{ia}}(x_i - \hat{r}^{(t)}_{ia})^2,\end{aligned} \tag{3.50}$$

where

$$\begin{aligned}\hat{r}^{(t)}_{ia} &= \hat{x}^{(t)}_i + v^{r\,(t)}_{ia}\sum_{b\neq a}a_{bi}\,\hat{s}^{(t)}_b,\\ v^{r\,(t)}_{ia} &= \Big(\sum_{b\neq a}a^2_{bi}\,v^{s\,(t)}_b\Big)^{-1}.\end{aligned} \tag{3.51}$$

Since we have approximated the distribution of $x_i \,\big|\, m^{(t+1)}_{i\to a}(x_i)$ with $p(x_i;\alpha)\,\mathcal{N}(x_i;\hat{r}^{(t)}_{ia}, v^{r\,(t)}_{ia})$, we can now calculate the mean $\hat{x}^{(t+1)}_{ia}$ and the variance $v^{x\,(t+1)}_{ia}$ of the message $m^{(t+1)}_{i\to a}(x_i)$ defined in (3.8). Furthermore, since this distribution of $x_i \,\big|\, m^{(t+1)}_{i\to a}(x_i)$ has the same structure as the one in (3.42), i.e., a local factor times a Gaussian distribution, we can use the results of (3.48), and together with Taylors expansion to write[6]

$$\hat{x}_{ia}^{(t+1)} \approx \hat{x}_i^{(t+1)} - a_{ai}\hat{s}_a^{(t)} v_i^{x\,(t+1)}, \tag{3.52}$$

where

$$\begin{aligned} \hat{x}_i^{(t+1)} &= \mathbb{E}\left\{x_i \,\middle|\, m_i^{(t+1)}(x_i)\right\}, \\ v_i^{x\,(t+1)} &= \operatorname{var}\left\{x_i \,\middle|\, m_i^{(t+1)}(x_i)\right\}, \end{aligned} \tag{3.53}$$

and $x_i \,\big|\, m_i^{(t+1)}(x_i)$ is approximately distributed as $p(x_i;\alpha)\,\mathcal{N}(x_i;\hat{r}_i^{(t)}, v_i^{r\,(t)})$, and

$$\begin{aligned} \hat{r}_i^{(t)} &\approx \hat{x}_i^{(t)} + v_i^{r\,(t)} \sum_b a_{bi}\,\hat{s}_b^{(t)}, \\ v_i^{r\,(t)} &\approx \Big(\sum_b a_{bi}^2\, v_b^{s\,(t)}\Big)^{-1}. \end{aligned} \tag{3.54}$$

It remains to approximate $\hat{p}_{ai}^{(t+1)}$ and $v_{ai}^{p\,(t+1)}$ with edge independent quantities. Going back to (3.43), we can write

$$\begin{aligned} \hat{p}_{ai}^{(t+1)} &= \sum_i a_{ai}\hat{x}_{ia}^{(t+1)} \overset{(3.52)}{\approx} \sum_i a_{ai}(\hat{x}_i^{(t+1)} - a_{ai}\hat{s}_a^{(t)} v_i^{x\,(t+1)}) \\ &= \sum_i a_{ai}\hat{x}_i^{(t+1)} - \hat{s}_a^{(t)} \sum_i a_{ai}^2 v_i^{x\,(t+1)} = \sum_i a_{ai}\hat{x}_i^{(t+1)} - \hat{s}_a^{(t)} v_a^{p\,(t+1)} = \hat{p}_a^{(t+1)}, \\ v_a^{p\,(t+1)} &\approx \sum_i a_{ai}^2 v_i^{x\,(t+1)}. \end{aligned} \tag{3.55}$$

Summary of the GAMP algorithm

Here, we summarize the key equations of the GAMP algorithm. Equations (3.55), and (3.48) constitute so-called measurement update (3.56), as they calculate posterior mean and variance of the linear mixtures $\mathbf{z}$. Similarly, with equations (3.54) and (3.53) we calculate calculate posterior mean and variance of the unknown sparse vector $\mathbf{x}$. Hence, equation (3.57) gives the so-called estimation update.

[6]Details can be found in [79, p. 17].

Measurement update:

$$\begin{aligned}
v_a^{p(t)} &= \sum_i a_{ai}^2 v_i^{x(t)},\\
\hat{p}_a^{(t)} &= \sum_i a_{ai}\hat{x}_i^{(t)} - \hat{s}_a^{(t-1)} v_a^{p(t)},\\
\hat{s}_a^{(t)} &= \frac{1}{v_a^{p(t)}}\Big(\mathbb{E}\Big\{z_a \,\Big|\, y_a;\, \hat{p}_a^{(t)},\, v_a^{p(t)}\Big\} - \hat{p}_a^{(t)}\Big),\\
v_a^{s(t)} &= \frac{1}{v_a^{p(t)}}\left(1 - \frac{\operatorname{var}\Big\{z_a \,\Big|\, y_a;\, \hat{p}_a^{(t)},\, v_a^{p(t)}\Big\}}{v_a^{p(t)}}\right),
\end{aligned} \tag{3.56}$$

where $p(z_a \,|\, y_a;\, \hat{p}_a^{(t)},\, v_a^{p(t)}) \propto p(y_a \,|\, z_a)\,\mathcal{N}(\hat{p}_a^{(t)},\, v_a^{p(t)})$.

Estimation update:

$$\begin{aligned}
\hat{r}_i^{(t)} &= \hat{x}_i^{(t)} + v_i^{r(t)} \sum_b a_{bi}\, \hat{s}_b^{(t)},\\
v_i^{r(t)} &= \Big(\sum_b a_{bi}^2\, v_b^{s(t)}\Big)^{-1},\\
\hat{x}_i^{(t+1)} &= \mathbb{E}\Big\{x_i \,\Big|\, \hat{r}_i^{(t)}\Big\},\\
v_i^{x(t+1)} &= \operatorname{var}\Big\{x_i \,\Big|\, \hat{r}_i^{(t)}\Big\},
\end{aligned} \tag{3.57}$$

where $p(x_i \,|\, \hat{r}_i^{(t)}) \propto p(x_i;\alpha)\,\mathcal{N}(x_i;\hat{r}_i^{(t)}, v_i^{r(t)})$.

Finally, the algorithm is shown in Algorithm 6, where we used vector notation. Here, functions $\mathrm{F}_1(\cdot)$, $\mathrm{F}_2(\cdot)$, $\mathrm{G}_1(\cdot)$, and $\mathrm{G}_2(\cdot)$ are applied component-wise and are given by

$$\begin{aligned}
\mathrm{F}_1(y,\hat{p},v_p) &= \frac{\mathbb{E}\{z \,|\, y\,;\, \hat{p}_a^{(t)},\, v_a^{p(t)}\} - \hat{p}}{v_p}, &\quad \mathrm{G}_1(\hat{r},v_r,\alpha) &= \mathbb{E}\{x \,|\, \hat{r}\,; v_i^{r(t)}, \alpha\},\\
\mathrm{F}_2(y,\hat{p},v_p) &= \frac{v_p - \operatorname{var}\{z \,|\, y\,;\, \hat{p}_a^{(t)},\, v_a^{p(t)}\}}{v_p^2}, &\quad \mathrm{G}_2(\hat{r},v_r,\alpha) &= \operatorname{var}\{x \,|\, \hat{r}\,; v_i^{r(t)}, \alpha\},
\end{aligned} \tag{3.58}$$

where

$$\begin{aligned}
p(z_a \,|\, y_a;\, \hat{p}_a^{(t)},\, v_a^{p(t)}) &\propto p(y_a \,|\, z_a)\,\mathcal{N}(\hat{p}_a^{(t)},\, v_a^{p(t)}),\\
p(x_i \,|\, \hat{r}_i^{(t)}; v_i^{r(t)}, \alpha) &\propto p(x_i;\alpha)\,\mathcal{N}(x_i;\hat{r}_i^{(t)}, v_i^{r(t)}).
\end{aligned} \tag{3.59}$$

Algorithm 6 Generalized approximate message passing algorithm

```
Input: measurement matrix $\mathbf{A}$, measurement vector $\mathbf{y}$, noise level $\sigma_w^2$ (optional),
    input channel distribution $p(x_i|\alpha)$, output channel distribution $p(y_a|z_a)$, stopping
    threshold $\varepsilon$ or $t_{\max}$
Initialization: $t = 0$, $\hat{\mathbf{x}}^{(0)} = \mathbb{E}\{\mathbf{x}\}$, $\mathbf{v}_{\mathbf{x}}^{(0)} = \mathrm{var}\{\mathbf{x}\}$, $\hat{\mathbf{s}}^{(0)} = 0_{m\times 1}$
do:
  1: $t = t + 1$                                                              ▷ increment iteration counter
  2: $\mathbf{v}^{p(t)} = (\mathbf{A} \bullet \mathbf{A})\mathbf{v}^{x(t-1)}$                    ▷ Measurement update - linear step
  3: $\hat{\mathbf{p}}^{(t)} = \mathbf{A}\hat{\mathbf{x}}^{(t-1)} - \mathbf{v}^{p(t)} \bullet \hat{\mathbf{s}}^{(t-1)}$     ▷ Measurement update - linear step
  4: $\hat{\mathbf{s}}^{(t)} = \mathrm{F}_1(\mathbf{y}, \hat{\mathbf{p}}^{(t)}, \mathbf{v}^{p(t)})$                 ▷ Measurement update - nonlinear step
  5: $\mathbf{v}^{s(t)} = \mathrm{F}_2(\mathbf{y}, \hat{\mathbf{p}}^{(t)}, \mathbf{v}^{p(t)})$                 ▷ Measurement update - nonlinear step
  6: $\mathbf{v}^{r(t)} = \left((\mathbf{A} \bullet \mathbf{A})^T \mathbf{v}^{s(t)}\right)^{-1}$              ▷ Estimation update - linear step
  7: $\hat{\mathbf{r}}^{(t)} = \hat{\mathbf{x}}^{(t-1)} + \mathbf{v}^{r(t)} \bullet (\mathbf{A}^T \hat{\mathbf{s}}^{(t)})$     ▷ Estimation update - linear step
  8: $\hat{\mathbf{x}}^{(t)} = \mathrm{G}_1(\hat{\mathbf{r}}^{(t)}, \mathbf{v}^{r(t)}; \alpha)$                  ▷ Estimation update - nonlinear step
  9: $\mathbf{v}^{x(t)} = \mathrm{G}_2(\hat{\mathbf{r}}^{(t)}, \mathbf{v}^{r(t)}; \alpha)$                  ▷ Estimation update - nonlinear step
while $t \leq t_{\max}$ or $\|\hat{\mathbf{x}}^{(t)} - \hat{\mathbf{x}}^{(t-1)}\|_2 \geq \varepsilon \, \|\hat{\mathbf{x}}^{(t)}\|_2$
Output: $\hat{\mathbf{x}} = \mathbf{x}^{(t)}$
```

Chapter 4

Analysis-by-Synthesis with Bayesian Approximate Message Passing Scalar Quantization for Compressed Sensing

In this chapter, I consider a scenario where sensors using *compressed sensing* (CS) observe a sparse signal, quantize the observations, and transmit the discrete valued data over a communication link with a low rate constraint. I start by stating the problem of quantization of CS measurements and providing a more intuitive derivation of the optimal quantizer than offered in literature. An approximation of the optimal quantizer is offered by the previously proposed *Analysis-by-Synthesis* (AbS) quantization scheme for CS. I present this quantization scheme in which, as a novelty, I adopt of the *Bayesian approximate message passing* (BAMP) algorithm as the recovery algorithm. I focus on source signals that can be modeled as a linear combination of a discrete component and a zero-mean Gaussian component; for those signals, I provide analytical expressions for the estimation functions of the BAMP algorithm. I compare the results of AbS when BAMP is used as the recovery algorithm with an AbS scheme known from literature, in which *orthogonal matching pursuit* (OMP) is used as the recovery algorithm. Additionally, I investigate different setups of the AbS scheme.

4.1 Introduction

As we have seen in Subsection 2.4.2, reconstruction with the classical CS recovery algorithms on quantized measurements results in poor accuracy [24, 45, 50]. Therefore, suitable algorithms, which take into account the whole measurement process, need to be developed to suppress the negative effects of quantization on the recovered source signal. In [76] an approximation of the *end-to-end mean squared error* (MSE) minimizing quantization scheme is proposed that uses the concept of AbS. In AbS, the neighbourhood of a scalar-quantized measurement vector is investigated, with the aim of finding a representation that will give a lower end-to-end MSE after applying a nonlinear recovery algorithm. The proposed scheme allows for the use of any CS reconstruction algorithm, and the authors adopt OMP for that task.

Recent advances in graph-based algorithms for CS recovery show promising results [32–34, 57, 71, 72]. In particular, BAMP [33, 34] is most appealing for its simplicity and small reconstruction errors. High reconstruction accuracy is possible since, as opposed to most classical recovery algorithms, the BAMP algorithm uses explicitly the information about source signal prior. Although it seems disadvantageous that the signal prior needs to be known, recent work [42, 82] demonstrated that the prior can be estimated during the iterations of the algorithm from the measurements alone. The explicit knowledge of the prior is, therefore, not required.

The above presented arguments motivate us to adopt BAMP as the reconstruction algorithm in the AbS scheme. In what follows, I start by giving an overview of the AbS scheme, which was presented in [76]. Subsequently, I explain some practical details of implementation of the BAMP algorithm and give explicit solutions for the BAMP operators for the assumed signal prior. Finally, I present performance results of the BAMP algorithm in AbS scheme in a fixed bit budget scenario. The results are compared with those from [76]. My results demonstrate that the BAMP algorithm significantly outperforms the much more complex OMP algorithm in the AbS quantization scheme.

4.2 Problem Statement

I assume that a sensor observes linear mixtures $\mathbf{y}$ of a sparse signal $\mathbf{x}$ according to (6.6), i.e.,

$$\mathbf{y} = \mathbf{A}\mathbf{x}, \tag{4.1}$$

where $\mathbf{A}$ is the $m \times N$ measurement matrix. The vector of linear mixtures $\mathbf{y}$ will be referred to as the vector of CS measurements. Furthermore, I assume that the CS measurements are quantized with r_y integer number of bits per measurement. The rate r_y can be expressed as

$$r_y = r_x/\delta, \tag{4.2}$$

where r_x is the given, fixed bit-rate per source component, and δ is the measurement ratio, i.e., m/N. It is assumed that the code alphabet $\mathcal{C} = \{c_0, c_1, ..., c_{2^{r_y}-1}\}$, containing the code symbols c_i, is designed offline by the Lloyd-Max algorithm [59].

I consider an encoder $\mathrm{E} : \mathbb{R}^m \rightarrow \mathcal{I}^m$ that operates on the entire vector $\mathbf{y}$, and outputs a vector of indices $\mathbf{i} \in \mathcal{I}^m$, where $\mathcal{I} \doteq \{0, 1, ..., 2^{r_y} - 1\}$. The codeword $\hat{\mathbf{y}}$ is determined by $\mathbf{i}$ and $\mathcal{C}$. The n-th entry $\hat{y}_n$ of the codeword $\hat{\mathbf{y}}$ is equal to symbol c_{i_n}; repeating this for each $n \in [m]$ produces the entire codeword $\hat{\mathbf{y}}$. The estimate $\hat{\mathbf{x}}$ of the source vector is then obtained from the codeword, according to

$$\hat{\mathbf{x}} = \Delta(\mathbf{A}, \hat{\mathbf{y}}), \tag{4.3}$$

where Δ is a chosen CS reconstruction algorithm.

Given a statistical model for $\mathbf{x}$, the alphabet $\mathcal{C}$, and the measurement matrix $\mathbf{A}$, our task is to find an encoder E, that performs well based on some performance metric. As suggested in [76], to compare different solutions I use the end-to-end MSE, defined as

$$\mathrm{MSE} = \mathbb{E}_{\mathbf{x}}\Big\{\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2\Big\}, \tag{4.4}$$

as the performance criterion.

4.2.1 Optimal quantization for a given alphabet

Adopting the end-to-end MSE as the performance criterion, it is observed in [76] that

$$\begin{aligned} \mathrm{MSE} &= \mathbb{E}_{\mathbf{x}}\Big\{\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2\Big\} = \mathbb{E}_{\mathbf{x}}\Big\{\|\mathbf{x} - \hat{\mathbf{x}}(\mathbf{i})\|_2^2\Big\} \\ &= \int_{\mathbf{y}} \mathbb{E}_{\mathbf{x}}\Big\{\|\mathbf{x} - \hat{\mathbf{x}}(\mathbf{i})\|_2^2 \,\Big|\, \mathbf{y} = \mathbf{y}\Big\} f_{\mathbf{y}}(\mathbf{y}) d\mathbf{y} \\ &= \int_{\mathbf{y}} \underbrace{\mathbb{E}_{\mathbf{x}}\Big\{\|\mathbf{x} - \hat{\mathbf{x}}(\mathrm{E}(\mathbf{y}))\|_2^2 \,\Big|\, \mathbf{y} = \mathbf{y}\Big\}}_{\mathbb{E}_{\mathbf{x}|\mathbf{y}}} f_{\mathbf{y}}(\mathbf{y}) d\mathbf{y}. \end{aligned} \tag{4.5}$$

In (4.5), the only term left for us to choose, i.e., the only "free parameter", is the encoder function $\mathrm{E} : \mathbb{R}^m \rightarrow \mathcal{I}^m$. Since $f(\mathbf{y})$ is a distribution, and hence a nonnegative function, the optimal encoder $\mathrm{E}^*(\mathbf{y})$ that minimizes the end-to-end MSE is the one that minimizes $\mathbb{E}_{\mathsf{x}|\mathsf{y}}$ for every observation $\mathbf{y}$. In other words, the optimal encoding $\mathrm{E}^*(\mathbf{y})$ is given by

$$\mathrm{E}^*(\mathbf{y}) = \mathbf{i}^* = \arg\min_{\mathbf{i}\in\mathcal{I}^m} \mathbb{E}_{\mathsf{x}} \left\{ \|\mathbf{x} - \hat{\mathbf{x}}(\mathbf{i})\|_2^2 \,\middle|\, \mathsf{y} = \mathbf{y} \right\}. \tag{4.6}$$

The last equation can be rewritten as

$$\begin{aligned}
\mathbf{i}^* &= \arg\min_{\mathbf{i}\in\mathcal{I}^m} \mathbb{E}_{\mathsf{x}}\left\{ \|\mathbf{x} - \hat{\mathbf{x}}(\mathbf{i})\|_2^2 \,\middle|\, \mathsf{y} = \mathbf{y} \right\} \\
&= \arg\min_{\mathbf{i}\in\mathcal{I}^m} \mathbb{E}_{\mathsf{x}}\left\{ \|\mathbf{x}\|_2^2 \,\middle|\, \mathsf{y} = \mathbf{y} \right\} + \mathbb{E}_{\mathsf{x}}\left\{ \|\hat{\mathbf{x}}(\mathbf{i})\|_2^2 \,\middle|\, \mathsf{y} = \mathbf{y} \right\} - 2\,\mathbb{E}_{\mathsf{x}}\left\{ \mathbf{x}^T\hat{\mathbf{x}}(\mathbf{i}) \,\middle|\, \mathsf{y} = \mathbf{y} \right\} \\
&\overset{(a)}{=} \arg\min_{\mathbf{i}\in\mathcal{I}^m} \mathbb{E}_{\mathsf{x}}\left\{ \|\hat{\mathbf{x}}(\mathbf{i})\|_2^2 \,\middle|\, \mathsf{y} = \mathbf{y} \right\} - 2\,\mathbb{E}_{\mathsf{x}}\left\{ \mathbf{x}^T\hat{\mathbf{x}}(\mathbf{i}) \,\middle|\, \mathsf{y} = \mathbf{y} \right\} \\
&\overset{(b)}{=} \arg\min_{\mathbf{i}\in\mathcal{I}^m} \|\hat{\mathbf{x}}(\mathbf{i})\|_2^2 - 2\,\hat{\mathbf{x}}(\mathbf{i})^T \mathbb{E}_{\mathsf{x}}\left\{ \mathbf{x} \,\middle|\, \mathsf{y} = \mathbf{y} \right\} \\
&= \arg\min_{\mathbf{i}\in\mathcal{I}^m} \|\hat{\mathbf{x}}(\mathbf{i})\|_2^2 - 2\,\hat{\mathbf{x}}(\mathbf{i})^T \tilde{\mathbf{x}}(\mathbf{y}),
\end{aligned} \tag{4.7}$$

where (a) follows from the fact that x is independent of $\mathbf{i}$ conditioned on y, (b) follows from statistical independence of $\hat{\mathsf{x}}$ and x conditioned on y, and $\tilde{\mathbf{x}}(\mathbf{y})$ denotes the *minimum mean squared error* (MMSE) estimate of the sparse vector x given the (unquantized) observation $\mathbf{y}$.

The minimization in (4.7) requires a search over all $2^{m\cdot r_y}$ possible index vectors $\mathbf{i}$. This poses a practical limitation as the computational complexity grows exponentially with the number of measurements. Therefore, an approximation of the optimal solution must be found with complexity that is at most polynomial m (rather than exponential).

4.2.2 Scalar quantization of the measurements

The simplest way is to directly quantize each measurement y_n separately, by searching for the nearest neighbour of y_n in the given alphabet $\mathcal{C}$, i.e., to use the scalar quantizer. The effect of the quantization on the recovered source signal $\hat{\mathbf{x}}$ is, therefore, not considered during the quantization; the advantage is simplicity. Later, treating quantization errors as additive noise, any CS recovery algorithm can be used for the reconstruction.

4.2.3 Quantization using Analysis-by-Synthesis

The AbS quantization scheme from [76], shown in Figure 4.1, offers a way to find a suboptimal, but computationally feasible solution of (4.7). The solution is found by iteratively choosing the best quantizer index i_n for some $n \in [m]$, while fixing all the

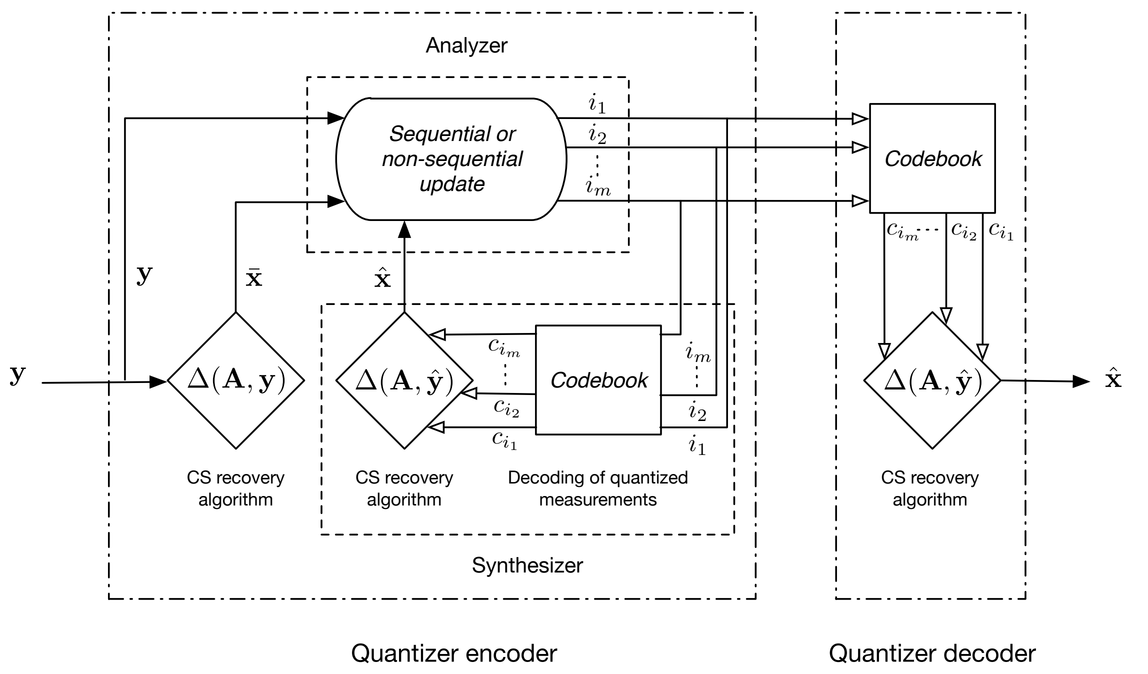


Fig. 4.1 Analysis-by-Synthesis Quantization Scheme: A suboptimal solution is found by iteratively choosing the best quantizer index i_n, while fixing all other encoded indices $\{i_k\}_{1\neq n}^{m}$.

other quantizer indices i_1,i_2, ..., i_{n-1}, i_{n+1}, ..., i_m. The optimality criterion for selecting a single quantizer index i_n^* to quantize the measurement y_n is then given by

$$i_n^* = \arg \min_{i_n \in \mathcal{I}\,;\,\{\mathbf{i} \setminus i_n\}\text{ fixed}} \left\{ \|\hat{\mathbf{x}}(\mathbf{i})\|_2^2 - 2\bar{\mathbf{x}}(\mathbf{y})^T \hat{\mathbf{x}}(\mathbf{i}) \right\}. \tag{4.8}$$

The last equation motivates the term "Analysis-by-Synthesis", and in particular:

- synthesis: for any choice of the index vector $\mathbf{i}$ the estimate $\hat{\mathbf{x}}$ is "synthesized" by the recovery algorithm Δ
- analysis: given all other indices, the chosen index i_n is the one that minimizes the end-to-end MSE

Note that $\tilde{\mathbf{x}}(\mathbf{y})$ equals $\mathbf{x}$ only if the reconstruction from the noiseless and unquantized measurements $\mathbf{y} = \mathbf{A}\mathbf{x}$ is exact. This requires, however, that the number of measurements m is sufficiently large. What "sufficient" exactly means depends on the chosen recovery algorithm (all the time assuming that the signal $\mathbf{x}$ is indeed strictly k-sparse). As m is a design parameter, due to too few measurements, it may be the case that the output of the CS recovery algorithms is not perfect. Furthermore, even if m was indeed large enough, the MMSE estimate $\tilde{\mathbf{x}}(\mathbf{y})$ is difficult to compute exactly in reasonable

Algorithm 7 AbS [76, AbS quantization]

```
Input: unquantized measurements $\mathbf{y}$, alphabet $\mathcal{C}$, measurement matrix $\mathbf{A}$, update
   type AbS_update (options: AbS_seq or AbS_nonseq), stopping threshold $\varepsilon_{\text{stop}}$
Initialization:
   $\hat{\mathbf{y}}^{(0)} = \text{nn}(\mathbf{y})$                    ▷ initialize with nearest neighbours
   $\bar{\mathbf{x}}(\mathbf{y}) = f(\mathbf{A}, \mathbf{y})$                ▷ compute approximation of the MMSE estimate
do:
   $l = l + 1$                                ▷ increment iteration counter
   $[\mathbf{i}^{(l)}, \hat{\mathbf{y}}^{(l)}, \hat{\mathbf{x}}^{(l)}] = \text{AbS\_update}(\hat{\mathbf{y}}^{(l-1)}, \bar{\mathbf{x}}(\mathbf{y}), \mathcal{C}, \mathbf{A})$   ▷ choose representations
while $\left| \left[ \|\hat{\mathbf{x}}^{(l)}\|_2^2 - 2\bar{\mathbf{x}}(\mathbf{y})^T \hat{\mathbf{x}}^{(l)} \right] - \left[ \|\hat{\mathbf{x}}^{(l-1)}\|_2^2 - 2\bar{\mathbf{x}}(\mathbf{y})^T \hat{\mathbf{x}}^{(l-1)} \right] \right| < \varepsilon_{\text{stop}}$
Output: $\mathbf{i}^{(l)}, \hat{\mathbf{y}}^{(l)}, \hat{\mathbf{x}}^{(l)}$
```

time as it involves multidimensional integration. Therefore, in (4.8), $\tilde{\mathbf{x}}(\mathbf{y})$ is replaced by the output of the specific CS recovery algorithm, i.e.,

$$\tilde{\mathbf{x}}(\mathbf{y}) \approx \bar{\mathbf{x}}(\mathbf{y}) = \Delta(\mathbf{A},\ \mathbf{y}). \tag{4.9}$$

Since the BAMP algorithm is designed to approximate $\tilde{\mathbf{x}}(\mathbf{y})$, it is likely to perform better than competing approaches (e.g., OMP), that, in fact, have a different optimization goal.

To start the step-wise optimization of the indices i_n, given by Algorithm 7, an initialization for the vector $\mathbf{i}$ is required. A sensible choice is to use the result of a scalar quantization of the components of the measurement vector $\mathbf{y}$. Furthermore, there are various ways to choose the order in which the quantizer indices i_n are updated. Two methods proposed in [76] are *sequential* and *non-sequential* selection.

Sequential update For the sequential selection, the indices are simply updated in their natural numerical order $n = 1, 2, \dots, m$. To update the entire vector $\mathbf{i}$, optimization in (4.8) is performed m times. The steps of the sequential update are presented in Algorithm 8.

Non-sequential update For the non-sequential selection on the other hand, the optimal index is calculated (but not yet updated) according to (4.8) for all indices that were *not* updated in any of the previous steps. The index i_k^* whose update would produce the smallest value $\|\hat{\mathbf{x}}(\mathbf{i})\|_2^2 - 2\bar{\mathbf{x}}(\mathbf{y})^T\hat{\mathbf{x}}(\mathbf{i})$, is selected and updated. This process repeats until all quantizer indices $\{i_n\}_{n=1}^m$ have been optimized. To update the entire

Algorithm 8 AbS sequential update [76, AbS_seq]

```
Input: quantized measurements $\hat{\mathbf{y}}^{(l-1)}$ or equivalently $\mathbf{i}^{(l-1)}$, approximation of the
  MMSE estimate $\bar{\mathbf{x}}(\mathbf{y})$, alphabet $\mathcal{C}$, measurement matrix $\mathbf{A}$
Initialization: $\hat{\mathbf{y}}^{(l)} = \hat{\mathbf{y}}^{(l-1)}$
  for $n = 1 : m$ do
      for $i = 0 : |\mathcal{C}| - 1$ do
          $\hat{y}_n^{(l)} = c_i$                              ▷ encode $n$-th entry of $\mathbf{y}$ with $c_i$
          $\hat{\mathbf{x}}^{(n,i)} = f(\mathbf{A}, \hat{\mathbf{y}}^{(l)})$       ▷ compute potential recovery of the sparse vector
      end for
      $i_n^* = \arg\min_{i\in\{0:|\mathcal{C}|-1\}} \left\{\|\hat{\mathbf{x}}^{(n,i)}\|_2^2 - 2\bar{\mathbf{x}}(\mathbf{y})^T\hat{\mathbf{x}}^{(n,i)}\right\}$ ▷ choose the best representation
      $[i_n^{(l)}, \hat{y}_n^{(l)}] = [i_n^*, c_{i_n^*}]$                 ▷ update index and representation
  end for
Output: $\mathbf{i}^{(l)}, \hat{\mathbf{y}}^{(l)}, \hat{\mathbf{x}}(\hat{\mathbf{y}}^{(l)})$
```

Algorithm 9 AbS nonsequential update [76, AbS_nonseq]

```
Input: quantized measurements $\hat{\mathbf{y}}^{(l-1)}$ or equivalently $\mathbf{i}^{(l-1)}$, approximation of the
  MMSE estimate $\bar{\mathbf{x}}(\mathbf{y})$, alphabet $\mathcal{C}$, measurement matrix $\mathbf{A}$
Initialization: $\hat{\mathbf{y}}^{(l)} = \hat{\mathbf{y}}^{(l-1)}, \mathcal{L} = \emptyset$
  while $|\mathcal{L}| < m$ do
      for each $n \in \{1 : m\} \setminus \mathcal{L}$ do
          $\hat{\mathbf{y}}^{(l)} = \hat{\mathbf{y}}^{(l-1)}$                    ▷ initialize quantized measurements
          for $i = 0 : |\mathcal{C}| - 1$ do
              $\hat{y}_n^{(l)} = c_i$                          ▷ encode $n$-th entry of $\mathbf{y}$ with $c_i$
              $\hat{\mathbf{x}}^{(n,i)} = f(\mathbf{A}, \hat{\mathbf{y}}^{(l)})$                ▷ compute potential recovery
              $\Delta(n,i) = \|\hat{\mathbf{x}}^{(n,i)}\|_2^2 - 2\bar{\mathbf{x}}(\mathbf{y})^T\hat{\mathbf{x}}^{(n,i)}$          ▷ compute cost
          end for
      end for
      $[n^*, i_{n^*}^*] = \arg\min_{[n,i]} \Delta(n,i)$     ▷ choose the best index and representation
      $[i_{n^*}^{(l)}, \hat{y}_{n^*}^{(l)}] = [i_{n^*}^*, c_{i_{n^*}^*}]$             ▷ update index and representation
      $\mathcal{L} = \mathcal{L} \bigcup \{n^*\}$                        ▷ update set of updated indices
  end while
Output: $\mathbf{i}^{(l)}, \hat{\mathbf{y}}^{(l)}, \hat{\mathbf{x}}(\hat{\mathbf{y}}^{(l)})$
```

vector $\mathbf{i}$, the optimization in (4.8) is performed $m(m+1)/2$ times. The steps of the nonsequential update are presented in Algorithm 9.

Independent of the selected update method, once all m elements of $\mathbf{i}$ have been updated, the whole update process is repeated until convergence is reached, i.e., until the solution $\hat{\mathbf{x}}(\mathbf{i})$ remains within a pre-chosen accuracy limit from one iteration to the next. It has been shown in [76] that the step-wise optimization in (4.8) will converge for any CS recovery algorithm.

4.3 BAMP as the Reconstruction Algorithm

We know from Chapter 3 that, if the source prior is known (or can be estimated from the measurements), the BAMP algorithm can be used to recover a sparse vector from noisy measurements. As can be seen from Algorithm 5, at each iteration, the BAMP algorithm uses the information about the noise variance σ_w^2 to calculate the effective noise variance. In classical noisy CS, the noise variance σ_w^2 is estimated at the receiver before running the algorithm. In our scenario, however, the noise is, in fact, the distortion due to the quantization. When quantizing the measurements, the independent-noise model becomes questionable (particularly at low rate) and the distortion is hard to find in advance and analytically.

Alternatively, the effective noise variance $c^{(t+1)}$ can be approximated. In [17, Chapter 9.5.1], the effective noise variance is approximated with $\frac{1}{m}\|\mathbf{z}^{(t)}\|_2^2$, where $\mathbf{z}^{(t)}$ is the residual defined in Algorithm 5. For the remainder of this chapter, the BAMP algorithm that estimates the effective noise variance using $\frac{1}{m}\|\mathbf{z}^{(t)}\|_2^2$ will be called the BAMP2 algorithm. On the other hand, the BAMP1 algorithm refers to the (classical) version of the BAMP algorithm with the known noise variance. In the following subsection, I discuss which version of the algorithm should be used within the AbS framework.

4.3.1 Analysis-by-Synthesis with BAMP1 or with BAMP2

The BAMP2 algorithm is particularly appealing in the AbS scheme, as the noise variance σ_w^2 is not used explicitly. On the other hand, since BAMP2 uses less prior knowledge (the noise variance) one would expect it to have worse performance compared to BAMP1.

To demonstrate the difference in performance between the BAMP1 and the BAMP2 algorithm I show results of a Monte-Carlo simulations. There, I assume that the CS measurements are corrupted with additive noise, i.e.,

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{w}, \tag{4.10}$$

where the entries of **w** are drawn from zero-mean Gaussian distribution with known variance. The dimension of the unknown vector N, sparsity k, and the measurement rate δ take values 512, 35, and 0.25, respectively. At each simulation instance, the entries of the measurement matrix A are drawn independently from a standard normal distribution, and the columns are normalized to have unit Euclidian norm. Furthermore, at each simulation instance, the source vector is drawn from the corresponding *Bernoulli-Gauss* (BG) mixture distribution. The unknown sparse source signal is reconstructed from noisy measurements using both the BAMP1 algorithm and the BAMP2 algorithm.

To compare different algorithms, I follow [76], and adopt *normalized mean squared error* (NMSE) as the performance metric. The NMSE is defined as

$$\text{NMSE/dB} = 10 \log_{10} \frac{\mathbb{E}_{\mathbf{x},\mathbf{w}} \left\{ \|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 \right\}}{\mathbb{E}_{\mathbf{x}} \left\{ \|\mathbf{x}\|_2^2 \right\}} , \tag{4.11}$$

where $\hat{\mathbf{x}}$ is the reconstruction of the source vector. In my results, I compute the NMSE by averaging results of *Monte-Carlo* (MC) simulations. Figure 4.2 shows the empirical NMSE of both BAMP algorithms against *signal-to-noise ratio* (SNR), where the SNR is defined as

$$\text{SNR} = \mathbb{E}_{\mathbf{x}} \left\{ \|\mathbf{A}\mathbf{x}\|_2^2 \right\} / \, \mathbb{E}_{\mathbf{w}} \left\{ \|\mathbf{w}\|_2^2 \right\}. \tag{4.12}$$

The figure shows that even with exact knowledge of the noise variance, BAMP1 does not produce significantly better results than BAMP2. Furthermore, as previously argued, within the AbS quantization framework the "noise variance" will not be known exactly in advance. Therefore, the performance of the BAMP1 algorithm might even deteriorate. In further investigations I will, hence, use only the BAMP2 algorithm.

4.3.2 BAMP operators for a sparse Gaussian signal prior

To obtain the BAMP operators $F(u_j; c)$ and $G(u_j; c)$, I assume that the *independent and identically distributed* (i.i.d.) prior p_{x_i} can be written as

$$p_{\mathsf{x}_i}(x; \epsilon, \sigma_x^2) = (1 - \epsilon)\delta(x) + \epsilon \mathcal{N}(x; 0, \sigma_x^2), \tag{4.13}$$

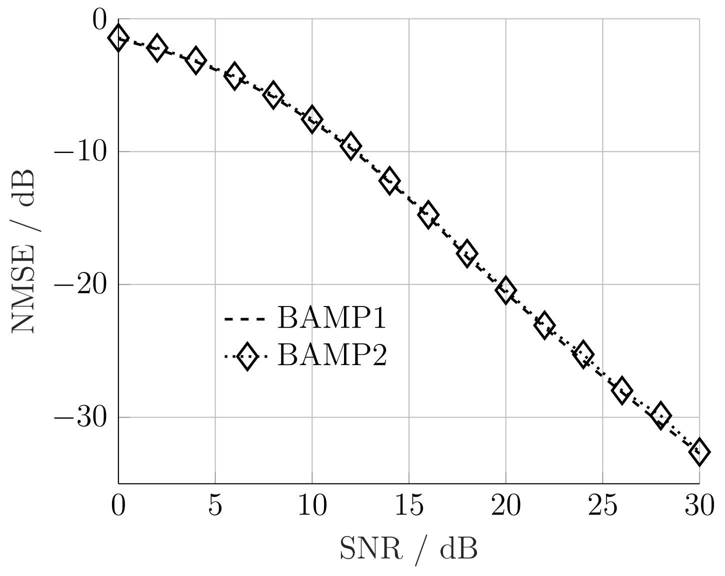


Fig. 4.2 NMSE in dB of BAMP1 and BAMP2 as a function of SNR. The parameters N, k, and δ are 512, 35 and 0.25, respectively.

where ϵ is the probability of nonzero value, and σ_x^2 is the power of the Gaussian component. This commonly used i.i.d. zero-mean BG mixture source prior is used in [76] as well. To obtain analytical expressions for $F(u_j;c)$ and $G(u_j;c)$, I use the results from Subsection 3.2.2. The operators can be written as

$$F(u_j;c) \;=\; u_j\,H(u_j), \tag{4.14}$$

$$G(u_j;c) \;=\; c\,H(u_j) \;+\; h(u_j)\;F^2(u_j;c), \tag{4.15}$$

where

$$H(u_j) = \frac{\sigma_x^2}{\sigma_x^2 + c}\,\frac{1}{1\;+\;h(u_j)}, \quad \text{and} \quad h(u_j) = \frac{1-\epsilon}{\epsilon}\sqrt{\frac{\sigma_x^2 + c}{c}}\;e^{-\frac{u_j^2}{2c}\frac{\sigma_x^2}{\sigma_x^2+c}}. \tag{4.16}$$

4.4 Numerical Results

To investigate the performance of the proposed quantization techniques I present the results of Monte-Carlo simulations, each with 1000 independent realizations of pairs of a source vector and a measurement matrix. The authors of [76] have kindly made available their Matlab code[1], so I could integrate the BAMP algorithm in their framework. For easier comparison of different reconstruction algorithms, I use the same

[1]https://people.kth.se/~amishi/reproducible_research.html

parameters as in [76]. In particular, the length of the source vector $N = 512$, and the sparsity level $k = 35$. This leads to a sparsity ratio of $k/N = 0.0684$. Consequently, the probability of a nonzero component $\epsilon = 0.0684$. The entries of the sensing matrix $\mathbf{A}$ and the non-zero components of the source vector $\mathbf{x}$ are drawn independently from a standard normal distribution. In order to have the same setup as in [76], the columns of the sensing matrix are later normalized to have unit Euclidean norm. This is, however, not a requirement of the BAMP algorithm. The alphabet $\mathcal{C}$ is designed off-line according to Lloyd's algorithm, with unquantized measurements as training data. The stopping thresholds for both the sequential and the non-sequential AbS are set to $\varepsilon_{\text{AbS}} = 10^{-6}$; where as the stopping threshold for the BAMP algorithms is set to $\varepsilon_{\text{BAMP}} = 10^{-4}$.

Throughout the simulations, the total amount of available bits (i.e., the bit budget) Nr_x is kept fixed. Since $r_y \triangleq Nr_x/m = r_x/\delta$, it follows that increasing the measurement rate will consequently decrease the amount of available bits per measurement component. For faster convergence, the AbS scheme is initialized with the nearest neighbour reproducer values.

4.4.1 Sequential or non-sequential update with BAMP2

As we have seen in Subsection 4.2.3, the non-sequential update in the AbS scheme produces significantly higher computational complexity than the sequential update. More specifically, in the non-sequential update the optimization in (4.8) is performed $m(m+1)/2$ times, compared to m times in the sequential update.

In Fig. 4.3, I analyse how much there is to be gained in terms of NMSE when using the non-sequential scheme (applied with BAMP2 for CS reconstruction). The NMSE against the measurement rate δ $(= m/N)$ for both types of updates with $r_x = 0.5$ and $r_x = 0.75$ bits per source component is shown in Fig. 4.3. We observe only a slight improvement in NMSE for non-sequential updates. The sequential update is, hence, preferable due to the much lower complexity.

4.4.2 BAMP2 or OMP in AbS

The performance of OMP as the reconstruction algorithm in the AbS quantization scheme has already been investigated in [76]. Here I compare those results to the ones obtained when BAMP2 is used as the reconstruction algorithm. Figs. 4.4 and 4.5 show the NMSE as a function of the measurement rate δ for different setups of the AbS scheme, and for given and fixed quantization bit-rate of $r_x = 0.5$ and $r_x = 0.75$,

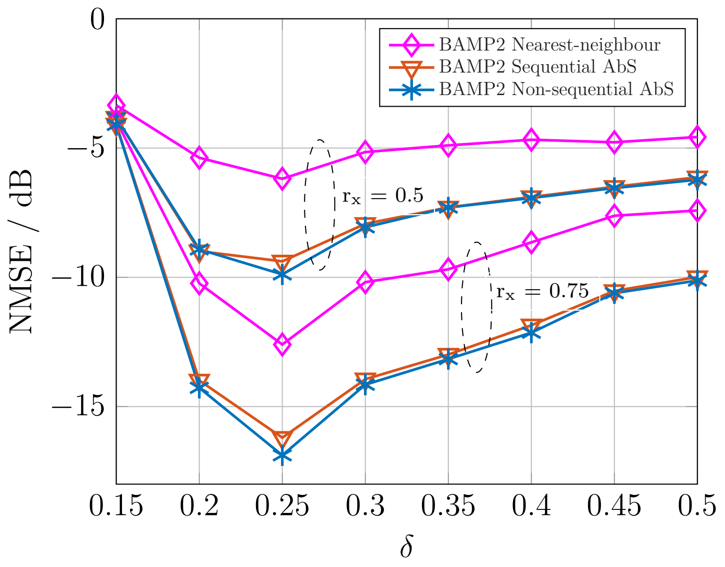


Fig. 4.3 NMSE vs. measurement rate δ $(= m/N)$ for sequential and non-sequential AbS with BAMP2. The comparison is for two fixed quantization bit-rates ($r_x = 0.5$ and $r_x = 0.75$ bits per component of the unknown sparse vector, equivalent to a total bit budget of $R_x = 256$ and $R_x = 384$ bits, respectively). Source vector dimension $N = 512$, sparsity $k = 35$.

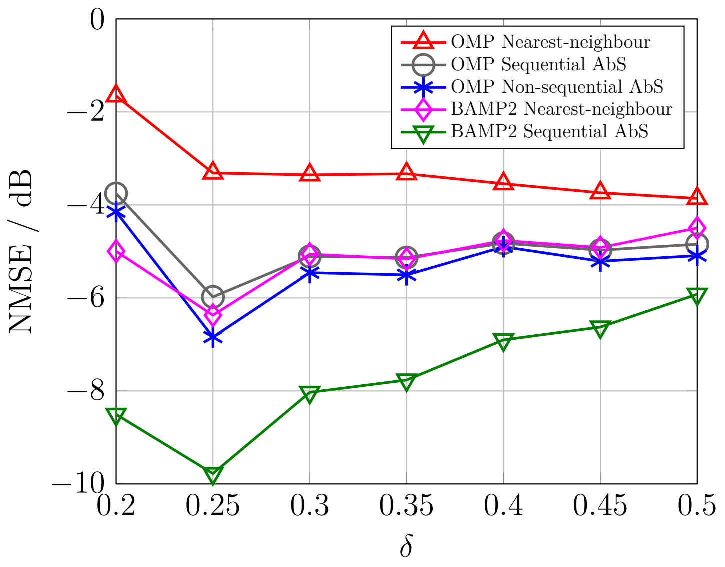


Fig. 4.4 NMSE vs. measurement rate δ $(= m/N)$ of AbS with OMP and BAMP2; fixed quantization bit-rate of $r_x = 0.5$ bits per component of the unknown sparse vector (equivalent to a total bit budget of $R_x = 256$ bits). Source vector dimension $N = 512$, sparsity $k = 35$.

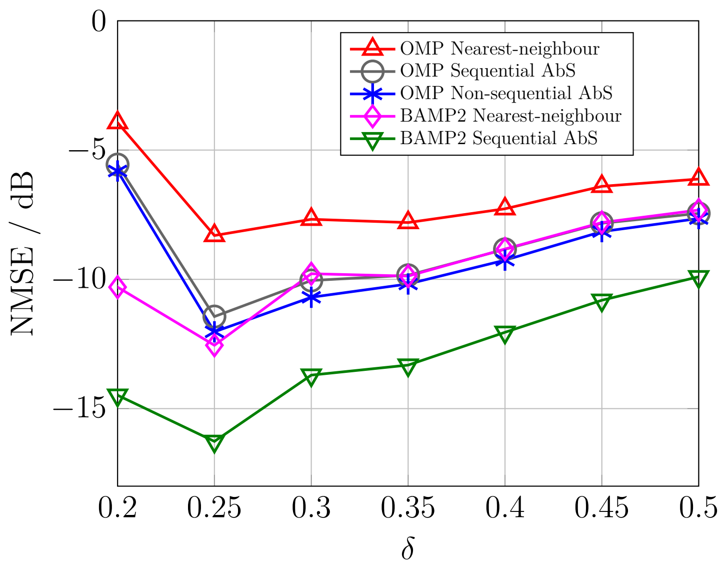


Fig. 4.5 NMSE vs. measurement rate δ ($= m/N$) of AbS with OMP and BAMP2; fixed quantization bit-rate of $r_x = 0.75$ bits per component of the unknown sparse vector (equivalent to a total bit budget of $R_x = 384$ bits). Source vector dimension $N = 512$, sparsity $k = 35$.

respectively. Both figures show that BAMP2 significantly outperforms the OMP algorithm and that the lowest NMSE is achieved for $\delta = 0.25$. Out of all considered measurement rates in my simulation, $\delta = 0.25$ gives the best trade-off between the number of measurements and the quantization bit depth, for a specific total bit budget. Finding the optimal measurement rate for this highly nonlinear problem is a difficult task that goes beyond the scope of this work. In Fig. 4.4, we can see that for $r_x = 0.5$, the gain of using sequential BAMP2 compared to *non*-sequential OMP in AbS is at least 1 dB. Specifically, for $\delta = 0.25$ the gain is roughly 3 dB. Results for $r_x = 0.75$ are shown in Fig. 4.5; the gain of using sequential BAMP2 compared to *non*-sequential OMP in AbS is at least 2 dB, with roughly 4 dB at $\delta = 0.25$. It should also be noted that nearest-neighbor coding with BAMP2 produces roughly the same performance as OMP in the AbS scheme.

For the measurement rate $\delta = 0.25$, Figure 4.6 shows the NMSE of AbS-quantization with OMP and BAMP2 as a function of rate r_x. Two conclusions can be drawn:

- The BAMP2 curves continue to descend beyond the rate of 1 bit per source component. The NMSE will also saturate at some point, as accuracy is then limited by the threshold of 10^{-4} for the BAMP2 iterations.

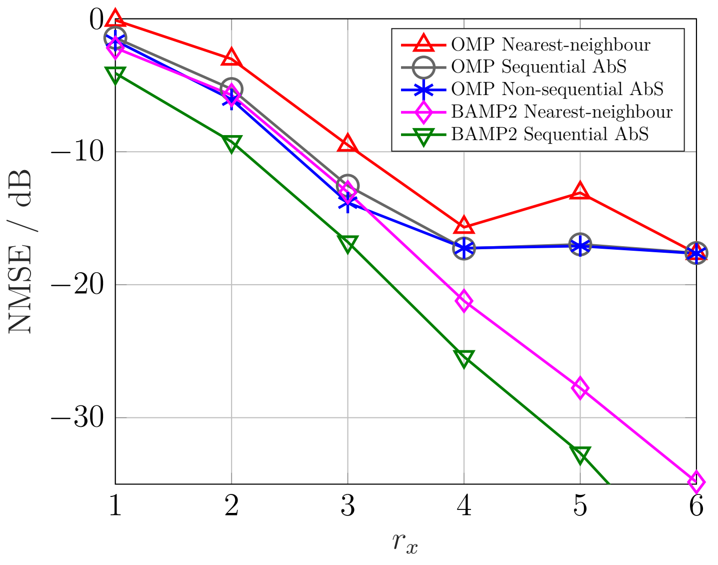


Fig. 4.6 NMSE vs. rate r_x for AbS quantization with OMP and BAMP2. Source vector dimension $N = 512$, sparsity $k = 35$, measurement rate $\delta = 0.25$.

- For rates below 1 bit per sample, the gain of using sequential BAMP2 compared to *non*-sequential OMP in AbS is about 4 dB. This clearly demonstrates the superior performance of BAMP2 that is due to the use of prior information.

Finally it should be noted that OMP is significantly more complex than BAMP2, as OMP requires the computation of a pseudo-inverse during its iterations. Hence, BAMP2 provides significantly better performance at significantly lower complexity.

4.5 Summary

In this chapter, I investigated the use of the BAMP algorithm as the recovery algorithm in the AbS framework for quantization of CS measurements. I focused on the scenario where the bit budget is constrained. In this case, the optimal measurement ratio, and consequently the optimal bit-rate per source component, for specific source prior is impossible to determine analytically. However, for a set of possible bit-rates, one can empirically determine, and later use, the one that shows the lowest NMSE.

My numerical experiments showed that it is preferable to have fewer but finely quantized measurements. Additionally, the results of the experiments demonstrate that the BAMP algorithm significantly outperforms the much more complex OMP algorithm, for both sequential and non-sequential updates in the AbS quantization scheme.

Chapter 5

Generalized Approximate Message Passing for Noisy and Quantized Compressed Sensing

In many practical applications, *compressed sensing* (CS) measurements are first scalar quantized and subsequently corrupted in different ways during storage or transmission over a noisy channel. Reconstruction by conventional CS algorithms on such highly distorted measurements results in poor accuracy. To address this problem, I use the well established *generalized approximate message passing* (GAMP) algorithm and adapt it to our specific problem: recovery of sparse vectors from quantized CS measurements corrupted with noise. I consider two cases:

- 1-bit CS ($R = 1$), where extreme quantization is enforced that captures only the sign of the measurements. Here, I allow for the quantized measurements to be corrupted with *additive white Gaussian noise* (AWGN).

- R-bit CS ($R > 1$), where each measurement is represented with only a few bits. This case, therefore, corresponds to a low-rate quantization scenario. I consider different communication channels tampering with the quantized measurements, namely the *symmetric discrete memoryless* (SDM) channel and the AWGN channel.

I provide analytical expressions for the necessary nonlinear updates of the GAMP algorithm for different channel models and different rates. I conduct numerical experiments and present performance results of the proposed scheme. The results show the superiority of the GAMP algorithm compared to conventional CS reconstruction algorithms for both SDM and AWGN channels.

5.1 Introduction

In many applications, including *magnetic resonance imaging* (MRI) [54, 55, 84, 85], sparse channel estimation [5, 70, 80] or photography [38], a coarse quantization of the CS measurements is unavoidable for further *digital signal processing* (DSP). By quantizating the CS measurements, as continuous input intervals are represented by discrete values, distortion is introduced in the observations. This distortion can be controlled by the size of the codebook; it is reduced by increasing the rate R, i.e., by using more bits per measurement to represent the unknown sparse vector in the measurement domain. Fnding the rate-distortion function for a specific CS recovery algorithm is, however, an even more difficult task compared to the classical (not CS) setting. The reason is that we want to minimize the end-to-end distortion between $\mathbf{x}$ and $\hat{\mathbf{x}} = \hat{\mathbf{x}}(\mathbf{y}) = \hat{\mathbf{x}}(Q(\mathbf{z}))$, which involves a nonlinear measurement system as well as a nonlinear recovery method, while quantizing the measurement vector $\mathbf{y}$. Much of the existing literature, therefore, focuses on finding suboptimal CS recovery algorithms from quantized measurements [13, 14, 23, 41, 45, 46, 48, 49, 51, 76, 78, 83, 86, 87]. This work follows the same line of research. In particular, I use the well established GAMP algorithm and adopt it to our problem: recovery of sparse vectors from *quantized compressed sensing* (QCS) measurements corrupted with noise.

The rest of this chapter is organized as follows. In Section 5.2, I define the problem of noisy QCS and formulate the mathematical model for the unknown sparse signal and the measurement process. In Section 5.3, I provide analytical expressions for the necessary nonlinear updates of the GAMP algorithm. Results of numerical experiments are presented in Section 5.4, and conclusions are drawn in Section 5.5.

5.2 Problem Statement

Next, I formulate the mathematical model for the unknown sparse signal and the measurement process.

5.2.1 Signal model

I assume that the components $\{\mathsf{x}_i\}_{i=1}^{N}$ of the unknown sparse vector $\mathbf{x}$ are *independent and identically distributed* (i.i.d.) realizations of the *Bernoulli-Gauss* (BG) mixture distribution, i.e.,

$$p_{\mathsf{x}_i}(x) = (1-\epsilon)\,\delta(x) + \epsilon\,\mathcal{N}(x\,;0,\sigma_x^2), \tag{5.1}$$

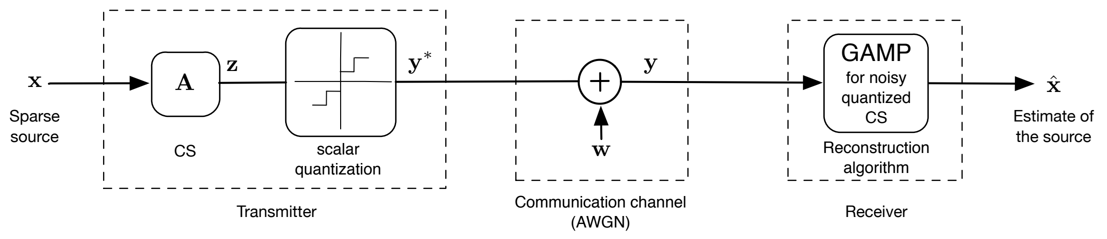


Fig. 5.1 The signal processing chain. A unknown k-sparse vector $\mathbf{x} \in \mathbb{R}^N$ is multiplied with a measurement matrix $\mathbf{A} \in \mathbb{R}^{m \times N}$ to obtain a vector $\mathbf{z} \in \mathbb{R}^m$ of CS measurements. Each component of $\mathbf{y}^*$ represents the quantized version of the respective component in $\mathbf{z}$. Symbols from $\mathbf{y}^*$ are sent through a communication channel to obtain the vector of received measurements $\mathbf{y}$.

where ϵ represents the probability of nonzero value, and σ_x^2 is the power of the Gaussian component. The transmitter from Fig. 5.1 forms a vector of CS measurements (i.e., linear mixtures) $\mathbf{z}$, i.e.,

$$\mathbf{z} = \mathbf{A}\mathbf{x}, \tag{5.2}$$

where $\mathbf{A} \in \mathbb{R}^{m \times N}$ is a Gaussian measurement matrix. The vector of CS measurements $\mathbf{z}$ is scalar quantized with R bits per measurement, using a codebook $\mathcal{C}$. The resulting vector of quantized CS measurements $\mathbf{y}^*$, that is subsequently transmitted over a communication channel, can be expressed as

$$\mathbf{y}^* = \mathcal{Q}(\mathbf{z}) = \mathcal{Q}(\mathbf{A}\mathbf{x}), \tag{5.3}$$

where $\mathcal{Q}(\cdot)$ is the quantization function defined in (2.20).

5.2.2 Measurement model

Noisy AWGN channel

Fig. 5.1 shows the corresponding transmission chain with an AWGN channel. In this case, the measurement vector $\mathbf{y}$ at the receiver, can be compactly expressed as

$$\mathbf{y} = \mathbf{y}^* + \mathbf{w} = \mathcal{Q}(\mathbf{A}\mathbf{x}) + \mathbf{w}\,, \tag{5.4}$$

where $\mathbf{w}$ is a noise vector. The entries of the noise vector are assumed to be *i.i.d.* realizations of a zero-mean Gaussian distribution with variance σ_{w}^2. Furthermore, in the case of 1-bit CS, the quantization function amounts to the sgn$(\cdot)$ function.

	$\mathbb{E}\{\mathsf{z} \mid \mathsf{y} = y\}$	$\mathbb{E}\{\mathsf{z}^2 \mid \mathsf{y} = y\}$	$f_{\mathsf{y}}(y)$ or $p_{\mathsf{y}}(q_k)$
AWGN	$f_{\mathsf{y}}(y)^{-1} \sum_k f_{\mathsf{w}}\big(y - q_k\big) m_k$	$f_{\mathsf{y}}(y)^{-1} \sum_k f_{\mathsf{w}}\big(y - q_k\big) s_k$	$\sum_k f_w(y - q_k) p_k$
SDM	$p_{\mathsf{y}}(q_k)^{-1} \big[p_e 2^{-R}\hat{p} + (1 - p_e) m_k\big]$	$p_{\mathsf{y}}(q_k)^{-1} \big[p_e 2^{-R}(v_p + \hat{p}^2) + (1 - p_e) s_k\big]$	$p_e 2^{-R} + (1 - p_e) p_k$
	$p_k = \Phi(b_{k+1}; \hat{p}, v_p) - \Phi(b_k; \hat{p}, v_p)$; $\quad m_k = \hat{p} p_k - v_p n(b_{k+1}; \hat{p}, v_p) - v_p n(b_k; \hat{p}, v_p)$; $s_k = p_k(\hat{p}^2 + v_p) - v_p \big[(b_{k+1} + \hat{p})\, n(b_{k+1}; \hat{p}, v_p) - (b_k + \hat{p})\, n(b_k; \hat{p}, v_p)\big]$		

Table 5.1 Scalar mean, power, and probability density function for the GAMP nonlinear measurement updates. For the AWGN channel the observation y can take any real value. For the SDM channel, y is constrained to the codebook $\mathcal{I}$.

Noisy SDM channel

In the case of a SDM channel, each measurement y_i is equal to the transmitted symbol y_i^* with probability $1 - (1 - 2^{-R})p_e$. Furthermore, it is equal to any other symbol from the codebook, i.e., $\mathcal{I} \setminus \mathsf{y}_i^*$, with probability $2^{-R}p_e$. Therefore, we can write the distribution of $\mathsf{y}_i \mid \mathsf{y}_i^*$ as

$$\mathsf{y}_i = \begin{cases} \mathsf{y}_i^* = \mathcal{Q}(\mathbf{A}_{i*}\mathbf{x}) & \text{w.p. } 1 - (1 - 2^{-R})p_e, \\ q_{k'} & \text{w.p. } 2^{-R}p_e, \end{cases} \tag{5.5}$$

where $q_{k'}$ is any code symbol from $\mathcal{I} \setminus \mathsf{y}_i^*$.

5.3 GAMP Algorithm for Noisy Quantized Compressed Sensing

To estimate the unknown sparse vector, I use the GAMP algorithm, whose steps are shown in Algorithm 6, Section 3.3. I use the scalar version of this algorithm, where the entries in $\mathbf{v}_s^t$ and $\mathbf{v}_x^t$ are the same within the respective vector. As heuristic experiments showed, this produces more stable implementation of the algorithm.

The source, described in Subsection 5.2.1, is modeled as an i.i.d. random vector, whose components are distributed according to a BG mixture. Therefore, to get the expressions for nonlinear functions $\mathrm{G}_1(\cdot)$ and $\mathrm{G}_2(\cdot)$ in (3.58), I can use the results from Subsection 3.2.2. On the other hand, for nonlinear functions $\mathrm{F}_1(\cdot)$ and $\mathrm{F}_2(\cdot)$ in (3.58), I derive closed-form expressions, respecting the measurement models given by (5.4) and (5.5). The derivation is provided in Appendix B, whereas a summary of the update functions is shown in Table 5.1.

5.4 Numerical Results

To investigate the performance of the proposed reconstruction algorithm I present the results of *Monte-Carlo* (MC) simulations. In the simulations, the nonzero components of the source vector $\mathbf{x}$ as well as the entries of $\mathbf{A}$ are drawn randomly from a standard normal distribution. The columns of the sensing matrix are later normalized to have unit Euclidean norm.
The stopping threshold for the GAMP algorithm is $\varepsilon = 10^{-2}$. The maximal number of iterations of the proposed algorithm for $R = 1$ and $R > 1$, is set to $t_{\max} = 64$ and $t_{\max} = 32$, respectively[1].

5.4.1 1-bit CS

To compare the performances against different 1-bit CS algorithms, the *mean squared error* (MSE), defined as

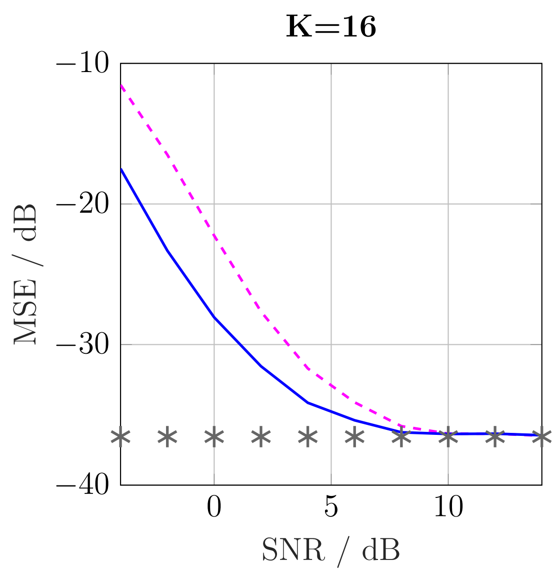
$$\mathrm{MSE/dB} = 10 \log_{10} \left\{ \overline{\left\| \frac{\mathbf{x}}{\|\mathbf{x}\|_2} - \frac{\hat{\mathbf{x}}}{\|\hat{\mathbf{x}}\|_2} \right\|_2^2} \right\}, \tag{5.6}$$

is used as a figure of merit. From (2.24) we know that 1-bit quantization eradicates the information about the l_2-norm of the source. Therefore, in (5.6) I normalize both the source vector and the estimate to have unit l_2-norm.

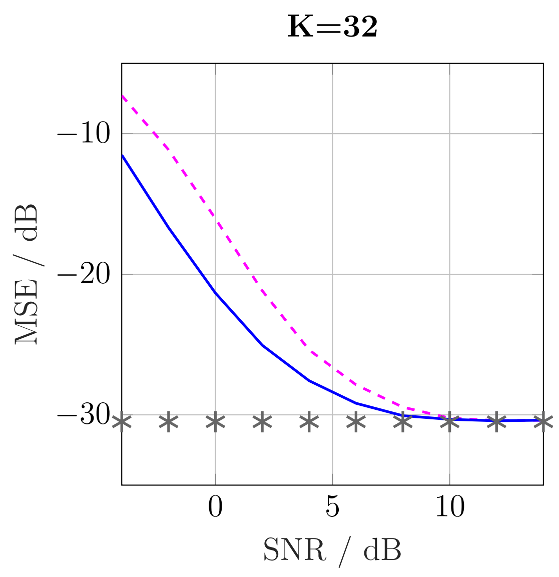
I compute the average empirical MSE over 1000 independent realizations of the source vector, the measurement matrix, and the noise vector. In each simulation, I acquire $m = 2000$ measurements of the underlying sparse vector of length $N = 512$. Each 1-bit CS measurement vector is corrupted with AWGN with power $\sigma_w^2 = 10^{-\mathrm{SNR}/10}$, where the SNR is defined as

$$\mathrm{SNR} = \mathbb{E}_{\mathbf{y}^*}\left\{\|\mathbf{y}^*\|^2\right\} / \mathbb{E}_{\mathbf{w}}\left\{\|\mathbf{w}\|_2^2\right\}. \tag{5.7}$$

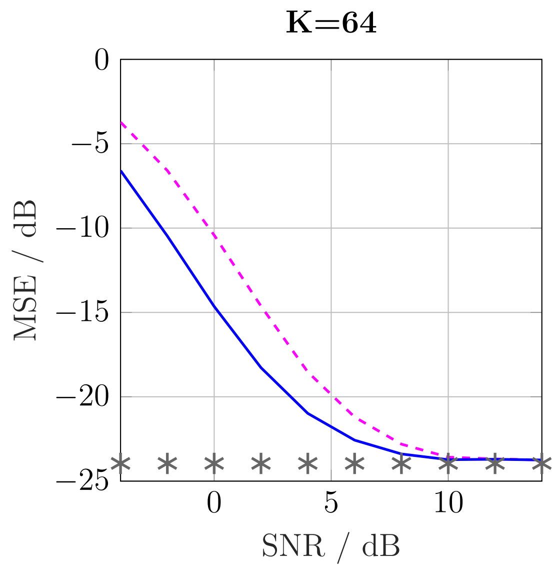
[1]This choice of the parameters gave satisfactory results heuristically.



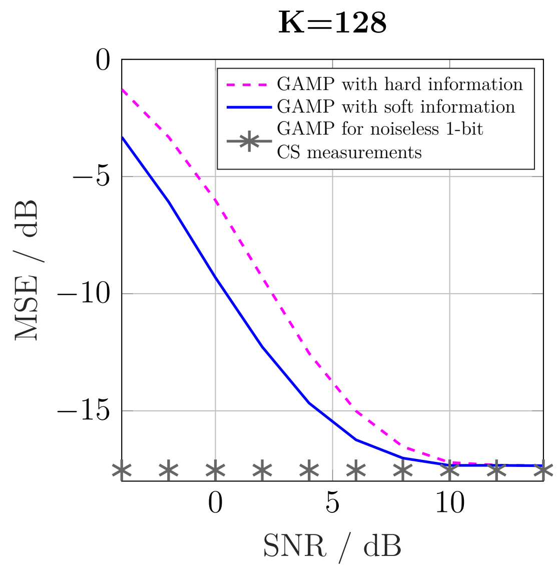
(a)



(b)



(c)



(d)

Fig. 5.2 Performance of different GAMP based algorithms for different sparsity levels k: a) $k = 16$, b) $k = 32$, c) $k = 64$, and d) $k = 128$. The dashed magenta line represents the "robustified" GAMP algorithm for 1-bit CS. The solid blue line represents the GAMP algorithm presented in this paper. The grey stars represent the limit set by the performance of the GAMP algorithm for noiseless 1-bit CS measurements. Parameters: $N = 512$, $m = 2000$.

Results

Since, in the case of 1-bit CS measurements in AWGN, the GAMP algorithm presented in this chapter uses real numbers as inputs, I refer to it as the GAMP algorithm with soft information. On the other hand, in [86] the authors present the GAMP algorithm with the "robustified activation function". They consider different ways corrupting the 1-bit CS measurements by noise: e.g., adding AWGN before quantization; flipping each bit with probability p_e after quantization. In all cases, the GAMP algorithm from [86] operates with symbols form a discrete alphabet, namely signs of the measurements. I, therefore, refer to this algorithms as the GAMP algorithm with hard information. Furthermore, since this algorithm operates on binary measurements, I feed it with quantized measurements $\mathrm{sgn}(\mathbf{y})$. It assumes that each bit was flipped with the probability of p_e, which I set to $p_e = \mathrm{F}(-1/\sqrt{\sigma_w^2})$.

In Fig. 5.2, I compare the performance of the GAMP with soft information with the performance of the GAMP with hard information. We see that within a large SNR range the GAMP algorithm with soft information outperforms (in the MSE sense) the GAMP algorithm that uses hard information. The gain, in terms of MSE, for SNR values below 2 dB is about 5 dB. As expected, for larger SNR values this gain diminishes, and both algorithms approach the limit set by the GAMP algorithm for noiseless 1-bit CS measurements.

5.4.2 R-bit CS

When quantizing with more than one bit per measurement, to validate the recovery potential of the GAMP algorithm, I use MSE, defined as

$$\mathrm{MSE/dB} = 10\log_{10}\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2, \tag{5.8}$$

as the performance metric. I average the MSE over 200 independent instances of the source x, the noise w, and the measurement matrix $\mathbf{A}$. In each simulation, I acquire $m = 512$ CS measurements of the underlying sparse vector of length $N = 512$. Every CS measurement is then scalar quantized with R bits per measurement using a fixed codebook. The code symbols are obtained using Loyd's optimization algorithm [59].

Quantized measurements are corrupted during transmission over both the AWGN channel and the SDM channel. In the case of the AWGN channel, each component of the noise vector is an independent realization of a normal distribution with variance $\sigma_{\mathsf{w}}^2 = 10^{-\mathrm{SNR}/10}$, where the SNR is defined in (5.7). In the case of the SDM channel, I

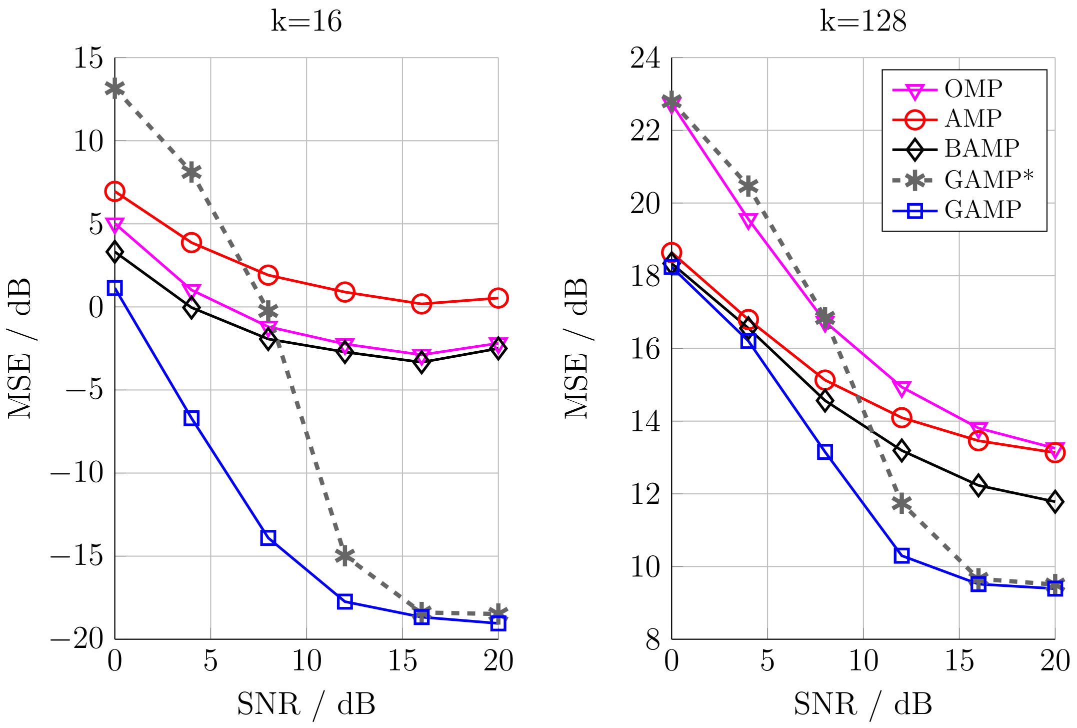


Fig. 5.3 MSE against SNR for different sparsity k in an AWGN channel. GAMP* denotes the GAMP algorithm for a noiseless channel. Parameters: $R = 2$, $m = 512$, $N = 512$.

flip each symbol from the vector of quantized measurements with probability p_e to a symbol from codebook $\mathcal{I}$.

Results

For $k = 16$ and $k = 128$, I conduct three sets of simulations, and present results of different reconstruction algorithms. In each simulation, I compute the empirical MSE of the *orthogonal matching pursuit* (OMP), *approximate message passing* (AMP), *Bayesian approximate message passing* (BAMP), and two GAMP algorithms. The OMP, AMP, and BAMP algorithms are oblivious of the quantization of the CS measurements. Furthermore, among two GAMP algorithms, there is the GAMP and the GAMP* algorithm. The GAMP* algorithm refers the algorithm from [48] that does not take the noisy channel into consideration.

In the first set, I quantize each CS measurement with 2 bits/sample and corrupt the quantized values with AWGN. Fig. 5.3 shows empirical MSE against SNR of different recovery algorithms. We can see that the gain of accounting for the noisy channel in the GAMP algorithm, can be as large as 10 dB for mid-range SNR values. For high

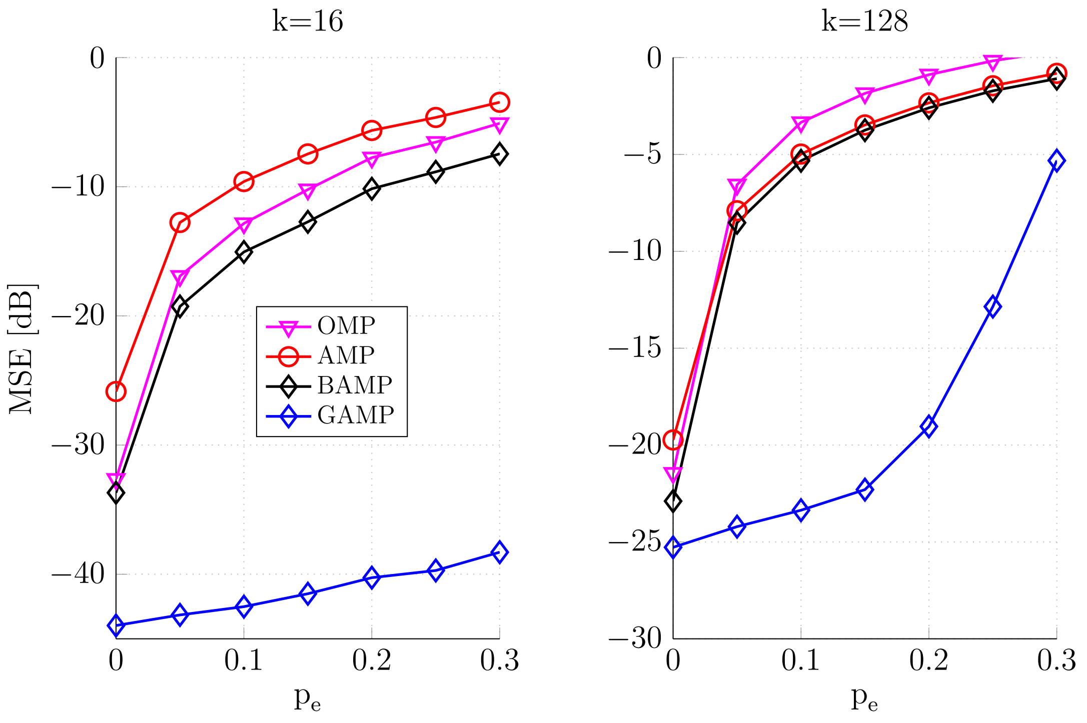


Fig. 5.4 MSE against the probability of the transmission error p_e for different k in a symmetric channel. Parameters: $R = 4$, $m = 512$, and $N = 512$.

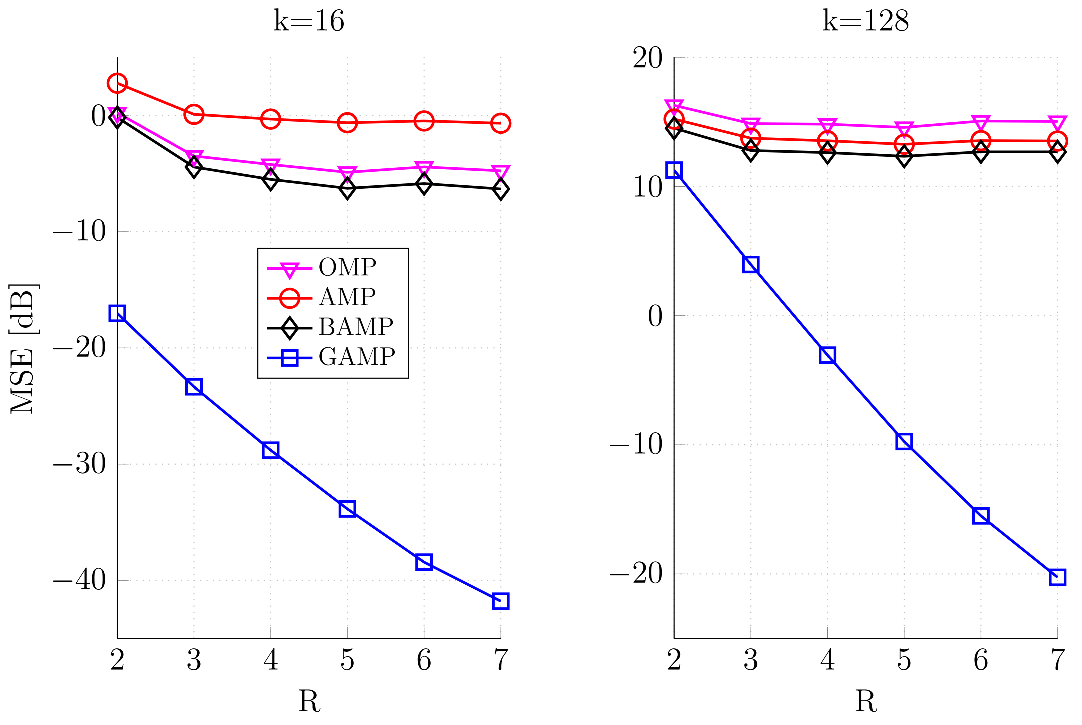


Fig. 5.5 MSE against R for different k in a symmetric channel. Parameters: $p_e = 0.05$, $m = 512$, and $N = 512$.

SNR values, the MSE of the GAMP* algorithm for noiseless quantized CS converges to the MSE of the GAMP for noisy quantized CS.

The second set of simulations corresponds to the scenario of sending quantized CS measurements through an SDM channel. I quantize each measurement with 4 bits/sample and randomly flip each symbol with probability p_e. Fig. 5.3 shows empirical MSE against p_e of different recovery algorithms. We can observe that, in this channel model, the GAMP algorithm significantly outperforms classical CS algorithms. For a low probability of flipping the source symbol, the GAMP algorithm gives at least 10 dB of gain compared to the classical CS algorithms. In a very destructive channel, the algorithms that ignore quantization of the CS measurements fail completely, while the GAMP algorithm still offers low-MSE recovery.

Finally, I fix the flip probability at $p_e = 0.05$ and consider quantizing measurements with different number of bits per measurement, ranging from 2 to 7 bits/measurement. The MSE versus SNR of different recovery algorithms is shown in Fig. 5.5. We observe that the MSE (in dB) of the GAMP algorithm almost linearly decreases as the quantization rate increases. I conclude that unlike other algorithms, the GAMP algorithm makes use of additional bits per measurement.

5.5 Summary

In this chapter, I discussed the performance of the GAMP algorithm for recovering unknown sparse vectors from noisy quantized CS measurements. Numerical results show a superior performance of this algorithm compared to other algorithms from literature.

In the case of 1-bit CS, for low SNR values, the algorithm outperforms the GAMP algorithm using hard information. Furthermore, for high SNR values, the algorithm approaches the limit set by the GAMP algorithm for the noiseless 1-bit CS measurements, without any significant increase in computational complexity.

In the case of R-bit CS, I considered different communication channels, namely the AWGN channel and the SDM channel. For the AWGN channel, my results show that the GAMP algorithm outperforms other algorithms from literature. For the SDM channel, the gain of using the GAMP algorithm is even larger. For a low probability of flipping the source symbol, the GAMP algorithm gives at least 10 dB of gain compared to the classical CS algorithms. Moreover, in a very destructive channel, the numerical experiments show the algorithms that ignore quantization of the CS measurements fail completely, while the GAMP algorithm still offers low-MSE recovery. Finally, unlike other classical CS algorithms, the GAMP algorithm makes use of additional bits per measurement.

Theoretical prediction of the MSE of the GAMP algorithm with the *state evolution* (SE) analysis is an interesting open research problem. Furthermore, this study can be extended with the vector version of the GAMP algorithm.

Chapter 6

Generalized Approximate Message Passing for Unlimited Sampling of CS Measurements

In many practical applications (e.g., image and audio processing, bio-medical applications and analysis of physiological data), the amplitude of the input signal might exceed the sensor's finite dynamic range $[-\lambda, +\lambda]$. In this case, a classical converter would saturate and clip the input, resulting in unwanted distortion. Alternatively, the input can be converted with a *self-reset analog to digital converter* (SR-ADC)). In a SR-ADC, the input samples exceeding the sensor's threshold λ are simply folded back to the interval $[-\lambda, +\lambda]$. The converter is then equivalent to the modulo λ operator.

As a counterpart to the previous chapters, where the focus was on quantization of the *compressed sensing* (CS) measurements, I now consider the problem of clipping the CS measurements. More specifically, I consider the *generalized approximate message passing* (GAMP) algorithm for recovering a sparse signal from modulo samples of its randomized projections. The modulo samples are obtained by a SR-ADC. In contrast to previous work on SR-ADC that consider sparse vectors either in time or frequency domain, I allow for sparse signals in any basis. Furthermore, I also consider a scenario where the randomized projections are sent through a communication channel before being digitizing by a SR-ADC. There, the channel is modeled as an *additive white Gaussian noise* (AWGN) channel. To show the effectiveness of the proposed approach, I conduct *Monte-Carlo* (MC) simulations for both noiseless and noisy case. The results show the ability of the proposed algorithm to fight the nonlinearity of the SR-ADC, as well as the possible additional distortion introduced by the AWGN channel. To the best of my knowledge, this is the first work that examines the effects of SR-ADC on phase transition curves of a CS recovery algorithm.

6.1 Introduction

Shannon's sampling theorem is a fundamental result in signal processing. It states that a continuous bandlimited signal can be perfectly reconstructed from a set of samples taken at a sampling rate proportional to the maximum frequency present in the signal (Nyquist rate) [75]. In the theorem it is assumed that the sampler has infinite precision and infinite dynamic range. This assumption is, however, not met in practical applications as the samples of the input signal might exceed sensor's finite dynamic range. In this case, a classical sampler would saturate and clip the input, resulting in unwanted distortion.

A standard approach to this problem is to attenuate the input, so that the saturation never occurs. Even though a rescaling of the input solves the problem of clipping, it increases the quantization distortion due to a coarser representation of the input. Alternatively, provided that the input is bandlimited, the authors in [7] propose sampling the input signal with a SR-ADC. In a SR-ADC, the input samples exceeding the sensor's threshold are simply folded back to its dynamic range $[-\lambda, +\lambda]$. The converter is then equivalent to the modulo λ operator. More formally, the SR-ADC with the parameter λ is defined by the mapping

$$\mathcal{M}_\lambda(t) = 2\lambda\left(\left[\!\left[\frac{t}{2\lambda} + \frac{1}{2}\right]\!\right] - \frac{1}{2}\right), \tag{6.1}$$

where $[\![t]\!] \triangleq t - \lfloor t \rfloor$ is the remainder of the division t by λ.

In Figure 6.1, I illustrate the effects of digitizing an input signal with a SR-ADC with $\lambda = 0.5$. We can observe that only those values of the received signal that are outside the range $[-0.5, +0.5]$ are affected by the converter. For each of those samples, the converter sums its value with $2k\lambda$, where $k \in \mathbb{Z}$ is chosen such that the sum is in the range$[-\lambda, +\lambda]$.

The authors of [7] prove that it is possible to recover any bandlimited signal from samples taken at regular intervals if:

- the norm of the input signal is known
- the bandwidth of the signal is normalized to π
- the sampling period satisfies $T \leq (2\pi e)^{-1}$

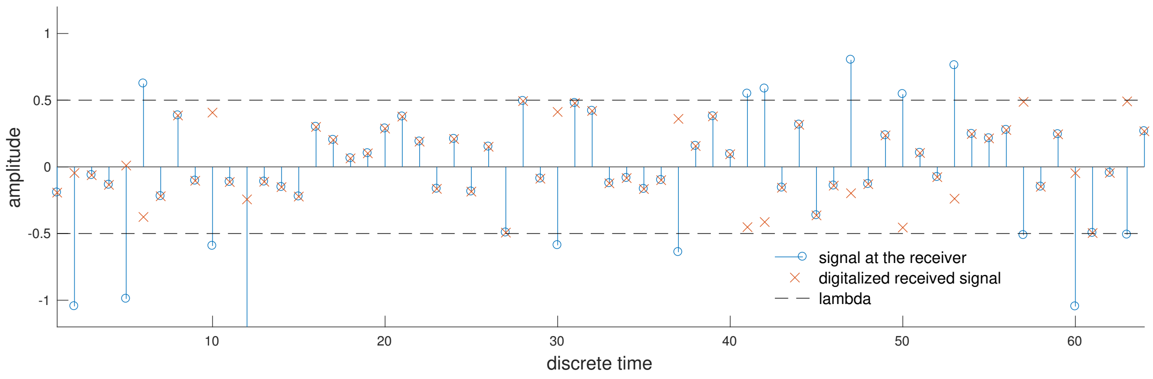


Fig. 6.1 An example of digitizing a signal with SR-ADC, with $\lambda = 0.5$. All the values inside interval $[-\lambda, \lambda]$ are kept undistorted, while the values outside this range are folded back to the interval $[-\lambda, \lambda]$.

Furthermore, apart from giving the sufficient conditions for perfect recovery, the authors present a stable recovery algorithm.

The problem of recovering signals from clipped measurements appears also when sampling certain sparse signals [2, 8, 9, 47]. In [8], two examples of a practical application are shown where the measurements of a sparse signal are clipped, namely ultra-wide band sensing and ultrasonic non-destructive testing. It was observed that during the calibration phase, the peaks of the amplitude are usually larger compared to those in the subsequent sensing phase. In those cases, a classical sampler will saturate and clip the measurements. On the other hand, the authors in [8] consider taking modulo measurements of the low-pass filtered k-sparse signal. The recovery of the sparse signal is then based on a two step approach:

1. The authors capitalize on their results from [7], to recover the low-pass representation of the sparse signal from modulo measurements.

2. Using the results from [11, 52, 81], the sparse signal is perfectly recovered from its low-pass projection.

The authors provide sufficient conditions for perfect recovery of the sparse signal, together with a constructive recovery algorithm.

In this chapter, I follow the work of [8], but instead of sampling a low-pass filtered version of a sparse signal, I consider taking CS measurements and digitizing them with a SR-ADC. This way we can sample signals that are sparse not only in time domain, but also in some other domains, e.g., wavelet domain. The class of signals I consider is, therefore, much broader then in [8]. Furthermore, I consider the case where the

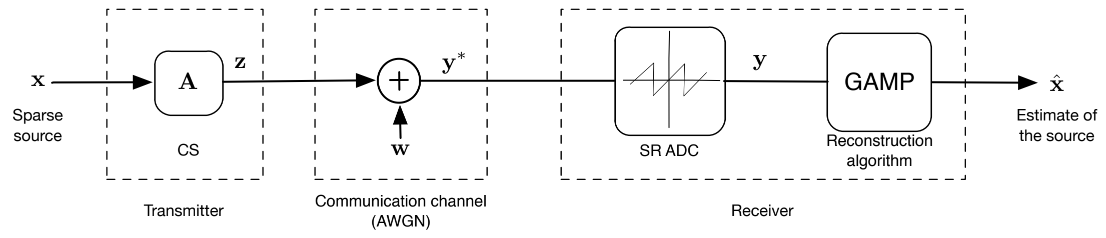


Fig. 6.2 The signal processing chain. The unknown k-sparse vector $\mathbf{x} \in \mathbb{R}^N$ is multiplied with measurement matrix $\mathbf{A}_{m\times N}$ to obtain a vector of CS measurements $\mathbf{z} \in \mathbb{R}^m$. The components of $\mathbf{z}$ are transmitted through an AWGN channel. At the receiver, the samples of the received signal $\mathbf{y}^*$ are digitized with a SR-ADC to obtain the vector of measurements $\mathbf{y}$. The GAMP algorithm is applied to produce an estimate $\hat{\mathbf{x}}$ of the unknown sparse signal $\mathbf{x}$.

measurements are possibly corrupted with noise. A possible application scenario is shown in Fig. 6.2, where a sparse signal is to be communicated to a receiver. To reduce traffic over the channel, we begin by constructing a vector of CS measurements of a sparse signal. That message vector is then transmitted over an AWGN channel and digitized at the receiver with a SR-ADC. Since the GAMP algorithm [71] was already successfully applied for recovery of sparse signals from CS measurements with nonlinear distortions [48, 49, 65, 66, 71, 86], I employ it as the recovery algorithm in our problem as well.

The rest of this chapter is organized as follows. In Section 6.2, I define the problem of recovering a sparse signal from, possibly noisy, modulo samples of CS measurements. I propose solving the problem with the GAMP algorithm. Therefore, in Section 6.3, I provide analytical expressions for the necessary nonlinear updates of the GAMP algorithm. Results of numerical experiments are presented in Section 6.4, and conclusions are drawn in Section 6.5.

6.2 Problem Statement

Next, I formulate the mathematical model for the unknown sparse signal and the measurement process.

6.2.1 Signal model

I assume that the components $\{\mathsf{x}_i\}_{i=1}^N$ of the unknown sparse vector $\mathbf{x}$ are *independent and identically distributed* (i.i.d.) realizations of the *Bernoulli-Gauss* (BG) mixture distribution, i.e.,

$$p_{\mathsf{x}_i}(x) = (1-\epsilon)\,\delta(x) + \epsilon\,\mathcal{N}(x\,;0,\sigma_x^2), \tag{6.2}$$

where ϵ represents the probability of nonzero value, and σ_x^2 is the power of the Gaussian component. I assume that the transmitter from Fig. 6.2 transmits a vector of CS linear mixtures $\mathbf{z}$, i.e.,

$$\mathbf{z} = \mathbf{A}\mathbf{x}, \tag{6.3}$$

where $\mathbf{A} \in \mathbb{R}^{m\times N}$ is a Gaussian measurement matrix. As before, the ratio of the number of transmitted symbols (rows) and the ambient dimension of the source vector N defines the sampling rate (indeterminacy) $\rho = m/N$.

6.2.2 Measurement model

I assume that the vector of CS measurements $\mathbf{z}$ is sent over an AWGN channel. Therefore, the signal at the receiver $\mathbf{y}^*$ can be written as

$$\mathbf{y}^* = \mathbf{z} + \mathbf{w} = \mathbf{A}\mathbf{x} + \mathbf{w}, \tag{6.4}$$

where $\mathbf{w}$ is i.i.d. zero-mean AWGN noise vector with the covariance matrix $\sigma_w^2\,\mathbf{I}$, i.e., $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \sigma_w^2\, I)$. Subsequently, each measurement y_i is a folded version of the i-th component of the received signal $\mathbf{y}^*$, i.e.,

$$\mathsf{y}_i = \mathcal{M}_\lambda(\mathsf{y}_i^*), \tag{6.5}$$

where $\mathcal{M}_\lambda(\cdot)$ represents the nonlinear mapping of the SR-ADC converter given in (6.1). I can compactly write the entire measurement process as

$$\mathbf{y} = \mathcal{M}_\lambda\Big(\mathbf{A}\mathbf{x} + \mathbf{w}\Big). \tag{6.6}$$

I note that the involved SR-ADC has infinite precision in the interval $[-\lambda, \lambda]$. Furthermore, for later discussions it is important to define the so-called simple function. Since $\mathbf{y}$ is a distorted version of $\mathbf{y}^*$, one could model that distortion as additive noise, i.e.,

$$\mathbf{y} = \mathbf{y}^* + \boldsymbol{\epsilon}_g. \tag{6.7}$$

The noise vector $\boldsymbol{\epsilon}_g$ in (6.7) is called the simple function and its entries belong to a set of discrete points $2\lambda\,\mathbb{Z}$.

In the following section, I show how to estimate $\mathbf{x}$ from $\mathbf{y}$ using the GAMP algorithm.

6.3 GAMP Algorithm for SR ADC

The general steps of the GAMP algorithm are given in Section 3.3, Algorithm 6. Respecting the signal model in (6.2), the algorithm is initialized according to

$$w\hat{\mathbf{x}}^0 = \mathbb{E}\{\mathbf{x}\} = 0, \ \ \mathbf{v}_{\mathbf{x}}^0 = \operatorname{var}\{\mathbf{x}\} = \epsilon\,\sigma_x^2, \ \ \hat{\mathbf{s}}^0 = \mathbf{y}. \tag{6.8}$$

Moreover, when implementing the algorithm for a specific problem, one needs to find analytical expressions for the nonlinear steps in (3.58), namely

$$\begin{aligned} \mathrm{F}_1(y,\hat{p},v_p) &= \frac{\mathbb{E}\{\mathsf{z}\,|\,y\} - \hat{p}}{v_p}, & \mathrm{G}_1(\hat{r},v_r;\,p_x) &= \mathbb{E}\{\mathsf{x}\,|\,\hat{r}\}, \\ \mathrm{F}_2(y,\hat{p},v_p) &= \frac{v_p - \operatorname{var}\{\mathsf{z}\,|\,y\}}{v_p^2}, & \mathrm{G}_2(\hat{r},v_r;\,p_x) &= \operatorname{var}\{\mathsf{x}\,|\,\hat{r}\}, \end{aligned} \tag{6.9}$$

where $\mathsf{z} \sim \mathcal{N}(\hat{p}, v_p)$, and $\mathsf{x} \sim \mathcal{N}(\hat{r}, v_r)$. Since the source is modeled as an i.i.d. random vector whose components are distributed according to a BG mixture, to get the expressions for nonlinear functions $\mathrm{G}_1(\cdot)$ and $\mathrm{G}_2(\cdot)$ in (6.9), I can use the results from Subsection 3.2.2. On the other hand, for nonlinear functions $\mathrm{F}_1(\cdot)$ and $\mathrm{F}_2(\cdot)$, in what follows, I derive analytical expressions considering the noisy channel model, given in (6.6). The derivation of those functions for the noiseless channel can be found in Appendix B.3.

It is worth noting that, since the expressions in (6.9) involve calculating means and variances, one could alternatively resort to numerical methods for approximating those terms.

6.3.1 Nonlinear steps for the AWGN channel and SR ADC

Here I assume that the vector of linear mixtures $\mathbf{z}$ is corrupted with AWGN with power σ_w^2 during the transmission (Fig. 6.2). At the receiver, the input signal is sampled with a SR-ADC with threshold parameter λ. For $y \in [-\lambda, \lambda]$, the conditional distribution $f_{\mathsf{y}}(y\,|\,z)$ is calculated as

$$\begin{aligned} f_{\mathsf{y}}(y\,|\,z) &= \int_{-\infty}^{+\infty} f_{\mathsf{y},\mathsf{y}^*}(y, y^*\,|\,z)\,dy^* = \int_{-\infty}^{+\infty} f_{\mathsf{y}^*}(y^*\,|\,z)\,f_{\mathsf{y}}(y\,|\,y^*, z)\,dy^* \\ &\overset{(a)}{=} \int_{-\infty}^{+\infty} \mathcal{N}(y^*\,;z,\sigma_w^2)\,f_{\mathsf{y}}(y\,|\,y^*)\,dy^* \\ &= \int_{-\infty}^{+\infty} \mathcal{N}(y^*\,;z,\sigma_w^2)\,\delta(y - \mathcal{M}_\lambda(y^*))\,dy^* = \sum_{k=-\infty}^{\infty} \mathcal{N}(y + 2k\lambda\,;z,\sigma_w^2), \end{aligned} \tag{6.10}$$

where in $\overset{(a)}{=}$ I use the fact that $\mathsf{y}^* \,|\, z \sim \mathcal{N}(z, \sigma_w^2)$, and the fact that y and z are statistically independent given y^*. Outside of the interval $[-\lambda, \lambda]$, the conditional distribution $f_\mathsf{y}(y \,|\, z) = 0$. Using Bayes' rule we can write the distribution $f_\mathsf{z}(z \,|\, y)$ as

$$\begin{aligned} f_\mathsf{z}(z \,|\, y) &= \frac{1}{f_\mathsf{y}(y)} f_\mathsf{y}(y \,|\, z)\, f_\mathsf{z}(z) = \frac{1}{f_\mathsf{y}(y)} \sum_{k=-\infty}^{\infty} \mathcal{N}(y + 2k\lambda\,; z, \sigma_w^2)\, \mathcal{N}(z\,; \mu_z, \sigma_z^2) \\ &= \frac{1}{f_\mathsf{y}(y)} \sum_{k=-\infty}^{\infty} \mathcal{N}(z\,; y + 2k\lambda, \sigma_w^2)\, \mathcal{N}(z\,; \mu_z, \sigma_z^2). \end{aligned} \tag{6.11}$$

The sum in (6.11) involves a product of two Gaussian distributions. In [1], it is shown that a product of two Gaussian distributions $f_\mathsf{x}(x)$ and $g_\mathsf{x}(x)$, with arbitrary means μ_f and μ_g and arbitrary variances σ_f^2 and σ_g^2, can be written as a scaled Gaussian distribution, i.e.,

$$f_\mathsf{x}(x) g_\mathsf{x}(x) = \frac{S_{fg}}{\sqrt{2\pi\sigma_{fg}^2}} \exp\left[-\frac{(x-\mu_{fg})^2}{2\,\sigma_{fg}^2}\right], \tag{6.12}$$

with

$$\sigma_{fg}^2 = \frac{\sigma_f^2 \sigma_g^2}{\sigma_f^2 + \sigma_g^2}, \quad \mu_{fg} = \frac{\mu_f \sigma_g^2 + \mu_g \sigma_f^2}{\sigma_f^2 + \sigma_g^2},$$

and the scaling factor S_{fg} having a form of a Gaussian distribution

$$S_{fg} = \frac{1}{\sqrt{2\pi(\sigma_f^2 + \sigma_g^2)}} \exp\left[-\frac{(\mu_f - \mu_g)^2}{2(\sigma_f^2 + \sigma_g^2)}\right]. \tag{6.13}$$

Therefore, the conditional distribution $f(z \,|\, y)$ in (6.11) can be written as

$$\begin{aligned} f_\mathsf{z}(z \,|\, y) &= \frac{1}{f_\mathsf{y}(y)} \sum_{k=-\infty}^{\infty} \mathcal{N}(z\,; y + 2k\lambda, \sigma_w^2)\, \mathcal{N}(z\,; \mu_z, \sigma_z^2) \\ &= \frac{1}{f_\mathsf{y}(y)} \sum_{k=-\infty}^{\infty} \mathcal{N}(0\,; y + 2k\lambda - \mu_z, \sigma_z^2 + \sigma_w^2)\, \mathcal{N}(z\,; \mu_{wz}, \sigma_{wz}^2) \\ &= \frac{1}{f_\mathsf{y}(y)} \sum_{k=-\infty}^{\infty} \gamma_k\, \mathcal{N}(z\,; \mu_{wz}, \sigma_{wz}^2), \end{aligned} \tag{6.14}$$

where

$$\begin{aligned} \sigma_{wz}^2 &= \frac{\sigma_w^2 \sigma_z^2}{\sigma_w^2 + \sigma_z^2}, \\ \mu_{wz} &= \left(\frac{y + 2k\lambda}{\sigma_w^2} + \frac{\mu_z}{\sigma_z^2}\right)\sigma_{wz}^2, \\ \gamma_k &= \mathcal{N}(0\,; y + 2k\lambda - \mu_z, \sigma_z^2 + \sigma_w^2). \end{aligned} \tag{6.15}$$

In (6.14), the term $p_{\mathsf{y}}(y)$ is a normalizing term that ensures that the conditional distribution $p_{\mathsf{z}}(z \,|\, y)$ integrates to 1. Therefore, it can be calculated as

$$\begin{aligned} f_{\mathsf{y}}(y) &= \int_{-\infty}^{+\infty} \sum_{k=-\infty}^{\infty} \gamma_k \, \mathcal{N}(z\,;\mu_{wz}, \sigma_{wz}^2)\, dz = \sum_{k=-\infty}^{\infty} \int_{-\infty}^{+\infty} \gamma_k \, \mathcal{N}(z\,;\mu_{wz}, \sigma_{wz}^2)\, dz \\ &= \sum_{k=-\infty}^{\infty} \gamma_k \int_{-\infty}^{+\infty} \mathcal{N}(z\,;\mu_{wz}, \sigma_{wz}^2)\, dz = \sum_{k=-\infty}^{\infty} \gamma_k. \end{aligned} \tag{6.16}$$

Similarly, $\mathbb{E}\{\mathsf{z} \,|\, y\}$ and $\mathbb{E}\{\mathsf{z}^2 \,|\, y\}$ can be calculated as

$$\begin{aligned} \mathbb{E}\{\mathsf{z} \,|\, y\} &= \int_{-\infty}^{\infty} z \, f_{\mathsf{z}}(z \,|\, y)\, dz = \int_{-\infty}^{\infty} z \, \frac{1}{f(y)} \sum_{k=-\infty}^{\infty} \gamma_k \, \mathcal{N}(z\,;\mu_{wz}, \sigma_{wz}^2)\, dz \\ &= \frac{1}{f(y)} \sum_{k=-\infty}^{\infty} \gamma_k \int_{-\infty}^{\infty} z \, \mathcal{N}(z\,;\mu_{wz}, \sigma_{wz}^2)\, dz = \frac{1}{f(y)} \sum_{k=-\infty}^{\infty} \gamma_k \, \mu_{wz}, \\ \mathbb{E}\{\mathsf{z}^2 \,|\, y\} &= \int_{-\infty}^{\infty} z^2 \, f_{\mathsf{z}}(z \,|\, y)\, dz = \int_{-\infty}^{\infty} z^2 \, \frac{1}{f(y)} \sum_{k=-\infty}^{\infty} \gamma_k \, \mathcal{N}(z\,;\mu_{wz}, \sigma_{wz}^2)\, dz \\ &= \frac{1}{f(y)} \sum_{k=-\infty}^{\infty} \gamma_k \int_{-\infty}^{\infty} z^2 \, \mathcal{N}(z\,;\mu_{wz}, \sigma_{wz}^2)\, dz = \frac{1}{f(y)} \sum_{k=-\infty}^{\infty} \gamma_k \, (\sigma_{wz}^2 + \mu_{wz}^2). \end{aligned} \tag{6.17}$$

Finally, the variance $\mathrm{var}\{\mathsf{z} \,|\, y\}$ is obtained as

$$\mathrm{var}\{\mathsf{z} \,|\, y\} = \mathbb{E}\{\mathsf{z}^2 \,|\, y\} - \big(\mathbb{E}\{\mathsf{z} \,|\, y\}\big)^2. \tag{6.18}$$

6.4 Numerical Results

To investigate the performance of the proposed reconstruction algorithm I present the results of MC simulations. In the simulations, the nonzero components of the source vector $\mathbf{x}$ as well as the entries of $\mathbf{A}$ are drawn randomly from a standard normal distribution. The columns of the sensing matrix are later normalized to have unit Euclidean norm. Each MC simulation corresponds to a fixed pair of the measurement ratio ρ and the probability of nonzero value ϵ. I set the length N of the sparse vector $\mathbf{x}$ to 256, and acquire n CS measurements $\{z_i\}_{i=1}^{n}$ of a k-sparse vector, where

$$n = \rho\, N, \quad \text{and} \quad k = \epsilon\, N. \tag{6.19}$$

Subsequently, the vector of CS measurement $\mathbf{z}$ is corrupted with an independent realization of AWGN noise vector $\mathbf{w}$. The power of the noise is $\sigma_w^2 = 10^{-\mathrm{SNR}/10}$, where the SNR is defined as

$$\mathrm{SNR} = \mathbb{E}_{\mathbf{y}^*}\left\{\|\mathbf{y}^*\|^2\right\}/\mathbb{E}_{\mathbf{w}}\left\{\|\mathbf{w}\|_2^2\right\}. \tag{6.20}$$

In the noiseless case, I simply set SNR $= \infty$. The SR-ADC threshold λ is fixed to 1. To validate the recovery potential of the GAMP algorithm, I use *mean squared error* (MSE), defined as

$$\mathrm{MSE/dB} = 10\log_{10}\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2, \tag{6.21}$$

as the performance metric. To compute the MSE for a specific pair (ρ, ϵ), I average results over 4000 independent realizations of the source vector, the measurement matrix, and the noise vector. In the noiseless case, my simulations showed that the algorithm either recovers the unknown vector almost perfectly (with very small MSE $\leq$ - 40dB), or fails completely. Therefore, in this case, the average number of successful recoveries (i.e., success rate), is used as the performance metric. I consider a recovery to be successful if the resulting MSE is ≤ -30dB.

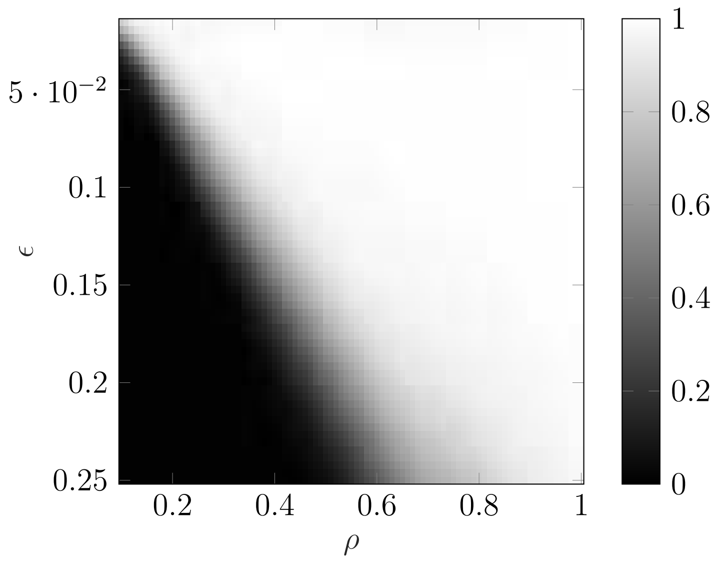
The stopping threshold for the algorithms is $\varepsilon = 10^{-3}$, where as the maximal number of iterations of the proposed algorithm is set to $t_{\max} = N/2 = 128$.

6.4.1 Recovery from noiseless modulo measurements

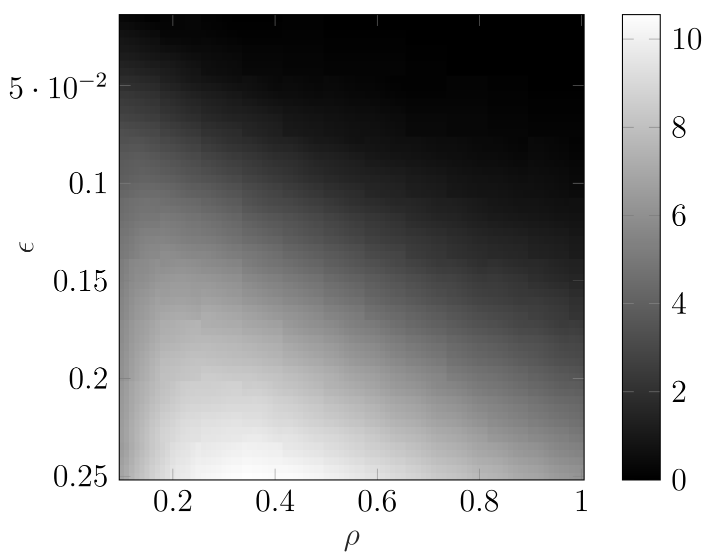
In Fig. 6.3, I show the success rate of the GAMP algorithm (Fig. 6.3a) and the average norm of the simple function $\|\boldsymbol{\epsilon}_g\|_0$ (Fig. 6.3b), both as a function of the measurement ratio ρ and the nonzero probability ϵ. The norm of the simple function provides a measure of how corrupted the measurements are, due to the signal acquisition using a SR-ADC. In Fig. 6.3a, we see a clear phase transition between unsuccessful (black) and successful (white) regions. While classical CS algorithms fail completely when $\|\boldsymbol{\epsilon}_g\|_0 \neq 0$, I observe that GAMP is able to cope with folded measurements. Moreover, for the considered values of ρ and ϵ, the phase transition curve is almost a linear function.

6.4.2 Recovery from noisy modulo measurements

In Fig. **??**, I show the MSE of the GAMP algorithm (Fig. **??**) and the average norm of the simple function $\|\boldsymbol{\epsilon}_g\|_0$ (Fig. **??**), both as a function of the measurement ratio ρ and the nonzero probability ϵ. In Fig. **??**, we observe that, compared to the noiseless case, the phase transition curve is shifted the right lower corner. This is to be expected, since the measurements are corrupted with AWGN (SNR = 20dB) before digitization, and more measurements are needed for accurate reconstruction.

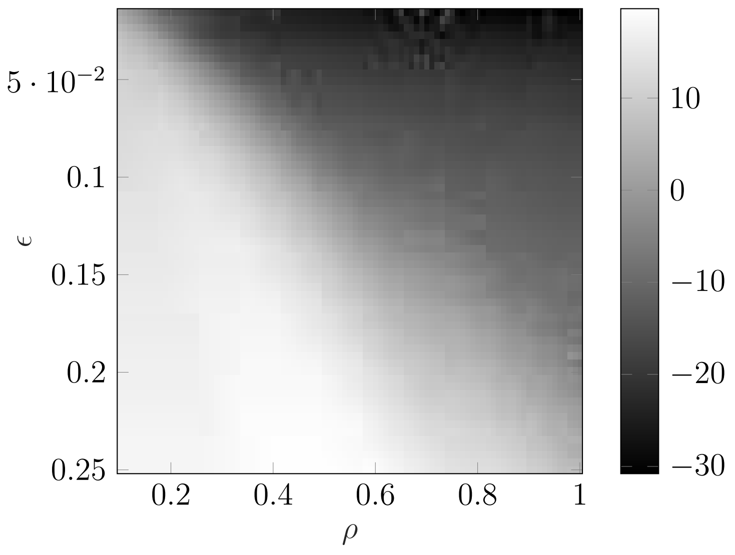


(a) Average success rate of GAMP

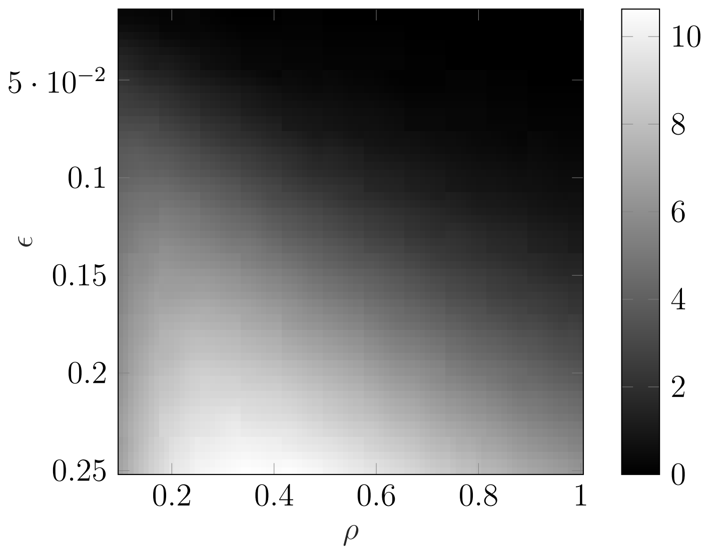


(b) Average norm of the simple function $\|\boldsymbol{\epsilon}_g\|_0$

Fig. 6.3 Average success rate of GAMP reconstruction algorithm on the left, and average norm of the simple function $\|\boldsymbol{\epsilon}_g\|_0$ on the right as a function of the nonzero probability ϵ and the measurement ratio ρ. The CS measurements are digitized with a SR-ADC with $\lambda = 1$.



(a) Average MSE in dB of GAMP



(b) Average norm of the simple function $\|\boldsymbol{\epsilon}_g\|_0$

Fig. 6.4 Average MSE in dB of GAMP reconstruction algorithm on the left, and average norm of the simple function $\|\boldsymbol{\epsilon}_g\|_0$ on the right as a function of the nonzero probability ϵ and the measurement ratio ρ. The CS measurements are corrupted with AWGN noise before being digitized with a SR-ADC with $\lambda = 1$. The SNR is set to 20dB.

6.5 Summary

I investigated the potential of applying the GAMP algorithm for recovery of a sparse signal from CS measurements digitized with a SR-ADC. Additionally, in contrast to the previous work on SR-ADC, I considered a scenario where the CS measurements (i.e., randomized projections) are sent through a communication channel, before being quantized by a SR-ADC. The channel is modeled as an AWGN channel.

To show the effectiveness of the proposed approach, I conducted MC simulations for both noiseless and noisy cases. The results of the numerical experiments show that for a certain set of problems, the GAMP algorithm is able to successfully recover a sparse signal from folded measurements, while classical CS algorithms fail completely. Moreover, unlike the previously proposed algorithm for recovery of sparse signals from folded measurements, the GAMP algorithm can cope with the noise introduced by a communication channel.

Having smaller and smaller λ makes measurements less and less informative. In the limit $\lambda \to 0$ the measurements carry no information. However, in practical scenarios with finite bit-budget per sample, too large λ leads to a coarse quantization. Therefore, one needs to make a good trade-off between large dynamic range and fine quantization resolution. It is an interesting research problem to investigate the effects of folding combined with a finite bit budget quantization of the folded measurements on the phase transition curves of the GAMP algorithm.

Chapter 7

Conclusions

I this thesis, I studied practical challenges of the practical application of the digital sensors of the future. The main aim of the thesis is to find robust and fast algorithms for sampling, quantization, compression, dequantization and recovery of the input signals. If the signal to be estimated is sparse and high dimensional, a novel *digital signal processing* (DSP) technique, called *compressed sensing* (CS), allows efficient recovery from (possibly noisy) compressed representation. In my thesis, I focus on two most prominent challenges of the practical application of CS, namely quantization of the CS measurements and miscalibrated sensors.

Regardless of the practical problem at consideration, i.e., quantization or miscalibration, the core task is, nonetheless, the recovery of a sparse signal from a noisy measurement. Among many CS recovery algorithms, the class of *approximate message passing* (AMP) algorithms stands out as the one with most potential for solving inverse problems involving quantized CS measurements. As my own contribution, in Chapter 3, I assume that the source prior consists of a weighted average of n-Gaussian distributions, each with potentially different mean and different variance, and derive closed-form expressions for the denoiser functions of the *Bayesian approximate message passing* (BAMP) algorithm. By choosing appropriate values for the means and the variances, one can model many practically interesting priors. Moreover, by picking a very small but still non-zero variance, one can even approximate discrete *probability mass functions* (pmfs).

In Chapter 4, I investigated the use of the BAMP algorithm as the recovery algorithm in the *Analysis-by-Synthesis* (AbS) framework for quantization of CS measurements. I focused on the scenario where the bit budget is constrained. My numerical experiments showed that it is preferable to have fewer but finely quantized measurements. Additionally, the results of the experiments demonstrate that the BAMP algorithm significantly outperforms the much more complex *orthogonal matching pursuit* (OMP) algorithm, when used in the AbS framework.

In Chapter 5, I studied the performance of the *generalized approximate message passing* (GAMP) algorithm for recovering unknown sparse vectors from noisy quantized CS measurements. I provide analytical expressions for the necessary nonlinear updates of the GAMP algorithm for different channel models and different rates. Numerical results show a superior performance of this algorithm compared to other algorithms from literature, for both, the *additive white Gaussian noise* (AWGN) channel and the *symmetric discrete memoryless* (SDM) channel.

Theoretical prediction of the *mean squared error* (MSE) of the GAMP algorithm with the *state evolution* (SE) analysis is an interesting open research problem. Furthermore, this study can be extended with the vector version of the GAMP algorithm.

Finally, in Chapter 6, I studied the problem of sampling signals using miscalibrated sensors, within the CS framework. In particular, I investigated the potential of applying the GAMP algorithm for recovery of a sparse signal from CS measurements digitized with a *self-reset analog to digital converter* (SR-ADC). Additionally, in contrast to the previous work on SR-ADC, I considered a scenario where the CS measurements (i.e., randomized projections) are sent through a communication channel, before being quantized by a SR-ADC. The channel is modeled as an AWGN channel. The results of the numerical experiments show that for a certain set of problems, the GAMP algorithm is able to successfully recover a sparse signal from folded measurements, while classical CS algorithms fail completely. Moreover, unlike the previously proposed algorithm for recovery of sparse signals from folded measurements, the GAMP algorithm can cope with the noise introduced by a communication channel.

It is an interesting research problem to investigate the effects of folding combined with a finite bit budget quantization of the folded measurements on the phase transition curves of the GAMP algorithm.

Appendix A

Estimation Updates of the GAMP Algorithm

A.1 Estimation Update for a Bernoulli-Gauss Mixture Prior

We assume that he overall prior can be written as

$$p_{\mathsf{x}_j}(x_j) = \gamma \underbrace{\sum_{m=1}^{N} \epsilon_m \delta(x_j - b_m)}_{p_D(x_j)} + (1-\gamma) \underbrace{\mathcal{N}(x_j;\, 0, \sigma^2)}_{p_C(x_j)} = \gamma p_D(x_j) + (1-\gamma) p_C(x_j)\ , \quad \text{(A.1)}$$

where $\sum_{m=1}^{N} \epsilon_m = 1$, and $\epsilon_m \geq 0$. Form (3.31), it follows that

$$\begin{aligned}
I_0(u_j; c) &= \gamma \int_{-\infty}^{+\infty} e^{-\frac{(x_j-u_j)^2}{2c}}\, p'_D(x_j)\, dx_j + (1-\gamma) \int_{-\infty}^{+\infty} e^{-\frac{(x_j-u_j)^2}{2c}}\, p'_C(x_j)\, dx_j, \\
I_1(u_j; c) &= \gamma \int_{-\infty}^{+\infty} x_j\, e^{-\frac{(x_j-u_j)^2}{2c}}\, p'_D(x_j)\, dx_j + (1-\gamma) \int_{-\infty}^{+\infty} x_j\, e^{-\frac{(x_j-u_j)^2}{2c}}\, p'_C(x_j)\, dx_j, \\
I_2(u_j; c) &= \gamma \int_{-\infty}^{+\infty} \operatorname{erf}\left(\frac{x_j - u_j}{\sqrt{2c}}\right) p'_D(x_j)\, dx_j + (1-\gamma) \int_{-\infty}^{+\infty} \operatorname{erf}\left(\frac{x_j - u_j}{\sqrt{2c}}\right) p'_C(x_j)\, dx_j.
\end{aligned} \quad \text{(A.2)}$$

It can be shown that the contributions of discrete part of the prior to each of the integrals above is

$$\begin{aligned}
I_{0D}(u_j;c) &= -\frac{\gamma}{c}\sum_{m=1}^{N}\epsilon_m(u_j-b_m)\,e^{-\frac{(u_j-b_m)^2}{2c}} \quad ,\\
I_{1D}(u_j;c) &= -\gamma\sum_{m=1}^{N}\epsilon_m\,e^{-\frac{(u_j-b_m)^2}{2c}}\left(1+\frac{b_m}{c}u_j-\frac{b_m^2}{c}\right) \quad ,\\
I_{2D}(u_j;c) &= -\gamma\sqrt{\frac{2}{\pi c}}\sum_{m=1}^{N}\epsilon_m\,e^{-\frac{(u_j-b_m)^2}{2c}} \quad .
\end{aligned} \tag{A.3}$$

What is left is to compute the contributions of the continuous part of the prior, namely $I_{0C}(u_j;c)$, $I_{1C}(u_j;c)$ and $I_{2C}(u_j;c)$. To compute each of these integrals we will again use integration by parts as follows.

Computation of the integral $I_{0C}(u_j;c)$

$$\begin{aligned}
I_{0C}(u_j;c) = \int_{-\infty}^{+\infty} e^{-\frac{(x_j-u_j)^2}{2c}}\,p_C'(x_j)\,dx_j &= \int_{-\infty}^{+\infty} e^{-\frac{(x_j-u_j)^2}{2c}}\frac{(-x_j)}{\sigma^3\sqrt{2\pi}}e^{-\frac{x_j^2}{2\sigma^2}}\,dx_j\\
&= e^{-\frac{(x_j-u_j)^2}{2c}}\frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{x_j^2}{2\sigma^2}}\Bigg|_{-\infty}^{+\infty} - \int_{-\infty}^{+\infty} -\frac{(x_j-u_j)}{c}e^{-\frac{(x_j-u_j)^2}{2c}}\frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{x_j^2}{2\sigma^2}}\,dx_j\\
&= -\frac{u_j}{c\sigma\sqrt{2\pi}}\int_{-\infty}^{+\infty} e^{-\frac{(x_j-u_j)^2}{2c}}e^{-\frac{x_j^2}{2\sigma^2}}\,dx_j - \frac{\sigma^2}{c}\int_{-\infty}^{+\infty} e^{-\frac{(x_j-u_j)^2}{2c}}\frac{-x_j}{\sigma^3\sqrt{2\pi}}e^{-\frac{x_j^2}{2\sigma^2}}\,dx_j\\
&= -\frac{u_j}{c\sigma\sqrt{2\pi}}\int_{-\infty}^{+\infty} e^{-\frac{(x_j-u_j)^2}{2c}}e^{-\frac{x_j^2}{2\sigma^2}}\,dx_j - \frac{\sigma^2}{c}I_{0C}(u_j;c).\\
&= -\frac{c}{c+\sigma^2}\frac{u_j}{c\sigma\sqrt{2\pi}}\frac{\sqrt{\pi}}{2}\sqrt{\frac{2c\sigma^2}{\sigma^2+c}}\mathrm{erf}\left(\frac{cx_j+\sigma^2(x_j-u_j)}{\sqrt{2c\sigma^2(\sigma^2+c)}}\right)e^{-\frac{1}{2}\frac{u_j^2}{\sigma^2+c}}\Bigg|_{-\infty}^{+\infty}\\
&= -\frac{\sqrt{c}\,u_j}{(\sigma^2+c)^{\frac{3}{2}}}\,e^{-\frac{1}{2}\frac{u_j^2}{\sigma^2+c}}.
\end{aligned} \tag{A.4}$$

Computation of the integral $I_{1C}(u_j; c)$

$$\begin{aligned}
I_{1C}(u_j;c) &= \int_{-\infty}^{+\infty} x_j\, e^{-\frac{(x_j-u_j)^2}{2c}}\, p'_C(x_j)\, dx_j = \int_{-\infty}^{+\infty} x_j\, \frac{(-x_j)}{\sigma^3\sqrt{2\pi}}\, e^{-\frac{(x_j-u_j)^2}{2c}} e^{-\frac{x_j^2}{2\sigma^2}}\, dx_j \\
&= x_j e^{-\frac{(x_j-u_j)^2}{2c}} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x_j^2}{2\sigma^2}}\Big|_{-\infty}^{+\infty} - \int_{-\infty}^{+\infty} \left(1 - x_j\frac{(x_j-u_j)}{c}\right) e^{-\frac{(x_j-u_j)^2}{2c}} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x_j^2}{2\sigma^2}}\, dx_j \\
&= -\frac{1}{\sigma\sqrt{2\pi}}\int_{-\infty}^{+\infty} e^{-\frac{(x_j-u_j)^2}{2c}} e^{-\frac{x_j^2}{2\sigma^2}}\, dx_j - \frac{u_j}{\sigma\sqrt{2\pi}\, c}\int_{-\infty}^{+\infty} x_j e^{-\frac{(x_j-u_j)^2}{2c}} e^{-\frac{x_j^2}{2\sigma^2}}\, dx_j \\
&\quad + \frac{1}{\sigma\sqrt{2\pi}\, c}\int_{-\infty}^{+\infty} x_j^2 e^{-\frac{(x_j-u_j)^2}{2c}} e^{-\frac{x_j^2}{2\sigma^2}}\, dx_j \\
&= -\frac{1}{\sigma\sqrt{2\pi}} \frac{\sqrt{\pi}}{2}\sqrt{\frac{2c\sigma^2}{\sigma^2+c}}\, \mathrm{erf}\left(\frac{cx_j+\sigma^2(x_j-u_j}{\sqrt{2c\sigma^2(\sigma^2+c)}}\right) e^{-\frac{1}{2}\frac{u_j^2}{\sigma^2+c}}\Big|_{-\infty}^{+\infty} \\
&\quad + + \frac{u_j\sigma^2}{c} I_{0C}(u_j;c) - \frac{\sigma^2}{c} I_{1C}(u_j;c) \\
&= -\frac{c}{(\sigma^2+c)}\sqrt{\frac{c}{\sigma^2+c}} e^{-\frac{1}{2}\frac{u_j^2}{\sigma^2+c}} - \frac{c}{(\sigma^2+c)} \frac{u_j\sigma^2}{c} \frac{\sqrt{c}\,u_j}{(\sigma^2+c)^{\frac{3}{2}}} e^{-\frac{1}{2}\frac{u_j^2}{\sigma^2+c}} \\
&= -\frac{\sqrt{c}\,\left(\sigma^2 c + c^2 + \sigma^2 u_j^2\right)}{(\sigma^2+c)^{\frac{5}{2}}} e^{-\frac{1}{2}\frac{u_j^2}{\sigma^2+c}}.
\end{aligned} \tag{A.5}$$

Computation of the integral $I_{2C}(u_j; c)$

$$\begin{aligned}
I_{2C}(u_j;c) &= \int_{-\infty}^{+\infty} \mathrm{erf}\left(\frac{x_j-u_j}{\sqrt{2c}}\right) p'_C(x_j)\, dx_j \\
&= \mathrm{erf}\left(\frac{x_j-u_j}{\sqrt{2c}}\right) p_C(x_j)\Big|_{-\infty}^{+\infty} + \int_{-\infty}^{+\infty} \frac{d}{dx_j}(\mathrm{erf}\left(\frac{x_j-u_j}{\sqrt{2c}}\right)) p_C(x_j)\, dx_j \\
&= \frac{1}{\sigma\sqrt{2\pi}}\mathrm{erf}\left(\frac{x_j-u_j}{\sqrt{2c}}\right) e^{-\frac{x_j^2}{2\sigma^2}}\Big|_{-\infty}^{+\infty} - \frac{1}{\sigma\sqrt{2\pi}}\sqrt{\frac{2}{\pi c}}\int_{-\infty}^{+\infty} e^{-\frac{(x_j-u_j)^2}{2c}} e^{-\frac{x_j^2}{2\sigma^2}}\, dx_j \\
&= -\frac{1}{\sigma\sqrt{2\pi}}\sqrt{\frac{2}{\pi c}}\, e^{-\frac{u_j^2}{2(\sigma^2+c)}} \int_{-\infty}^{+\infty} e^{-\frac{\left((\sigma^2+c)x_j-\sigma^2 u_j\right)^2}{\sqrt{2\sigma^2 c(\sigma^2+c)}}}\, dx_j \\
&= -\frac{1}{\sigma\sqrt{2\pi}}\sqrt{\frac{2}{\pi c}}\frac{\sqrt{\pi}}{2}\sqrt{\frac{2\sigma^2 c}{\sigma^2+c}}\, e^{-\frac{u_j^2}{2(\sigma^2+c)}}\, \mathrm{erf}\left(\frac{cx_j+\sigma^2(x_j-u_j}{\sqrt{2c\sigma^2(\sigma^2+c)}}\right)\Big|_{-\infty}^{+\infty} \\
&= -\sqrt{\frac{2}{\pi(\sigma^2+c)}}\, e^{-\frac{u_j^2}{2(\sigma^2+c)}}.
\end{aligned} \tag{A.6}$$

It should be noted that integrals in (A.4), (A.5) and (A.6) hold when $\frac{1}{c} + \frac{1}{\sigma^2} > 0$, which is always true since both c and σ^2 represent variances. Finally we can write

$$\begin{aligned}
I_0(u_j;c) &= -\frac{\gamma}{c} \sum_{m=1}^{N} \epsilon_m (u_j - b_m)\, e^{-\frac{(u_j-b_m)^2}{2c}} \quad - (1-\gamma) \frac{\sqrt{c}\, u_j}{(\sigma^2+c)^{\frac{3}{2}}} \times e^{-\frac{1}{2}\frac{u_j^2}{\sigma^2+c}}, \\
I_1(u_j;c) &= -\gamma \sum_{m=1}^{N} \epsilon_m\, e^{-\frac{(u_j-b_m)^2}{2c}} \left(1 + \frac{b_m}{c} u_j - \frac{b_m^2}{c}\right) \quad - (1-\gamma) \frac{\sqrt{c}\, \left(\sigma^2 c + c^2 + \sigma^2 u_j^2\right)}{(\sigma^2+c)^{\frac{5}{2}}} e^{-\frac{1}{2}\frac{u_j^2}{\sigma^2+c}}, \\
I_2(u_j;c) &= -\gamma \sqrt{\frac{2}{\pi c}} \sum_{m=1}^{N} \epsilon_m\, e^{-\frac{(u_j-b_m)^2}{2c}} \quad - (1-\gamma) \sqrt{\frac{2}{\pi(\sigma^2+c)}} e^{-\frac{u_j^2}{2(\sigma^2+c)}}
\end{aligned} \tag{A.7}$$

A.2 Estimation Update for a Weighted Average of n-Gaussian Distributions

Here we assume that the prior (the discrete part together with the continuous part) can be approximated by a weighted average of n-Gaussian distributions as

$$p(x_j) = \sum_{k=1}^{n} \gamma_k p_k(x_j) = \sum_{k=1}^{n} \frac{\gamma_k}{\sigma_k \sqrt{2\pi}}\; e^{-\frac{\left(x_j-\mu_k\right)^2}{2\sigma_k^2}},$$

where $\sum_{k=1}^{n} \gamma_k = 1$. Form (3.31), it follows that

$$\begin{aligned}
I_0(u_j;c) &= \int_{-\infty}^{+\infty} e^{-\frac{\left(x_j-u_j\right)^2}{2c}}\; p'(x_j)\, dx_j \\
&= \sum_{k=1}^{n} \gamma_k \int_{-\infty}^{+\infty} e^{-\frac{\left(x_j-u_j\right)^2}{2c}}\; p'_k(x_j)\, dx_j = \sum_{k=1}^{n} \gamma_k I_{0k}(u_j;c), \\
I_1(u_j;c) &= \int_{-\infty}^{+\infty} x_j\, e^{-\frac{\left(x_j-u_j\right)^2}{2c}}\; p'(x_j)\, dx_j \\
&= \sum_{k=1}^{n} \gamma_k \int_{-\infty}^{+\infty} x_j\, e^{-\frac{\left(x_j-u_j\right)^2}{2c}}\; p'_k(x_j)\, dx_j = \sum_{k=1}^{n} \gamma_k I_{1k}(u_j;c), \\
I_2(u_j;c) &= \int_{-\infty}^{+\infty} \operatorname{erf}\left(\frac{x_j-u_j}{\sqrt{2c}}\right) p'(x_j)\, dx_j \\
&= \sum_{k=1}^{n} \gamma_k \int_{-\infty}^{+\infty} \operatorname{erf}\left(\frac{x_j-u_j}{\sqrt{2c}}\right) p'_k(x_j)\, dx_j = \sum_{k=1}^{n} \gamma_k I_{2k}(u_j;c).
\end{aligned} \tag{A.8}$$

Now we can obtain expressions for $I_0(u_j;c)$, $I_1(u_j;c)$ and $I_2(u_j;c)$ by calculating contributions of a single Gaussian distribution, namely $I_{0k}(u_j;c)$, $I_{1k}(u_j;c)$ and $I_{2k}(u_j;c)$.

Computation of the integral $I_{0k}(u_j;c)$

$$\begin{aligned} I_{0k}(u_j;c) &= \int_{-\infty}^{+\infty} e^{-\frac{\left(x_j-u_j\right)^2}{2c}}\, p_k'(x_j)\,dx_j = \int_{-\infty}^{+\infty} e^{-\frac{\left(x_j-u_j\right)^2}{2c}} \frac{-(x_j-\mu_k)}{\sigma_k^3\sqrt{2\pi}} e^{-\frac{\left(x_j-\mu_k\right)^2}{2\sigma_k^2}}\,dx_j \\ &= \int_{-\infty}^{+\infty} e^{-\frac{\left(\left(x_j-\mu_k\right)-\left(u_j-\mu_k\right)\right)^2}{2c}} \frac{-(x_j-\mu_k)}{\sigma_k^3\sqrt{2\pi}} e^{-\frac{\left(x_j-\mu_k\right)^2}{2\sigma_k^2}}\,dx_j. \end{aligned} \tag{A.9}$$

Substituting $x_j-\mu_k = x_j'$ and consequently $dx_j = dx_j'$ it follows that

$$I_{0k}(u_j;c) = \int_{-\infty}^{+\infty} e^{-\frac{\left(x_j'-\left(u_j-\mu_k\right)\right)^2}{2c}} \frac{(-x_j')}{\sigma_k^3\sqrt{2\pi}} e^{-\frac{x_j'^2}{2\sigma_k^2}}\,dx_j'. \tag{A.10}$$

Using result from (A.4) we can write

$$I_{0k}(u_j;c) = -\frac{\sqrt{c}\,(u_j-\mu_k)}{(\sigma_k^2+c)^{\frac{3}{2}}}\, e^{-\frac{1}{2}\frac{\left(u_j-\mu_k\right)^2}{\sigma_k^2+c}}. \tag{A.11}$$

Computation of the integral $I_{1k}(u_j;c)$

$$\begin{aligned} I_{1k}(u_j;c) &= \int_{-\infty}^{+\infty} x_j\, e^{-\frac{(x_j-u_j)^2}{2c}}\, p_k'(x_j)\,dx_j = \int_{-\infty}^{+\infty} x_j\, e^{-\frac{(x_j-u_j)^2}{2c}} \frac{-(x_j-\mu_k)}{\sigma_k^3\sqrt{2\pi}} e^{-\frac{\left(x_j-\mu_k\right)^2}{2\sigma_k^2}}\,dx_j \\ &= \int_{-\infty}^{+\infty} (x_j-\mu_k)\, e^{-\frac{\left(\left(x_j-\mu_k\right)-\left(u_j-\mu_k\right)\right)^2}{2c}} \frac{-(x_j-\mu_k)}{\sigma_k^3\sqrt{2\pi}} e^{-\frac{\left(x_j-\mu_k\right)^2}{2\sigma_k^2}}\,dx_j + \mu_k I_{0k}(u_j;c). \end{aligned} \tag{A.12}$$

Using the same substitution as in case of calculating $I_{0k}(u_j;c)$ it follows

$$I_{1k}(u_j;c) = \int_{-\infty}^{+\infty} x_j' \frac{-x_j'}{\sigma_k^3\sqrt{2\pi}} e^{-\frac{\left(x_j'-\left(u_j-\mu_k\right)\right)^2}{2c}} e^{-\frac{x_j'^2}{2\sigma_k^2}}\,dx_j' + \mu_k I_{0k}(u_j;c). \tag{A.13}$$

Using result from (A.5) we can write

$$
\begin{aligned}
I_{1k}(u_j;c) &= -\frac{\sqrt{c}\,\left(\sigma_k^2 c + c^2 + \sigma_k^2\,(u_j-\mu_k)^2\right)}{(\sigma_k^2+c)^{\frac{5}{2}}} e^{-\frac{1}{2}\frac{\left(u_j-\mu_k\right)^2}{\sigma_k^2+c}} + \mu_k I_{0k}(u_j;c) \\
&= -\frac{\sqrt{c}\,\left(\sigma_k^2 c + c^2 + \sigma_k^2\,(u_j-\mu_k)^2\right)}{(\sigma_k^2+c)^{\frac{5}{2}}} e^{-\frac{1}{2}\frac{\left(u_j-\mu_k\right)^2}{\sigma_k^2+c}} - \mu_k \frac{\sqrt{c}\,(u_j-\mu_k)}{(\sigma_k^2+c)^{\frac{3}{2}}}\, e^{-\frac{1}{2}\frac{\left(u_j-\mu_k\right)^2}{\sigma_k^2+c}} \\
&= -\frac{\sqrt{c}\,\left(\sigma_k^2 c + c^2 + \sigma_k^2\,(u_j-\mu_k)^2 + \mu_k\,(u_j-\mu_k)\,(\sigma_k^2+c)\right)}{(\sigma_k^2+c)^{\frac{5}{2}}} e^{-\frac{1}{2}\frac{\left(u_j-\mu_k\right)^2}{\sigma_k^2+c}}\,.
\end{aligned}
\tag{A.14}
$$

Computation of the integral $I_{2k}(u_j;c)$

$$
I_{2k}(u_j;c) = \int_{-\infty}^{+\infty} \operatorname{erf}\left(\frac{x_j-u_j}{\sqrt{2c}}\right) p_k'(x_j)\,dx_j = \int_{-\infty}^{+\infty} \operatorname{erf}\left(\frac{x_j-u_j}{\sqrt{2c}}\right) \frac{-\left(x_j-\mu_k\right)}{\sigma_k^3\sqrt{2\pi}} e^{-\frac{\left(x_j-\mu_k\right)^2}{2\sigma_k^2}}\,dx_j.
\tag{A.15}
$$

Again, making substitution $x_j - \mu_k = x_j'$ and consequently $dx_j = dx_j'$ it follows that

$$
I_{2k}(u_j;c) = \int_{-\infty}^{+\infty} \operatorname{erf}\left(\frac{x_j' - (u_j-\mu_k)}{\sqrt{2c}}\right) \frac{-x_j'}{\sigma_k^3\sqrt{2\pi}} e^{-\frac{x_j'^2}{2\sigma_k^2}}\,dx_j'\,,
\tag{A.16}
$$

and together with the results from (A.6) we can write

$$
I_{2k}(u_j;c) = -\sqrt{\frac{2}{\pi(\sigma_k^2+c)}}\, e^{-\frac{1}{2}\frac{\left(u_j-\mu_k\right)^2}{\sigma_k^2+c}}\,.
\tag{A.17}
$$

Finally we have

$$
\begin{aligned}
I_0(u_j;c,\alpha) &= -\sqrt{2\pi c}\sum_{k=1}^{n} \gamma_k\,\mathcal{N}(u_j;\mu_k,\sigma_{k,c}^2)\frac{u_j-\mu_k}{\sigma_{k,c}^2}, \\
I_1(u_j;c,\alpha) &= -\sqrt{2\pi c}\sum_{k=1}^{n} \gamma_k\,\mathcal{N}(u_j;\mu_k,\sigma_{k,c}^2)\left[\frac{\sigma_k^2(u_j-\mu_k)^2}{\sigma_{k,c}^4} + \frac{c+\mu_k(u_j-\mu_k)}{\sigma_{k,c}^2}\right], \\
I_2(u_j;c,\alpha) &= -2\sum_{k=1}^{n} \gamma_k\,\mathcal{N}(u_j;\mu_k,\sigma_{k,c}^2),
\end{aligned}
\tag{A.18}
$$

where $\sigma_{k,c}^2 = \sigma_k^2 + c$.

Appendix B

Measurement Updates of the GAMP Algorithm

B.1 Measurement Update for the SDM Channel

Since $\mathsf{z} \sim \mathcal{N}(\mu_z, \sigma_z^2)$ we can write

$$f_{\mathsf{z}}(z \,|\, \mathsf{y} = q_k) = \frac{f_{\mathsf{z}}(z, \mathsf{y} = q_k)}{p_{\mathsf{y}}(q_k)} = \frac{p_{\mathsf{y}}(q_k \,|\, z)}{p_{\mathsf{y}}(q_k)} f_{\mathsf{z}}(z) = \frac{p_{\mathsf{y}}(q_k \,|\, z)}{p_{\mathsf{y}}(q_k)} \mathcal{N}(z;\, \mu_z, \sigma_z^2). \tag{B.1}$$

First we need to find the expression for $p_{\mathsf{y}}(q_k \,|\, z)$ which summarizes the measurement proces. In this case, each measurement y_i has the value of $Q(z)$ with the probability $(1-\epsilon)$, or any value from the set of symbols with the probanility ϵ. This means that eventhough the symbol of $Q(z)$ is flipped, it might be flipped to the same symbol. Flipping of a symbol is modeled with random variable b which takes a value from the set $\{0, 1\}$ (1 means that a symbol was flipped).

$$\begin{aligned} p_{\mathsf{y}}(q_k \,|\, z) &= \sum_{b\in\{0,1\}} p_{\mathsf{y}}(q_k, b \,|\, z) = \sum_{b\in\{0,1\}} p_{\mathsf{y}}(q_k \,|\, b, z)\, p(b \,|\, z) = \sum_{b\in\{0,1\}} p_{\mathsf{y}}(q_k \,|\, b, z)\, p(b) \\ &= p_{\mathsf{y}}(q_k \,|\, b = 1, z)\, p(b = 1) + p_{\mathsf{y}}(q_k \,|\, b = 0, z)\, p(b = 0) \\ &= \epsilon 2^{-R} + p_{\mathsf{y}}(q_k \,|\, b = 0, z)(1 - \epsilon) \end{aligned} \tag{B.2}$$

Now we can write $p_{\mathsf{y}}(q_k)$ as

$$\begin{aligned} p_{\mathsf{y}}(q_k) &= \int_{-\infty}^{+\infty} f_{\mathsf{y},\mathsf{z}}(q_k, z)\, dz = \int_{-\infty}^{+\infty} p_{\mathsf{y}}(q_k \,|\, z) f_{\mathsf{z}}(z)\, dz \\ &= \int_{-\infty}^{+\infty} \left[\epsilon 2^{-R} + p_{\mathsf{y}}(q_k \,|\, b=0, z)(1-\epsilon)\right] f_{\mathsf{z}}(z)\, dz \\ &= \epsilon 2^{-R} + (1-\epsilon) \int_{-\infty}^{+\infty} p_{\mathsf{y}}(q_k \,|\, b=0, z) f_{\mathsf{z}}(z)\, dz \\ &= \epsilon 2^{-R} + (1-\epsilon) p_{\hat{\mathsf{y}}}(q_k). \end{aligned} \tag{B.3}$$

Here $p_{\hat{\mathsf{y}}}(q_k)$ denotes the probability that y takes the value q_k where no symbols are being flipped (noiseless R-bit *generalized approximate message passing* (GAMP)). The mean and the power of $\mathsf{z} \,|\, \mathsf{y} = q_k$ can be calculated as

$$\begin{aligned} \mathbb{E}\{\mathsf{z} \,|\, \mathsf{y} = q_k\} &= \int_{-\infty}^{\infty} z f_{\mathsf{z}}(z \,|\, \mathsf{y} = q_k)\, dz = \int_{-\infty}^{\infty} z \frac{p_{\mathsf{y}}(q_k \,|\, z)}{p_{\mathsf{y}}(q_k)} f_{\mathsf{z}}(z)\, dz \\ &= \frac{1}{p_{\mathsf{y}}(q_k)} \int_{-\infty}^{\infty} z \left[\epsilon 2^{-R} + (1-\epsilon) p_{\mathsf{y}}(q_k \,|\, b=0, z)\right] f_{\mathsf{z}}(z)\, dz \\ &= \frac{1}{p_{\mathsf{y}}(q_k)} \left[\epsilon 2^{-R} \mu + (1-\epsilon) \frac{p_{\hat{\mathsf{y}}}(q_k)}{p_{\hat{\mathsf{y}}}(q_k)} \int_{-\infty}^{\infty} z p_{\mathsf{y}}(q_k \,|\, b=0, z)\, f_{\mathsf{z}}(z)\, dz\right] \\ &= \frac{1}{p_{\mathsf{y}}(q_k)} \left[\epsilon 2^{-R} \mu + (1-\epsilon) p_{\hat{\mathsf{y}}}(q_k)\, \mathbb{E}\{\hat{\mathsf{z}} \,|\, \mathsf{y} = q_k\}\right], \end{aligned} \tag{B.4}$$

$$\begin{aligned} \mathbb{E}\{\mathsf{z}^2 \,|\, y = q_k\} &= \int_{-\infty}^{\infty} z^2 f_{\mathsf{z}}(z \,|\, \mathsf{y} = q_k)\, dz = \int_{-\infty}^{\infty} z^2 \frac{p_{\mathsf{y}}(q_k \,|\, z)}{p_{\mathsf{y}}(q_k)} f_{\mathsf{z}}(z)\, dz \\ &= \frac{1}{p_{\mathsf{y}}(q_k)} \int_{-\infty}^{\infty} z^2 \left[\epsilon 2^{-R} + (1-\epsilon) p_{\mathsf{y}}(q_k \,|\, b=0, z)\right] f_{\mathsf{z}}(z)\, dz \\ &= \frac{1}{p_{\mathsf{y}}(q_k)} \left[\epsilon 2^{-R} (\sigma^2 + \mu^2) + (1-\epsilon) \frac{p_{\hat{\mathsf{y}}}(q_k)}{p_{\hat{\mathsf{y}}}(q_k)} \int_{-\infty}^{\infty} z^2 p_{\mathsf{y}}(q_k \,|\, b=0, z)\, f_{\mathsf{z}}(z)\, dz\right] \\ &= \frac{1}{p_{\mathsf{y}}(q_k)} \left[\epsilon 2^{-R} (\sigma^2 + \mu^2) + (1-\epsilon) p_{\hat{\mathsf{y}}}(q_k)\, \mathbb{E}\{\hat{\mathsf{z}}^2 \,|\, \mathsf{y} = q_k\}\right]. \end{aligned} \tag{B.5}$$

B.2 Measurement Update for the AWGN Channel

Since $z \sim \mathcal{N}(\mu_z, \sigma_z^2)$ we can write

$$f_{\mathsf{z}}(z \,|\, y) = \frac{f_{\mathsf{z},\mathsf{y}}(z, y)}{f_{\mathsf{y}}(y)} = \frac{f_{\mathsf{y}}(y \,|\, z)}{f_{\mathsf{y}}(y)} f_{\mathsf{z}}(z) = \frac{f_{\mathsf{y}}(y \,|\, z)}{f_{\mathsf{y}}(y)} \mathcal{N}(z;\, \mu_z, \sigma_z^2). \tag{B.6}$$

The conditional *probability density function* (pdf) $\mathsf{y}\,|\,\mathsf{z}$ can be written as

$$f_{\mathsf{y}}(y\,|\,z) = f_{\mathsf{y}}(y\,|\,Q(z)) = f_{\mathsf{y}}\Big(Q(z) + w = y\,|\,Q(z)\Big) = f_{\mathsf{w}}\Big(y - Q(z)\Big). \tag{B.7}$$

In (B.6), $f_{\mathsf{y}}(y)$ is a normalizing term that can be calculated as

$$\begin{aligned} f_{\mathsf{y}}(y) &= \int_{-\infty}^{\infty} f_{\mathsf{y,z}}(y,z)\,dz = \int_{-\infty}^{\infty} f_{\mathsf{y}}(y\,|\,z) f_{\mathsf{z}}(z)\,dz = \int_{-\infty}^{\infty} f_{\mathsf{w}}\Big(y - Q(z)\Big) f_{\mathsf{z}}(z)\,dz \\ &= \sum_{k=0}^{2^R-1} \int_{b_k}^{b_{k+1}} f_{\mathsf{w}}\Big(y - Q(z)\Big) f_{\mathsf{z}}(z)\,dz = \sum_{k=0}^{2^R-1} \int_{b_k}^{b_{k+1}} f_{\mathsf{w}}(y - q_k) f_{\mathsf{z}}(z)\,dz \\ &= \sum_{k=0}^{2^R-1} f_{\mathsf{w}}(y - q_k) \int_{b_k}^{b_{k+1}} f_{\mathsf{z}}(z)\,dz = \sum_{k=0}^{2^R-1} f_{\mathsf{w}}(y - q_k) p_k, \end{aligned} \tag{B.8}$$

where

$$p_k = \frac{1}{2}\left[\operatorname{erf}\left(\frac{b_{k+1} - \mu_z}{\sqrt{2\sigma_z^2}}\right) - \operatorname{erf}\left(\frac{b_k - \mu_z}{\sqrt{2\sigma_z^2}}\right)\right]. \tag{B.9}$$

The mean and the power of $\mathsf{z}\,|\,\mathsf{y}$ can be calculated as

$$\begin{aligned} \mathbb{E}\{\mathsf{z}\,|\,y\} &= \int_{-\infty}^{\infty} z f_{\mathsf{z}}(z\,|\,y)\,dz = \int_{-\infty}^{\infty} z \frac{f_{\mathsf{y}}(y\,|\,z)}{f_{\mathsf{y}}(y)} f_{\mathsf{z}}(z)\,dz = \frac{1}{f_{\mathsf{y}}(y)} \int_{-\infty}^{\infty} z f_{\mathsf{y}}(y\,|\,z) f_{\mathsf{z}}(z)\,dz \\ &= \frac{1}{f_{\mathsf{y}}(y)} \int_{-\infty}^{\infty} z f_{\mathsf{w}}\Big(y - Q(z)\Big) f_{\mathsf{z}}(z)\,dz = \frac{1}{f_{\mathsf{y}}(y)} \sum_{k=0}^{2^R-1} \int_{b_k}^{b_{k+1}} z f_{\mathsf{w}}\Big(y - Q(z)\Big) f_{\mathsf{z}}(z)\,dz \\ &= \frac{1}{f_{\mathsf{y}}(y)} \sum_{k=0}^{2^R-1} \int_{b_k}^{b_{k+1}} z f_{\mathsf{w}}\Big(y - q_k\Big) f_{\mathsf{z}}(z)\,dz = \frac{1}{f_{\mathsf{y}}(y)} \sum_{k=0}^{2^R-1} f_{\mathsf{w}}\Big(y - q_k\Big) \int_{b_k}^{b_{k+1}} z f_{\mathsf{z}}(z)\,dz \\ &= \frac{1}{f_{\mathsf{y}}(y)} \sum_{k=0}^{2^R-1} f_{\mathsf{w}}\Big(y - q_k\Big) m_k, \end{aligned} \tag{B.10}$$

where

$$\begin{aligned} m_k &= \mu p_k - \sqrt{\frac{\sigma_z^2}{2\pi}} \left(e^{-\frac{(b_{k+1}-\mu_z)^2}{2\sigma_z^2}} - e^{-\frac{(b_k-\mu_z)^2}{2\sigma_z^2}}\right) \\ &= \mu p_k - \sigma_z^2 \Big(\mathcal{N}(b_{k+1}; \mu_z, \sigma_z^2) - \mathcal{N}(b_k; \mu_z, \sigma_z^2)\Big). \end{aligned} \tag{B.11}$$

$$\begin{aligned}\mathbb{E}\{\mathsf{z}^2 \,|\, y\} &= \int_{-\infty}^{\infty} z^2 f_{\mathsf{z}}(z \,|\, y)\, dz = \int_{-\infty}^{\infty} z^2 \frac{f_{\mathsf{y}}(y \,|\, z)}{f_{\mathsf{y}}(y)} f_{\mathsf{z}}(z)\, dz = \frac{1}{f_{\mathsf{y}}(y)} \int_{-\infty}^{\infty} z^2 f_{\mathsf{y}}(y \,|\, z) f_{\mathsf{z}}(z)\, dz \\ &= \frac{1}{f_{\mathsf{y}}(y)} \int_{-\infty}^{\infty} z^2 f_{\mathsf{w}}\Big(y - Q(z)\Big) f_{\mathsf{z}}(z)\, dz = \frac{1}{f_{\mathsf{y}}(y)} \sum_{k=0}^{2^R-1} \int_{b_k}^{b_{k+1}} z^2 f_{\mathsf{w}}\Big(y - Q(z)\Big) f_{\mathsf{z}}(z)\, dz \\ &= \frac{1}{f_{\mathsf{y}}(y)} \sum_{k=0}^{2^R-1} \int_{b_k}^{b_{k+1}} z^2 f_{\mathsf{w}}\Big(y - q_k\Big) f_{\mathsf{z}}(z)\, dz = \frac{1}{f_{\mathsf{y}}(y)} \sum_{k=0}^{2^R-1} f_{\mathsf{w}}\Big(y - q_k\Big) \int_{b_k}^{b_{k+1}} z^2 f_{\mathsf{z}}(z)\, dz \\ &= \frac{1}{f_{\mathsf{y}}(y)} \sum_{k=0}^{2^R-1} f_{\mathsf{w}}\Big(y - q_k\Big) s_k,\end{aligned} \tag{B.12}$$

where

$$s_k = (\sigma_z^2 + \mu^2) p_k - \sigma_z^2 \Big((b_{k+1} + \mu_z) \mathcal{N}(b_{k+1}; \mu_z, \sigma_z^2) - (b_k + \mu_z) \mathcal{N}(b_k; \mu_z, \sigma_z^2) \Big). \tag{B.13}$$

B.3 Measurement Update for the SR ADC

Using Bayes' rule we can write the distribution $f_{\mathsf{z}}(z \,|\, y)$ as

$$f_{\mathsf{z}}(z \,|\, y) = \frac{1}{f_{\mathsf{y}}(y)} f_{\mathsf{y}}(y \,|\, z)\, f_{\mathsf{z}}(z). \tag{B.14}$$

In the case of the noiseless channel and *self-reset analog to digital converter* (SR-ADC), the conditional distribution that models the measurement process $f_{\mathsf{y}}(y \,|\, z)$ amounts to

$$f_{\mathsf{y}}(y \,|\, z) = \delta(y - \mathcal{M}_\lambda(z)). \tag{B.15}$$

Therefore, we can rewrite (B.14) as

$$f_{\mathsf{z}}(z \,|\, y) = \frac{1}{f_{\mathsf{y}}(y)} \delta(y - \mathcal{M}_\lambda(z))\, \mathcal{N}(z\,; \mu_z, \sigma_z^2), \tag{B.16}$$

where $f_{\mathsf{y}}(y)$ is the normalizing term that ensures that the conditional distribution integrates to 1. Therefore, it can be calculated as

$$\begin{aligned} f_{\mathsf{y}}(y) &= \int_{-\infty}^{+\infty} f_{\mathsf{y},\mathsf{z}}(y,z)\,dz = \int_{-\infty}^{+\infty} f_{\mathsf{y}}(y\,|\,z)\,f_{\mathsf{z}}(z)\,dz = \int_{-\infty}^{+\infty} f_{\mathsf{y}}(y\,|\,z)\,\mathcal{N}(z\,;\mu_z,\sigma_z^2)\,dz \\ &= \int_{-\infty}^{+\infty} \delta(y-\mathcal{M}_\lambda(z))\,\mathcal{N}(z\,;\mu_z,\sigma_z^2)\,dz = \sum_{z:\mathcal{M}_\lambda(z)=y} \mathcal{N}(z\,;\mu_z,\sigma_z^2) \\ &= \sum_{k=-\infty}^{\infty} \underbrace{\mathcal{N}(y+2k\lambda\,;\mu_z,\sigma_z^2)}_{\gamma_k} = \sum_{k=-\infty}^{\infty} \gamma_k \end{aligned} \tag{B.17}$$

It follows that the $\mathbb{E}\{\mathsf{z}\,|\,y\}$ and $\mathrm{var}\{\mathsf{z}\,|\,y\}$ $(=\mathbb{E}\{\mathsf{z}^2\,|\,y\}-(\mathbb{E}\{\mathsf{z}\,|\,y\})^2)$ can be calculated as

$$\begin{aligned} \mathbb{E}\{\mathsf{z}\,|\,y\} &= \int_{-\infty}^{\infty} z\,f_{\mathsf{z}}(z\,|\,y)\,dz = \frac{1}{f_{\mathsf{y}}(y)}\int_{-\infty}^{\infty} z\,\delta(y-\mathcal{M}_\lambda(z))\,\mathcal{N}(z\,;\mu_z,\sigma_z^2)\,dz \\ &= \frac{1}{f_{\mathsf{y}}(y)}\sum_{z:\mathcal{M}_\lambda(z)=y} z\,\mathcal{N}(z\,;\mu_z,\sigma_z^2) = \frac{1}{f_{\mathsf{y}}(y)}\sum_{k=-\infty}^{\infty}(y+2k\lambda)\,\gamma_k\,, \end{aligned} \tag{B.18}$$

and

$$\begin{aligned} \mathbb{E}\{\mathsf{z}^2\,|\,y\} &= \int_{-\infty}^{\infty} z^2\,f_{\mathsf{z}}(z\,|\,y)dz = \frac{1}{f_{\mathsf{y}}(y)}\int_{-\infty}^{\infty} z^2\delta(y-\mathcal{M}_\lambda(z))\,\mathcal{N}(z\,;\mu_z,\sigma_z^2)dz \\ &= \frac{1}{f_{\mathsf{y}}(y)}\sum_{z:\mathcal{M}_\lambda(z)=y} z^2\,\mathcal{N}(z\,;\mu_z,\sigma_z^2) = \frac{1}{f_{\mathsf{y}}(y)}\sum_{k=-\infty}^{\infty}(y+2k\lambda)^2\,\gamma_k. \end{aligned} \tag{B.19}$$

List of Acronyms

AbS	Analysis-by-Synthesis
AMP	approximate message passing
AOP	adaptive outlier pursuit
AWGN	additive white Gaussian noise
BAMP	Bayesian approximate message passing
BG	Bernoulli-Gauss
BIHT	binary iterative hard thresholding
BP	basis pursuit
BPDN	basis pursuit denoising
BSC	binary symmetric channel
cdf	cumulative distribution function
CLS	constrained least squares
CoSaMP	compressive sampling matching pursuit
CS	compressed sensing
DCT-II	2 dimensional discrete cosine transformation
DSP	digital signal processing
GAMP	generalized approximate message passing
IHT	iterative hard thresholding
i.i.d.	independent and identically distributed
IST	iterative soft thresholding
LASSO	least absolute shrinkage and selection operator
LS	least squares

MAP	maximum a posteriori
MC	Monte-Carlo
ML	maximum likelihood
MMSE	minimum mean squared error
MPDQ	message-passing de-quantization
MRI	magnetic resonance imaging
MSE	mean squared error
MSP	matching sign pursuit
NMSE	normalized mean squared error
OMP	orthogonal matching pursuit
pdf	probability density function
pmf	probability mass function
QCS	quantized compressed sensing
RIP	restricted isometry property
RSS	restricted-step shrinkage
SDM	symmetric discrete memoryless
SE	state evolution
SNR	signal-to-noise ratio
SP	subspace pursuit
SR-ADC	self-reset analog to digital converter
US	unlimited sampling
whp	with high probability

References

[1] A Bromiley, P. (2003). Products and convolutions of gaussian distributions.

[2] Anderson, C. R., Venkatesh, S., Ibrahim, J. E., Buehrer, R. M., and Reed, J. H. (2009). Analysis and implementation of a time-interleaved adc array for a software-defined uwb receiver. *IEEE Transactions on Vehicular Technology*, 58(8):4046–4063.

[3] Baraniuk, R. G. (2011). More is less: Signal processing and the data deluge. *Science*, 331(6018):717–719.

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

[5] Berger, C. R., Wang, Z., Huang, J., and Zhou, S. (2010). Application of compressive sensing to sparse channel estimation. *IEEE Communications Magazine*, 48(11):164–174.

[6] Berry, A. C. (1941). The accuracy of the gaussian approximation to the sum of independent variates. *Transactions of the American Mathematical Society*, 49(1):122–136.

[7] Bhandari, A., Krahmer, F., and Raskar, R. (2017). On unlimited sampling. In *2017 International Conference on Sampling Theory and Applications (SampTA)*, pages 31–35.

[8] Bhandari, A., Krahmer, F., and Raskar, R. (2018). Unlimited sampling of sparse signals. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4569–4573.

[9] Bhandari, A., Wallace, A. M., and Raskar, R. (2016). Super-resolved time-of-flight sensing via fri sampling theory. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4009–4013.

[10] Billingsley, P. (1999). *Convergence of probability measures*. Wiley Series in Probability and Statistics: Probability and Statistics. John Wiley & Sons Inc., New York, second edition. A Wiley-Interscience Publication.

[11] Blu, T., Dragotti, P., Vetterli, M., Marziliano, P., and Coulot, L. (2008). Sparse sampling of signal innovations. *IEEE Signal Processing Magazine*, 25(2):31–40.

[12] Blumensath, T. and Davies, M. E. (2008). Iterative thresholding for sparse approximations. *Journal of Fourier Analysis and Applications*, 14(5-6):629–654.

[13] Boufounos, P. T. (2009). Greedy sparse signal reconstruction from sign measurements. In *2009 Conference Record of the Forty-Third Asilomar Conference on Signals, Systems and Computers*, pages 1305–1309.

[14] Boufounos, P. T. and Baraniuk, R. G. (2008). 1-bit compressive sensing. In *Information Sciences and Systems, 2008. CISS 2008. 42nd Annual Conference on*, pages 16–21.

[15] Bourgain, J., Dilworth, S. J., Ford, K., Konyagin, S., and Kutzarova, D. (2010). Explicit constructions of RIP matrices and related problems. *arXiv e-prints*, page arXiv:1008.4535.

[16] Brynjolfsson, E. and McAfee, A. (2014). *The Second Machine Age: Work, Progress, and Prosperity in a Time of Brilliant Technologies*. W. W. Norton.

[17] C. Eldar, Y. and Kutyniok, G., editors (2012). *Compressed Sensing: Theory and Applications*. Cambridge University Press.

[18] Candes, E., Romberg, J., and Tao, T. (2006). Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics*, 59(8):1207–1223.

[19] Candes, E. J. (2008). The restricted isometry property and its implications for compressed sensing. *Comptes Rendus Mathematique*, 346(9):589 – 592.

[20] Candes, E. J., Romberg, J., and Tao, T. (2006). Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509.

[21] Candès, E. J. and Tao, T. (2005). Decoding by linear programming. *IEEE Transactions on Information Theory*, 51(12):4203–4215.

[22] Cover, T. M. and Thomas, J. A. (2006). *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, New York, NY, USA.

[23] Dai, W. and Milenkovic, O. (2011). Information theoretical and algorithmic approaches to quantized compressive sensing. *IEEE Transactions on Communications*, 59(7):1857–1866.

[24] Dai, W., Pham, H. V., and Milenkovic, O. (2009a). A comparative study of quantized compressive sensing schemes. In *2009 IEEE International Symposium on Information Theory*, pages 11–15.

[25] Dai, W., Pham, H. V., and Milenkovic, O. (2009b). Distortion-rate functions for quantized compressive sensing. In *2009 IEEE Information Theory Workshop on Networking and Information Theory*, pages 171–175.

[26] Daubechies, I., Defrise, M., and De Mol, C. (2004). An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics*, 57(11):1413–1457.

[27] Davenport, M. A., Duarte, M. F., Eldar, Y. C., and Kutyniok, G. (2011). Introduction to compressed sensing. *Preprint*.

[28] Davenport, M. A. and Wakin, M. B. (2010). Analysis of orthogonal matching pursuit using the restricted isometry property. *IEEE Transactions on Information Theory*, 56(9):4395–4401.

[29] DeVore, R. A. (2007). Deterministic constructions of compressed sensing matrices. *Journal of Complexity*, 23(4):918 – 925. Festschrift for the 60th Birthday of Henryk Woźniakowski.

[30] Donoho, D. L. (2006). Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306.

[31] Donoho, D. L., Elad, M., and Temlyakov, V. N. (2006). Stable recovery of sparse overcomplete representations in the presence of noise. *IEEE Transactions on Information Theory*, 52(1):6–18.

[32] Donoho, D. L., Maleki, A., and Montanari, A. (2009). Message-passing algorithms for compressed sensing. *Proceedings of the National Academy of Sciences*, 106(45):18914–18919.

[33] Donoho, D. L., Maleki, A., and Montanari, A. (2010a). Message passing algorithms for compressed sensing: I. motivation and construction. In *IEEE Workshop on Information Theory*, pages 1–5. IEEE.

[34] Donoho, D. L., Maleki, A., and Montanari, A. (2010b). Message passing algorithms for compressed sensing: II. analysis and validation. In *2010 IEEE Workshop on Information Theory*, pages 1–5.

[35] Donoho, D. L., Maleki, A., and Montanari, A. (2011). How to design message passing algorithms for compressed sensing. *preprint*.

[36] Donoho, D. L., Maleki, A., and Montanari, A. (2011). The noise-sensitivity phase transition in compressed sensing. *IEEE Transactions on Information Theory*, 57(10):6920–6941.

[37] Dragotti, P. L., Vetterli, M., and Blu, T. (2007). Sampling moments and reconstructing signals of finite rate of innovation: Shannon meets strang amp;ndash;fix. *IEEE Transactions on Signal Processing*, 55(5):1741–1757.

[38] Duarte, M. F., Davenport, M. A., Takbar, D., Laska, J. N., Sun, T., Kelly, K. F., and Baraniuk, R. G. (2008). Single-pixel imaging via compressive sampling. *IEEE Signal Processing Magazine*, 25(2):83–91.

[39] Foucart, S. and Rauhut, H. (2013). *A Mathematical Introduction to Compressive Sensing*, volume 1. Birkhäuser Basel.

[40] Goertz, N. and Hannak, G. (2017). Fast bayesian signal recovery in compressed sensing with partially unknown discrete prior. In *WSA 2017; 21th International ITG Workshop on Smart Antennas*, pages 1–8.

[41] Gunturk, C., Lammers, M., Powell, A., Saab, R., and Yilmaz, O. (2010). Sigma delta quantization for compressed sensing. In *Proc. Annual Conference on Information Sciences and Systems (CISS)*, pages 1–6.

[42] Guo, C. and Davies, M. (2014). Bayesian optimal compressed sensing without priors: Parametric SURE approximate message passing. In *Proc. 22nd European Signal Processing Conference (EUSIPCO)*, pages 1347–1351, Lisbon, Portugal.

[43] Hannak, G. (2017). *Bayesian Approximate Message Passing for Distributed Compressed Sensing*. PhD thesis, E389.

[44] Indyk, P. (2008). Explicit constructions for compressed sensing of sparse signals. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '08, pages 30–33, Philadelphia, PA, USA. Society for Industrial and Applied Mathematics.

[45] Jacques, L., Hammond, D., and Fadili, J. (2011). Dequantizing compressed sensing: When oversampling and non-Gaussian constraints combine. *IEEE Trans. on Inf. Theory*, 57(1):559–571.

[46] Jacques, L., Laska, J. N., Boufounos, P. T., and Baraniuk, R. G. (2013). Robust 1-bit compressive sensing via binary stable embeddings of sparse vectors. *IEEE Transactions on Information Theory*, 59(4):2082–2102.

[47] Jol, H. M. (2009). *Ground Penetrating Radar Theory and Applications*. Elsevier, Amsterdam. An optional note.

[48] Kamilov, U., Goyal, V., and Rangan, S. (2012a). Message-passing de-quantization with applications to compressed sensing. *IEEE Trans. on Signal Processing*, 60(12):6270–6281.

[49] Kamilov, U. S., Bourquard, A., Amini, A., and Unser, M. (2012b). One-bit measurements with adaptive thresholds. *IEEE Signal Processing Letters*, 19(10):607–610.

[50] Laska, J. N., Boufounos, P. T., Davenport, M. A., and Baraniuk, R. G. (2011a). Democracy in action: Quantization, saturation, and compressive sensing. *Applied and Computational Harmonic Analysis*, 31(3):429–443.

[51] Laska, J. N., Wen, Z., Yin, W., and Baraniuk, R. G. (2011b). Trust, but verify: Fast and accurate signal recovery from 1-bit compressive measurements. *IEEE Transactions on Signal Processing*, 59(11):5289–5301.

[52] Li, L. and P. Speed, T. (2001). Parametric deconvolution of positive spike trains. *Ann. Stat.*, 28.

[53] Liu, E. and Temlyakov, V. N. (2012). The orthogonal super greedy algorithm and applications in compressed sensing. *IEEE Transactions on Information Theory*, 58(4):2040–2047.

[54] Lustig, M., Donoho, D., and Pauly, J. M. (2007). Sparse MRI: The application of compressed sensing for rapid MR imaging. *Magnetic Resonance in Medicine*, 58(6):1182–1195.

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

[56] Maleh, R. (2011). Improved RIP analysis of orthogonal matching pursuit. *CoRR*, abs/1102.4311.

[57] Maleki, A. (2010). *PhD Thesis: Approximate Message Passing Algorithms for Compressed Sensing*. Stanford University.

[58] Maleki, A. and Donoho, D. L. (2010). Optimally tuned iterative reconstruction algorithms for compressed sensing. *IEEE Journal of Selected Topics in Signal Processing*, 4(2):330–341.

[59] Max, J. (1960). Quantizing for minimum distortion. *IRE Transactions on Information Theory*, IT-6(1):7–12.

[60] McEwen, A. and Cassimally, H. (2013). *Designing the Internet of Things*. Wiley Publishing, 1st edition.

[61] Mo, Q. and Shen, Y. (2012). A remark on the restricted isometry property in orthogonal matching pursuit. *IEEE Transactions on Information Theory*, 58(6):3654–3656.

[62] Montanari, A. (2012). Graphical models concepts in compressed sensing. *Compressed Sensing: Theory and Applications*, pages 394–438.

[63] Mousavi, A., Maleki, A., and Baraniuk, R. G. (2013). Parameterless optimal approximate message passing. *CoRR*, abs/1311.0035.

[64] Musa, O. and Goertz, N. (2015). Quantization of compressed sensing measurements using analysis-by-synthesis with bayesian-optimal approximate message passing. In *2015 IEEE 16th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pages 510–514.

[65] Musa, O., Hannak, G., and Goertz, N. (2016). Generalized approximate message passing for one-bit compressed sensing with AWGN. In *2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 1428–1432.

[66] Musa, O., Hannak, G., and Goertz, N. (2017). Efficient recovery from noisy quantized compressed sensing using generalized approximate message passing. In *2017 IEEE 7th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pages 1–5.

[67] Musa, O., Jung, P., and Goertz, N. (2018). Generalized approximate message passing for unlimited sampling of sparse signals. In *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 336–340.

[68] Muthukrishnan, S. (2005). Data streams: Algorithms and applications. *Foundations and Trends® in Theoretical Computer Science*, 1(2):117–236.

[69] Needell, D. and Tropp, J. (2009). Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(3):301 – 321.

[70] Paredes, J. L., Arce, G. R., and Wang, Z. (2007). Ultra-wideband compressed sensing: Channel estimation. *IEEE Journal of Selected Topics in Signal Processing*, 1(3):383–395.

[71] Rangan, S. (2011). Generalized approximate message passing for estimation with random linear mixing. In *2011 IEEE International Symposium on Information Theory Proceedings*, pages 2168–2172.

[72] Rangan, S., Fletcher, A. K., Goyal, V. K., and Schniter, P. (2012). Hybrid generalized approximate message passing with applications to structured sparsity. In *IEEE International Symposium on Information Theory*, pages 1236–1240. IEEE.

[73] Schaller, R. R. (1997). Moore's law: past, present and future. *IEEE Spectrum*, 34(6):52–59.

[74] Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423.

[75] Shannon, C. E. (1949). Communication in the presence of noise. *Proc. Institute of Radio Engineers*, 37(1):10–21.

[76] Shirazinia, A., Chatterjee, S., and Skoglund, M. (2013). Analysis-by-synthesis quantization for compressed sensing measurements. *IEEE Trans. on Signal Processing*, 61(22):5789–5800.

[77] Stein, C. (1986). Approximate computation of expectations. *Lecture Notes-Monograph Series*, 7:i–164.

[78] Sun, J. Z. and Goyal, V. K. (2009). Optimal quantization of random measurements in compressed sensing. In *2009 IEEE International Symposium on Information Theory*, pages 6–10.

[79] T., P. J. (2014). *PhD Thesis: Approximate Message Passing Algorithms for Generalized Bilinear Inference*. The Ohio State University.

[80] Taubock, G., Hlawatsch, F., Eiwen, D., and Rauhut, H. (2010). Compressive estimation of doubly selective channels in multicarrier systems: Leakage effects and sparsity-enhancing processing. *IEEE Journal of Selected Topics in Signal Processing*, 4(2):255–271.

[81] Vetterli, M., Marziliano, P., and Blu, T. (2002). Sampling signals with finite rate of innovation. *IEEE Transactions on Signal Processing*, 50(6):1417–1428.

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

[83] Yan, M., Yang, Y., and Osher, S. (2012). Robust 1-bit compressive sensing using adaptive outlier pursuit. *IEEE Transactions on Signal Processing*, 60(7):3868–3875.

[84] Zhang, Y., Dong, Z., Phillips, P., Wang, S., Ji, G., and Yang, J. (2015). Exponential wavelet iterative shrinkage thresholding algorithm for compressed sensing magnetic resonance imaging. *Information Sciences*, 322:115–132.

[85] Zhang, Y., Peterson, B. S., Ji, G., and Dong, Z. (2014). Energy preserved sampling for compressed sensing MRI. *Computational and Mathematical Methods in Medicine*, 2014.

[86] Ziniel, J., Schniter, P., and Sederberg, P. (2015). Binary linear classification and feature selection via generalized approximate message passing. *IEEE Transactions on Signal Processing*, 63(8):2020–2032.

[87] Zymnis, A., Boyd, S., and Candes, E. (2010). Compressed sensing with quantized measurements. *IEEE Signal Processing Letters*, 17(2):149–152.