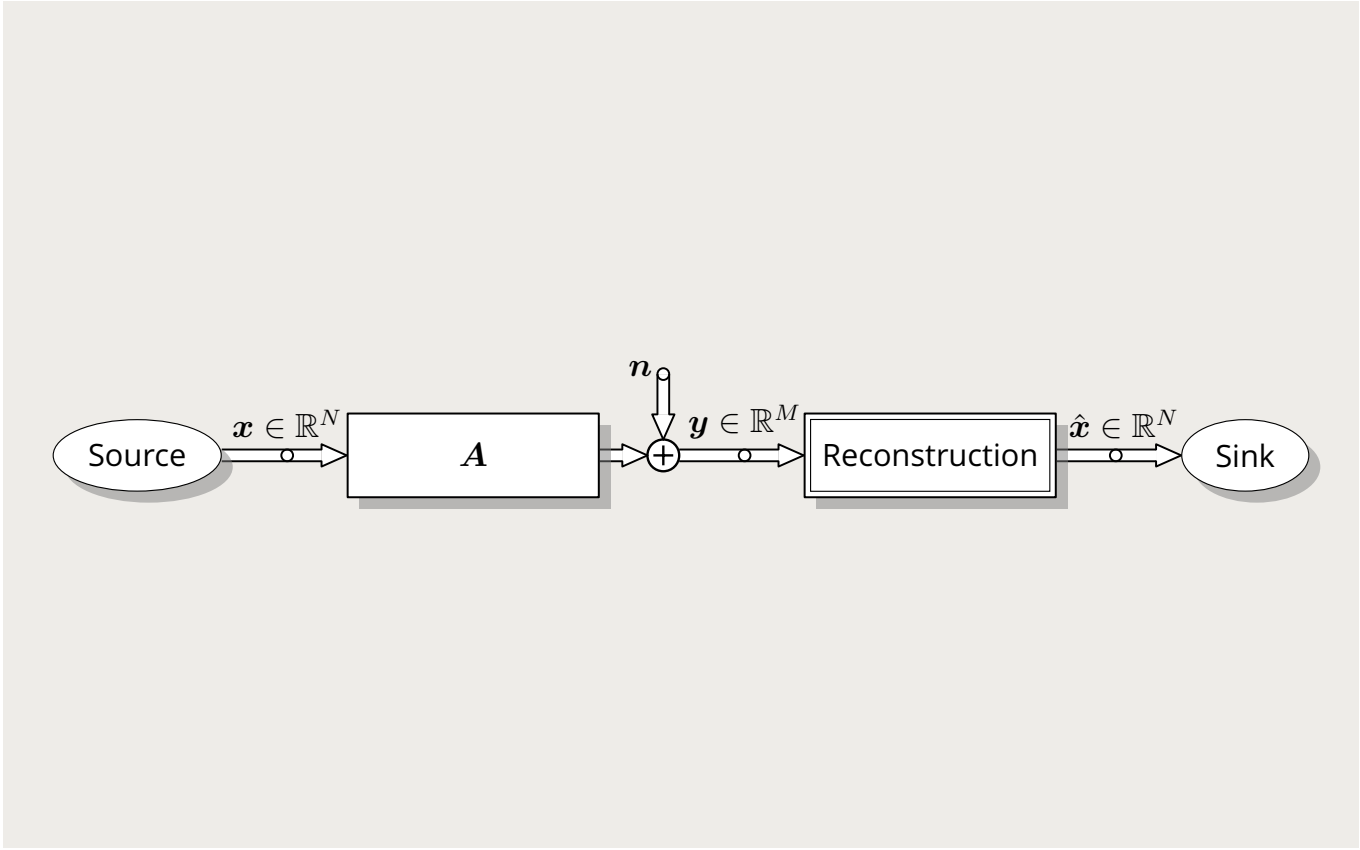
ulm university universität
uulm

Carmen Sippel

Turbo and Message Passing Approaches for Iterative Signal Recovery in Compressed Sensing



Band 5
Schriftenreihe des Instituts für Nachrichtentechnik

Carmen Sippel

Turbo and Message Passing Approaches for Iterative Signal Recovery in Compressed Sensing

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.dnb.de abrufbar.

Dissertation, Universität Ulm,
Fakultät für Ingenieurwissenschaften, Informatik und Psychologie, 2023

Impressum

Universität Ulm
Institut für Nachrichtentechnik
Prof. Dr.-Ing. Robert Fischer
Albert-Einstein-Allee 43
89081 Ulm

Eine Übersicht über alle Bände der Schriftenreihe finden Sie unter
https://nt.uni-ulm.de/schriften

e-ISBN 978-3-948303-38-9

universität ulm

Turbo and Message Passing Approaches for Iterative Signal Recovery in Compressed Sensing

DISSERTATION

zur Erlangung des akademischen Grades eines

DOKTOR-INGENIEURS
(Dr.-Ing.)

der Fakultät für Ingenieurwissenschaften,
Informatik und Psychologie der Universität Ulm

von

Carmen Sippel

aus Crailsheim

Gutachter:	Prof. Dr.-Ing. Robert Fischer
	Prof. Dr.-Ing. Volker Kühn
Amtierende Dekanin:	Prof. Dr. Anke Huckauf

Ulm, 16. Juni 2023

Acknowledgment

On the way to creating this thesis more than just one person was involved. Foremost, I would like to thank my supervisor, Prof. Dr.-Ing. Robert Fischer, for giving me the chance to work on the topic of compressed sensing and for the exhaustive discussions about it, where I could learn so much that all of it did not fit into this thesis. Moreover, I would like to thank him for enabling the cooperation with the institute for digital communications in Erlangen, as well as the colleagues in Erlangen for the inclusion to the institute. I also would like to thank the Deutsche Forschungsgemeinschaft (DFG) for funding most parts of my position. My thanks also go to Prof. Dr.-Ing. Martin Bossert, former head of the institute, for the opportunity to broaden the scope of my knowledge in a collaboration with coding theorists previous to the start of my actual PhD topic. Furthermore, I would like to thank Prof. Dr.-Ing. Volker Kühn for agreeing to examine my thesis on short notice and giving me the chance to present my work to his institute. Further thanks go to Prof. Dr. Walter Karlen and Prof. Dr.-Ing. Klaus Dietmayer for being part of the colloquium.

I want to thank my current and former scientific colleagues for the fruitful discussions and the administrative staff for their aid in all organizational and technical matters, as well as the institute as a whole for enjoyable breaks, which contributed to fulfilling working days. Among them, but not restricted to my colleagues, I have to thank the people that proofread this thesis and helped enhancing it. Last but not least, I want to thank my family for encouraging me to make my own decisions and my partner for motivating me at all times and his support in every aspect of life.

Abstract

The thesis at hand deals with recovery algorithms for the reconstruction of signals from noisy and linearly compressed measurements. The procedure of linearly compressing a signal with known structure is known as compressed sensing (CS); the recovery is possible under certain conditions.

Meanwhile, there is a number of algorithms that are designed for this kind of task; each with own benefits and drawbacks. The thesis focuses on algorithms that can be derived from a suitable processing in a factor graph, which embodies the structural dependencies of the CS problem. This approach already leads to a variety of algorithms.

The most compartmentalized factorization leads to a form of the factor graph from which the so-called message passing (MP) algorithm can be derived. The opposite case, a factorization with fewest possible factors results in a graph with two factors, yielding Turbo-type algorithms that iterate between those two factors. In the thesis, both types of algorithms are derived from the respective factor graph by constrained optimization of the Kullback–Leibler divergence between the posterior and a suitably adapted substitute distribution, i.e., the respective algorithms are reconciled as instances of the same basic idea.

The thesis also comprises sequential algorithms, which are derived from factorizations of the problem that lie in between the two extreme cases explained above. The resulting algorithms can be seen as sequentialized versions of the Turbo-type algorithms. There are two possibilities. One way to sequentialize the algorithm is by iterating through the measurements; the other is to iterate through the signal components (additionally to iterating between the two main parts). The sequentialization allows for various schedules, which are examined in numerical simulations.

Furthermore, an estimation-theoretic view is cast on the processing between the factors. The result reveals possibilities for improvement of the algorithms. Several variants of the algorithms are proposed and compared in numerical simulations.

Contents

1. Introduction and Outline

Compressed sensing (a.k.a. compressive sampling) [CRT06, Don06] denotes the strategy of simplifying the compression procedure at the point where data is acquired, at the cost of higher effort for reconstruction of the signal at a receiving point. Hence, the strategy is in need for suitable algorithms that recover the signal as exact as possible with least possible expense.

Starting with ℓ_1-minimization[1] [CRT06], which is a convex relaxation of the problem but still has relatively high complexity, the journey of possible reconstruction algorithms went over so-called *greedy* algorithms such as *orthogonal matching pursuit* (OMP) [PRK93] and *compressive sampling matching pursuit* (CoSaMP) [NT09], which *successively* build a support set for the recovered signal, to *iterative* algorithms. Both, greedy and iterative algorithms reduce the computational complexity compared to ℓ_1-minimization. For the iterative algorithms, there is to mention iterative hard thresholding (IHT) [BD09] and iterative soft thresholding (IST) [DDD04], which both iterate between the solution of the linear system of equations obtained from the (noisy) compressed measurements—we denote this as the *channel-constrained* part—and a solution that follows a given signal structure—denoted as *signal-constrained* part—the difference being the way of adjusting to the signal structure. While being much simpler than ℓ_1-minimization, the recovery performance of these initial iterative algorithms suffered.

A huge step in terms of recovery performance was made with the invention of *approximate message passing* (AMP) [DMM09, Mal11] without increasing the complexity significantly compared to the other iterative algorithms. One step more complex and more robust to various compression scenarios is the so-called *vector approximate message passing* (VAMP) [RSF19] algorithm. The eponymous message-passing approach is an inference technique that is widely used in communications engineering, for example in the decoding of error-correcting codes such as low-density parity-check (LDPC) codes [Gal62] and Turbo codes [BGT93, HOP96]. Message passing (MP) usually works on graphical models, such as *factor graphs* [KFL01, Loe04], that represent the structure of the problem in detail, leading to messages being passed on the many edges in the graph, thereby creating the name. On the contrary, both, the AMP and the

[1] Usually performed by the simplex algorithm [DT97]. For the definition of the ℓ_1-norm, see Appendix A.

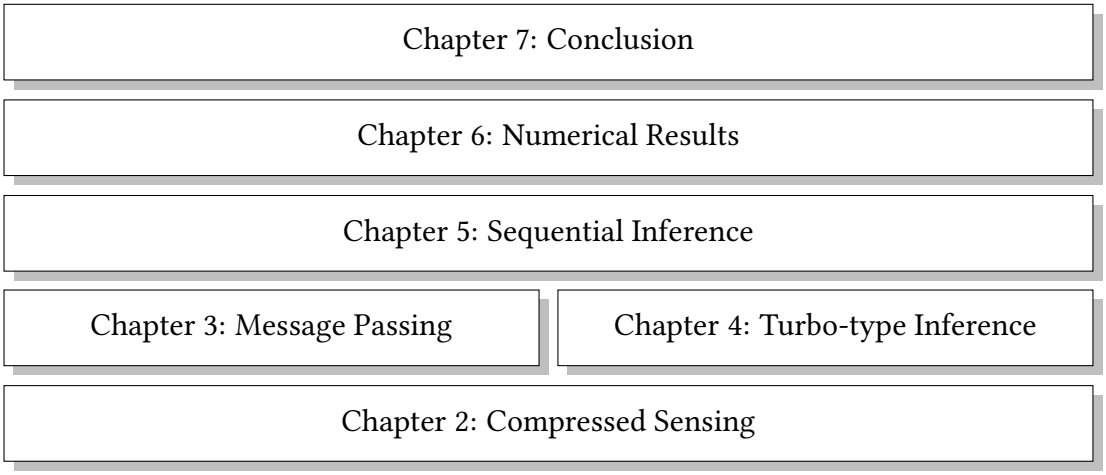


Figure 1.1: Visualization of the structure of the dissertation.

VAMP algorithm, combine so much of the detailed processing that in fact a Turbo-type processing results; similarly to IHT and IST, the iteration is between two parts but the recovery performance is much better.

In this thesis, the message-passing and Turbo-type approach are contrasted and both are derived from the framework of constrained optimization for the compressed sensing problem. Moreover, sequential approaches that stand in between both strategies are discussed. Furthermore, an estimation-theoretic point of view is cast on the exchange of the variables between the different solutions; leading to improved variants of the algorithms. The outline of the thesis is illustrated in Fig. 1.1 and described more detailed in the following.

Chapter 2 contains the theoretical basis to understand the reconstruction problem behind compressed sensing upon which the rest of the thesis is built. It defines the basic terms, especially for the compressed sensing model and states the estimations that are necessary to obtain solutions for the subproblems (signal- and channel-constrained part). Furthermore, we motivate why it is sensible to use the Kullback–Leibler divergence as cost function for the constrained optimization and state the different forms of the factor graph, that will be used in the subsequent chapters to derive the algorithms.

The next two chapters are depicted next to each other in Fig. 1.1, because they contain two opposing approaches to solve the recovery problem.

Chapter 3 focuses on the message-passing-type inference for CS with a detailed view on the connections in the factor graph. A message passing (MP) algorithm as in [KMS+12] is derived as result from minimizing the Kullback–Leibler divergence for a substitute distribution that is tailored to the factor graph for CS under expectation constraints. Similar derivations can be found in [YFW05] and [HOW+05] for general inference problems; here, we focus on the CS problem. The result stresses the importance of the computation of *extrinsic* [BG96] for the exchange of parameters between solutions of the subproblems.

In Chapter 4, the same derivation is used to obtain Turbo-type algorithms; the difference being how the factor graph is considered. In contrast to Chapter 3, the factor graph is combined to only contain two major parts between which the iterations take place. The results

lead among others to the famous VAMP [RSF19] algorithm; with the *extrinsic* being the crucial point making VAMP able to stand out from the other iterative algorithms IHT, IST, and, e.g., iterative soft-feedback (ISF) [SF16], which is processed as IHT and IST, but uses the same signal-constrained estimation as VAMP. Additionally, several adaptions of the derived algorithms are discussed that result from an estimation-theoretic view on the extrinsic calculations.

Chapter 5 discusses two sequential recovery approaches that lie in between the two extreme cases of Turbo-type and message-passing-type algorithms. The derivation utilizes the same constrained optimization strategy as before for yet different factor graph representations. The chapter is divided into two parts; the first part (Sec. 5.1) considers the sequential processing of the signal components, while the second part (Sec. 5.2) focuses on sequentially processing the measurements.

In Chapter 6, the results of numerical simulations comparing the derived algorithms in different scenarios are shown. We depict the compressibility capability of the derived algorithms in the standard scenario for CS in Sec. 6.3, show in Sec. 6.4 a scenario, where the estimation-theoretic adaptions from Chapter 4 can beat the standard VAMP algorithm, discuss schedules for the sequential processing of signal components in Sec. 6.5, and present a scenario, where the sequential processing of measurements can improve over VAMP in Sec. 6.6.

Chapter 7 contains the conclusions and future work.

The Appendices A–D contain basic mathematical notions that are used throughout the thesis, Appendix E collects the recovery algorithms that are presented in the thesis and in Appendix F the notation and abbreviations used in the thesis can be found.

2. Compressed Sensing

The idea for compressed sensing (CS) dates back to 2006, where two groups of scientists independently realized that it is possible to reconstruct information from a linearly compressed signal under certain conditions [CRT06, Don06]. This is astonishing in the sense that the linear compression does not make use of actual features of the signal. So far, compression techniques had considered the signal itself to find out how to suitably compress it. Considering the signal, let's call it $\boldsymbol{x}$, as vector[1] of an N-dimensional vector space (here $\mathbb{R}^N$), the new method requires only a vector-matrix-multiplication (we denote the matrix by $\boldsymbol{A} \in \mathbb{R}^{M\times N}$ with $M < N$) and is therefore completely independent of signal $\boldsymbol{x}$. The result of the compression is called $\boldsymbol{y} \in \mathbb{R}^M$ here. In engineering applications it usually inherits noise in the form of an additional random vector[2] $\mathbf{n} \in \mathbb{R}^M$. The benefit of the linear compression technique lies in the fact that the computational effort is shifted to the recovery of signal $\boldsymbol{x}$ from the measurements $\boldsymbol{y}$ together with known sensing matrix $\boldsymbol{A}$, whereas the deterministic technique splits the processing effort equally between compression and decompression. Especially the compression part is tremendously simplified by the linear acquisition of the data, since no deeper knowledge of $\boldsymbol{x}$ is leveraged. Usual decompression obtains what it believes to be the original signal $\boldsymbol{x}$ directly from $\boldsymbol{y}$, which carries the recipe to the reconstruction. In the compressed sensing approach, $\boldsymbol{y}$ does not carry this recipe and therefore requires powerful algorithms for the reconstruction of the signal. The two strategies are summarized and compared in Fig. 2.1. Double strike boxes indicate non-linear functions.

Naturally, the question arises, how reconstruction of the signal $\boldsymbol{x}$ is possible at all, if the compression process is not tailored the signal.

The answer lies in the structure of the signal.

Common sources for signals, such as photographs, speech, or binary transmission streams, although providing different realizations, usually inherit a certain statistic that applies to all

[1]The notion of vectors and vector spaces is briefly introduced in Appendix A; the general notation is explained in Appendix F.2

[2]Note that variables in serif font, e.g., x or $\boldsymbol{x}$, denote realizations of random variables or vectors in sans-serif font, e.g., x or $\mathbf{x}$.

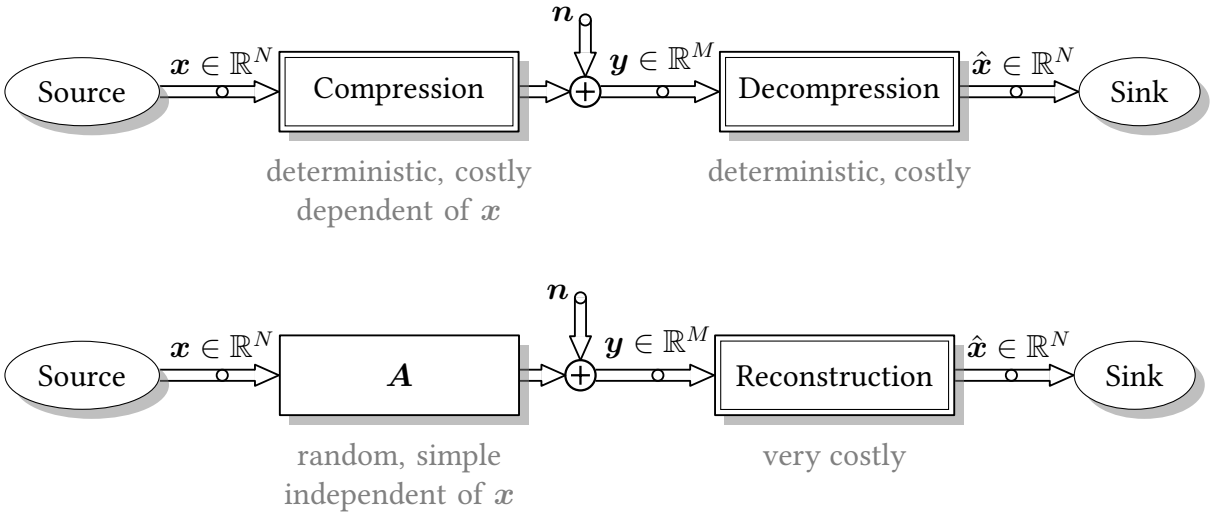


Figure 2.1: Usual transmission scheme (upper row) and compressed sensing idea (lower row).

realizations. Making use of this statistic is key to signal recovery in CS. A basic notion that specifies the structure of the signal is its *sparsity* [EK12]. A signal is said to be *sparse* if it has few non-zero entries; the sparsity being the number of non-zero entries. If the signals of a source can be represented such that they inherit a certain sparsity, this sparsity gives a natural bound on how much the signals can be compressed, i.e., the observation dimension M [EK12].

With this knowledge it is possible to describe our model for CS in detail.

2.1 System Model

Following an engineering perspective, a noisy model for the measurements in CS is considered

$$\mathbf{y} = \boldsymbol{A}\mathbf{x} + \mathbf{n} \in \mathbb{R}^M \,, \tag{2.1}$$

where the sensing matrix $\boldsymbol{A} \in \mathbb{R}^{M\times N}$ is known and $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \sigma_\mathsf{n}^2 \boldsymbol{I}_M)$ is independent and identically distributed (i.i.d.) Gaussian. Entries and rows of the sensing matrix are defined as follows

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1N} \\ a_{21} & a_{22} & \dots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{M1} & a_{M2} & \dots & a_{MN} \end{bmatrix} = \begin{bmatrix} \boldsymbol{a}_1^\top \\ \boldsymbol{a}_2^\top \\ \vdots \\ \boldsymbol{a}_M^\top \end{bmatrix} \,, \quad \boldsymbol{a}_i = [a_{i1},\, a_{i2},\, \dots,\, a_{iN}]^\top \,. \tag{2.2}$$

So far, it has been withhold that the sensing matrix also needs to fulfill some properties for recovery to be possible [EK12]. The basic point is that columns of the sensing matrix should be as incoherent[3] as possible to each other, so that the observation vector $\boldsymbol{y}$ carries as much (different) information as possible about the individual entries of $\boldsymbol{x}$. A counterexample for a coherent matrix is one with two equal columns [EK12]; the corresponding elements of $\boldsymbol{x}$ influence $\boldsymbol{y}$ in the same way and can therefore not be distinguished anymore. Another astonishing

[3]For the definition of the *coherence*, see Appendix A.

fact figured out in [Don06] is that, when the elements a_{ij} ($i \in \{1, \ldots, M\}, j \in \{1, \ldots, N\}$) are drawn i.i.d. for example from a Gaussian distribution, the resulting matrix is likely to have good properties. This fact will be used in the simulations in Chapter 6.

Let us turn to the statistic of the signal $\mathbf{x}$. For simplicity, it is assumed that the elements x_j of $\mathbf{x} = [\mathsf{x}_1, \ldots, \mathsf{x}_N]^\top$ are i.i.d. with marginal probability density function (pdf) $\mathsf{f}_\mathsf{x}(x_j)$, meaning the pdf of $\mathbf{x}$ is separable,

$$\mathsf{f}_\mathbf{x}(\boldsymbol{x}) = \prod_{j=1}^{N} \mathsf{f}_\mathsf{x}(x_j) \; . \tag{2.3}$$

A zero-mean signal $\mathbf{x}$ is assumed throughout the thesis. The fact that $\mathbf{x}$ is sparse is considered via a Dirac delta function $\delta(x)$ at zero in the marginal pdf. A possible pdf is given by [SF14]

$$\mathsf{f}_\mathsf{x}(x) = \frac{s}{2N}\delta(x+1) + \frac{N-s}{N}\delta(x) + \frac{s}{2N}\delta(x-1) \; , \tag{2.4}$$

which we call the discrete ternary (DT) prior. A prior that is very commonly used in CS is the so-called Bernoulli-Gaussian (BG) prior, whose non-zero elements are Gaussian distributed. The definition used in this thesis is given by

$$\mathsf{f}_\mathsf{x}(x) = \left(1 - \frac{s}{N}\right)\delta(x) + \frac{s/N}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) \; . \tag{2.5}$$

The task in CS is to recover the signal $\boldsymbol{x}$ for given observation $\boldsymbol{y}$, sensing matrix $\boldsymbol{A}$ and known noise variance σ_n^2.

2.2 Estimation Criteria

The problem of recovering the signal $\boldsymbol{x}$ from the compressed measurements $\boldsymbol{y}$ under a probabilistic view can be solved by finding the vector $\hat{\boldsymbol{x}}_{\mathrm{MAP}}$ that maximizes the probability that under the condition that $\boldsymbol{y}$ was measured, $\boldsymbol{x}$ is the vector which has been compressed, i.e.,

$$\hat{\boldsymbol{x}}_{\mathrm{MAP}} = \operatorname*{argmax}_{\boldsymbol{x}} \mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x}) \; , \tag{2.6}$$

which is the so-called maximum a-posteriori (MAP) criterion [Kay93].

Using Bayes' rule, the conditional pdf or posterior, corresponding to the CS problem, is given by

$$\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x}) = \frac{1}{\mathsf{f}_\mathbf{y}(\boldsymbol{y})} \mathsf{f}_{\mathbf{y}|\mathbf{x}}(\boldsymbol{x}) \cdot \mathsf{f}_\mathbf{x}(\boldsymbol{x}) \tag{2.7}$$

with channel part or likelihood function ($\|\cdot\|_2$ denotes the ℓ_2-norm, i.e., for a real vector $\boldsymbol{x}$ it is $\|\boldsymbol{x}\|_2^2 = \boldsymbol{x}^\top \boldsymbol{x}$, cf. Appendix A)

$$\mathsf{f}_{\mathbf{y}|\mathbf{x}}(\boldsymbol{x}) = \mathsf{f}_\mathbf{n}(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}) = \frac{1}{\sqrt{(2\pi\sigma_\mathsf{n}^2)^M}} \exp\left(-\frac{1}{2\sigma_\mathsf{n}^2} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2\right) \; . \tag{2.8}$$

We call $\mathsf{f}_\mathbf{x}(\boldsymbol{x})$ the signal part or a-priori distribution or simply *prior*.

2.2.1 Minimum Mean-Squared Error Estimation

Another approach is the minimum mean-squared error (MMSE) estimation criterion, which leads to the conditional mean[4] [Kay93][5]

$$\hat{\boldsymbol{x}}_{\mathrm{MMSE}} \stackrel{\mathrm{def}}{=} \mathrm{E}_{\mathsf{x}}\{\mathsf{x} \mid \boldsymbol{y}\} = \mathrm{E}_{\mathsf{x}\sim\mathsf{f}_{\mathsf{x}|\mathsf{y}}}\{\mathsf{x}\} = \int \boldsymbol{x}\, \mathsf{f}_{\mathsf{x}|\mathsf{y}}(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x}\ . \tag{2.9}$$

MAP and MMSE estimate coincide if $\boldsymbol{x}$ and $\boldsymbol{y}$ are jointly Gaussian (which is not the case for the given problem). Obtaining these solutions directly is infeasible due to the high dimensionality of the signal $\boldsymbol{x}$. Hence, there is a need for tractable procedures to recover the signal $\boldsymbol{x}$. First, two separate MMSE solutions are considered, which solve the two main parts of the problem by approximating the other, respectively.

2.2.1.1 LMMSE Estimator

Considering the *channel part* $\mathsf{f}_{\mathsf{y}|\mathsf{x}}(\boldsymbol{x})$ of the conditional pdf (2.7) and assuming that the signal vector is Gaussian distributed with mean $\boldsymbol{m}_{\mathsf{x}}$ and covariance matrix $\boldsymbol{\Phi}_{\mathsf{x}}$, i.e., $\mathsf{x} \sim \mathcal{N}(\boldsymbol{m}_{\mathsf{x}},\, \boldsymbol{\Phi}_{\mathsf{x}})$, (which is not the case in CS) linear estimation is optimal in the MMSE sense [Kay93]. The corresponding conditional mean estimator reads [Kay93]

$$\begin{aligned}
\boldsymbol{m}_{\mathrm{c}} &= \mathrm{E}_{\mathsf{x}}\{\mathsf{x} \mid \boldsymbol{y}\} = \frac{\int \boldsymbol{x}\mathsf{f}_{\mathsf{y}|\mathsf{x}}(\boldsymbol{x})\mathsf{f}_{\mathsf{x}}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}}{\int \mathsf{f}_{\mathsf{y}|\mathsf{x}}(\boldsymbol{x})\mathsf{f}_{\mathsf{x}}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}} \\
&= \boldsymbol{m}_{\mathsf{x}} + \boldsymbol{\Phi}_{\mathsf{x}}\boldsymbol{A}^{\mathsf{T}}(\boldsymbol{A}\boldsymbol{\Phi}_{\mathsf{x}}\boldsymbol{A}^{\mathsf{T}} + \sigma_{\mathsf{n}}^2\boldsymbol{I}_M)^{-1}(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{m}_{\mathsf{x}}) \\
&= \boldsymbol{m}_{\mathsf{x}} + (\boldsymbol{A}^{\mathsf{T}}\boldsymbol{A} + \sigma_{\mathsf{n}}^2\boldsymbol{\Phi}_{\mathsf{x}}^{-1})^{-1}\boldsymbol{A}^{\mathsf{T}}(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{m}_{\mathsf{x}})\ ,
\end{aligned} \tag{2.10}$$

the conditional covariance matrix of the estimation error is obtained as [Kay93]

$$\boldsymbol{\Phi}_{\mathrm{c}} = \mathrm{E}_{\mathsf{x}}\Big\{(\mathsf{x} - \boldsymbol{m}_{\mathrm{c}})(\mathsf{x} - \boldsymbol{m}_{\mathrm{c}})^{\mathsf{T}} \mid \boldsymbol{y}\Big\} = \sigma_{\mathsf{n}}^2\left(\boldsymbol{A}^{\mathsf{T}}\boldsymbol{A} + \sigma_{\mathsf{n}}^2\boldsymbol{\Phi}_{\mathsf{x}}^{-1}\right)^{-1}\ . \tag{2.11}$$

The diagonal entries of the covariance matrix are denoted by $\sigma_{\mathrm{c},j}^2 \stackrel{\mathrm{def}}{=} [\boldsymbol{\Phi}_{\mathrm{c}}]_{jj}$. The estimates $\boldsymbol{m}_{\mathrm{c}}$ and $\boldsymbol{\Phi}_{\mathrm{c}}$ are called the channel-constrained estimates or LMMSE (linear minimum mean-squared error) estimates.

Note that the second line in (2.10) results in a less complex implementation, since an $M \times M$ matrix has to be inverted, which is easier than inverting the $N \times N$ matrix in the third line, because in the CS scenario typically $M < N$.

2.2.1.2 LMMSE Estimator for Scalar Observation

Now, consider that only the ith row $\boldsymbol{a}_i^{\mathsf{T}} = [a_{i1},\, \ldots,\, a_{iN}]$ of the sensing matrix $\boldsymbol{A}$ is given and, thus, only a scalar observation y_i is measured, i.e., ($\mathsf{n}_i \sim \mathcal{N}(0,\, \sigma_{\mathsf{n}}^2)$ for $i \in \{1,\, \ldots,\, M\}$)

$$\mathsf{y}_i = \boldsymbol{a}_i^{\mathsf{T}}\mathsf{x} + \mathsf{n}_i\ . \tag{2.12}$$

[4] For convenience two notations are used to denote the estimation w.r.t. a certain pdf, either by indicating which pdf is used inside the integral or by conditioning the argument of the expectation.

[5] Whenever the integration boundaries are missing, the integral is restricted to the domain of the integrand.

However, as above a Gaussian signal vector $\mathbf{x}$ with mean $\boldsymbol{m}_{\mathsf{x}}$ and covariance matrix $\boldsymbol{\Phi}_{\mathsf{x}}$ is assumed. Then, the optimal conditional estimator for the entire signal vector $\boldsymbol{x}$ is again linear and looks as follows

$$\boldsymbol{m}_{\mathrm{c},i} = \mathrm{E}_{\mathbf{x}}\{\mathbf{x} \mid y_i\} = \boldsymbol{m}_{\mathsf{x}} + \boldsymbol{\Phi}_{\mathsf{x}}\boldsymbol{a}_i(\boldsymbol{a}_i^{\top}\boldsymbol{\Phi}_{\mathsf{x}}\boldsymbol{a}_i + \sigma_{\mathsf{n}}^2)^{-1}(y_i - \boldsymbol{a}_i^{\top}\boldsymbol{m}_{\mathsf{x}}) \;, \tag{2.13}$$

with (conditional) covariance matrix of the estimation error

$$\boldsymbol{\Phi}_{\mathrm{c},i} = \mathrm{E}_{\mathbf{x}}\Big\{(\mathbf{x} - \boldsymbol{m}_{\mathrm{c},i})(\mathbf{x} - \boldsymbol{m}_{\mathrm{c},i})^{\top} \mid y_i\Big\} = \sigma_{\mathsf{n}}^2\left(\boldsymbol{a}_i\boldsymbol{a}_i^{\top} + \sigma_{\mathsf{n}}^2\boldsymbol{\Phi}_{\mathsf{x}}^{-1}\right)^{-1} \;. \tag{2.14}$$

Note that the inverse in (2.13) is scalar. Via the Sherman-Morrison-Woodbury identity [GV96], the computation of the conditional variance $\boldsymbol{\Phi}_{\mathrm{c},i}$ can as well be written with a scalar inversion, instead of inverting the $N \times N$ matrix as stated above, which results in

$$\boldsymbol{\Phi}_{\mathrm{c},i} = \boldsymbol{\Phi}_{\mathsf{x}} - \frac{1}{\sigma_{\mathsf{n}}^2 + \boldsymbol{a}_i^{\top}\boldsymbol{\Phi}_{\mathsf{x}}\boldsymbol{a}_i}\boldsymbol{\Phi}_{\mathsf{x}}\boldsymbol{a}_i\boldsymbol{a}_i^{\top}\boldsymbol{\Phi}_{\mathsf{x}} \;. \tag{2.15}$$

With $\boldsymbol{m}_{\mathrm{c},i} = [m_{\mathrm{c},i,1},\, \ldots,\, m_{\mathrm{c},i,N}]^{\top}$ and $\sigma_{\mathrm{c},i,j}^2 = [\boldsymbol{\Phi}_{\mathrm{c},i}]_{jj}$ as well as $\boldsymbol{m}_{\mathsf{x}} = [m_{\mathsf{x},1},\, \ldots,\, m_{\mathsf{x},N}]^{\top}$ and $\sigma_{\mathsf{x},j}^2 = [\boldsymbol{\Phi}_{\mathsf{x}}]_{jj}$ the respective jth components can be written as

$$\begin{aligned} m_{\mathrm{c},i,j} &= m_{\mathsf{x},j} + \frac{\sigma_{\mathsf{x},j}^2 a_{ij}\left(y_i - \sum_{j'=1}^{N} a_{ij'}m_{\mathsf{x},j'}\right)}{\sigma_{\mathsf{n}}^2 + \sum_{j'=1}^{N}\sigma_{\mathsf{x},j'}^2 a_{ij'}^2} \\ &= \frac{m_{\mathsf{x},j}\sigma_{\mathsf{n}}^2 + m_{\mathsf{x},j}\sum_{\substack{j'=1\\ j'\neq j}}^{N}\sigma_{\mathsf{x},j'}^2 a_{ij'}^2 + \sigma_{\mathsf{x},j}^2 a_{ij}\left(y_i - \sum_{\substack{j'=1\\ j'\neq j}}^{N} a_{ij'}m_{\mathsf{x},j'}\right)}{\sigma_{\mathsf{n}}^2 + \sum_{j'=1}^{N}\sigma_{\mathsf{x},j'}^2 a_{ij'}^2} \;, \end{aligned} \tag{2.16}$$

$$\begin{aligned} \sigma_{\mathrm{c},i,j}^2 &= \sigma_{\mathsf{x},j}^2\left(1 - \frac{\sigma_{\mathsf{x},j}^2 a_{ij}^2}{\sigma_{\mathsf{n}}^2 + \sum_{j'=1}^{N}\sigma_{\mathsf{x},j'}^2 a_{ij'}^2}\right) \\ &= \sigma_{\mathsf{x},j}^2 \frac{\sigma_{\mathsf{n}}^2 + \sum_{\substack{j'=1\\ j'\neq j}}^{N}\sigma_{\mathsf{x},j'}^2 a_{ij'}^2}{\sigma_{\mathsf{n}}^2 + \sum_{j'=1}^{N}\sigma_{\mathsf{x},j'}^2 a_{ij'}^2} \;. \end{aligned} \tag{2.17}$$

2.2.1.3 Individual Non-Linear MMSE Estimator

The opposite view considers the *signal part* $\mathsf{f}_{\mathbf{x}}(\boldsymbol{x})$ and assumes Gaussian noise. Indeed, Gaussian noise is given in the model (2.1), however, the form of $\mathsf{f}_{\mathbf{y}|\mathbf{x}}(\boldsymbol{x})$ in (2.8) with couplings between $\boldsymbol{x}$ and $\boldsymbol{y}$ given by the sensing matrix makes the problem (2.9) intractable. Since separability is assumed, i.e., independent random variables x_j, the channel-constrained part of the posterior is approximated by ignoring the couplings and assuming independent measurements $\tilde{x}_{\mathrm{s},j}$, which represent the signal x_j in zero-mean Gaussian noise, i.e., the observation is a realization of the random variable

$$\tilde{\mathsf{x}}_{\mathrm{s},j} = \mathsf{x}_j + \mathsf{n}_{\mathrm{s},j} \;, \quad j \in \{1,\, \ldots,\, N\} \;, \tag{2.18}$$

where $\mathsf{x}_j \sim \mathsf{f}_{\mathsf{x}}(x_j)$ and $\mathsf{n}_{\mathrm{s},j} \sim \mathcal{N}(0, \tilde{\sigma}_{\mathrm{s},j}^2)$. The corresponding estimator is non-linear but can be computed per component, i.e., the conditional mean $\boldsymbol{m}_{\mathrm{s}} = [m_{\mathrm{s},1},\, \ldots,\, m_{\mathrm{s},N}]^{\top}$ can be

individually computed via

$$m_{\mathrm{s},j} = \mathrm{E}_{\mathsf{x}_j}\{\mathsf{x}_j \mid \tilde{x}_{\mathrm{s},j}\} = \frac{\int x_j \mathsf{f}_{\mathsf{x}}(x_j) \exp\left(-\frac{(\tilde{x}_{\mathrm{s},j}-x_j)^2}{2\tilde{\sigma}^2_{\mathrm{s},j}}\right) \mathrm{d}x_j}{\int \mathsf{f}_{\mathsf{x}}(x_j) \exp\left(-\frac{(\tilde{x}_{\mathrm{s},j}-x_j)^2}{2\tilde{\sigma}^2_{\mathrm{s},j}}\right) \mathrm{d}x_j} \,, \tag{2.19}$$

respectively, the conditional variance of estimation error

$$\sigma^2_{\mathrm{s},j} = \mathrm{E}_{\mathsf{x}_j}\Big\{(\mathsf{x}_j - m_{\mathrm{s},j})^2 \mid \tilde{x}_{\mathrm{s},j}\Big\} = \frac{\int (x_j - m_{\mathrm{s},j})^2 \mathsf{f}_{\mathsf{x}}(x_j) \exp\left(-\frac{(\tilde{x}_{\mathrm{s},j}-x_j)^2}{2\tilde{\sigma}^2_{\mathrm{s},j}}\right) \mathrm{d}x_j}{\int \mathsf{f}_{\mathsf{x}}(x_j) \exp\left(-\frac{(\tilde{x}_{\mathrm{s},j}-x_j)^2}{2\tilde{\sigma}^2_{\mathrm{s},j}}\right) \mathrm{d}x_j} \,, \tag{2.20}$$

for $j \in \{1, \ldots, N\}$. The exact computation depends of course on the prior $\mathsf{f}_{\mathsf{x}}(x_j)$ itself. Note that the estimation requires both variables, $\tilde{x}_{\mathrm{s},j}$ and $\tilde{\sigma}^2_{\mathrm{s},j}$, as inputs, however, we will only denote this in the conditional expectation in case that the respective dependency needs to be emphasized. In the CS literature, these computations are usually referred to as *denoising* and the respective estimator is the *denoiser*. Here, $m_{\mathrm{s},j}$ and $\sigma^2_{\mathrm{s},j}$ ($j \in \{1, \ldots, N\}$) may be called signal-constrained estimates.

2.2.2 Approximate Inference

In Sec. 2.2 (cf. (2.6) and (2.9)) it was shown that $\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})$ is the function of interest in the CS scenario. One way to approach the CS problem is to replace the "complicated" posterior $\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})$ by a simpler substitute distribution that is called $\mathsf{q}(\boldsymbol{x})$. A natural question that arises is how to choose $\mathsf{q}(\boldsymbol{x})$ suitably in order to approximate the posterior adequately. In the following, we derive why it is suitable to approach the target distribution $\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})$ by minimizing the Kullback–Leibler divergence or cross entropy [KL51] ($\log(\cdot)$ refers to the natural logarithm)

$$\mathrm{D}_{\mathrm{KL}}\Big(\mathsf{q}(\boldsymbol{x}) \,||\, \mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})\Big) \overset{\text{def}}{=} \int \mathsf{q}(\boldsymbol{x}) \log \frac{\mathsf{q}(\boldsymbol{x})}{\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})} \,\mathrm{d}\boldsymbol{x} \,. \tag{2.21}$$

To that end, it is motivated why the (differential) entropy of the substitute distribution $\mathsf{q}(\boldsymbol{x})$ should be maximized, just to find that the entropy itself should also be used to connect the substitute distribution $\mathsf{q}(\boldsymbol{x})$ with the target distribution $\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})$. The result is also motivated in [SJ80, SJ81] under the name *cross entropy minimization*. It is related to the so-called *maximum entropy method* [Jay82, WJ08, Wu12], but utilizes additionally prior knowledge. It is furthermore used in [Moh97a] for the design of iterative decoders and detectors in digital communication scenarios.

2.2.2.1 Moment Constraint

The substitute distribution $\mathsf{q}(\boldsymbol{x})$ should relate to the target $\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})$ in some way. This is obtained by requiring the equality of expectations based on the two distributions of some yet to be specified moment. Let $U(\boldsymbol{x})$ denote the statistic of interest, then it is required that

$$\mathrm{E}_{\mathbf{x}\sim\mathsf{q}}\{U(\mathbf{x})\} = \mathrm{E}_{\mathbf{x}\sim\mathsf{f}_{\mathbf{x}|\mathbf{y}}}\{U(\mathbf{x})\} \,. \tag{2.22}$$

Since there are many distributions $\mathsf{q}(\boldsymbol{x})$ that fulfill this condition, a criterion that specifies how to choose a "suitable" one is needed.

2.2.2.2 Entropy Maximization

In order to answer the question how to approximate the posterior suitably, we make use of the information theoretic quantity of uncertainty, as introduced by Shannon in 1948 [Sha48]. Since the problem under consideration deals with continuous distributions, indeed, the *differential entropy* is required. For a distribution $\mathsf{q}(\boldsymbol{x})$, it is defined as

$$\mathrm{h}(\mathsf{q}(\boldsymbol{x})) = -\int \mathsf{q}(\boldsymbol{x}) \log \mathsf{q}(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} \; . \tag{2.23}$$

The notion of entropy offers a way to measure the uncertainty of a random variable, which gives us a criterion for the adjustment of $\mathsf{q}(\boldsymbol{x})$. Indeed, maximizing the (differential) entropy (2.23) prevents us from preferring certain possibilities over others when there is no reason to do so [Jay57], i.e., a distribution with maximum entropy allows the most flexibility in choosing $\boldsymbol{x}$. This philosophy is known as *maximum entropy method* [Jay82, WJ08, Wu12].

For example, in the discrete case with finite set $\mathcal{X}$, the uniform distribution is the one that maximizes entropy [Sha48], which means that every possible $\boldsymbol{x} \in \mathcal{X}$ has the same probability of being the solution; no element is preferred over another. Under a moment constraint the distribution may change. With specified second order moment, the Gaussian distribution is the distribution that maximizes (differential) entropy (in the continuous case), which means that the Gaussian distribution is the "most random" one in this case [HF14]. In order to derive this for the scalar case in our notation, the maximization is formulated as minimization task with certain equality constraints[6]

$$\begin{aligned} \text{minimize} \quad & -\mathrm{h}(\mathsf{q}(x)) \\ \text{subject to} \quad & \mathrm{E}_{\mathsf{x}\sim\mathsf{q}}\left\{\mathsf{x}^2\right\} = \sigma_{\mathsf{x}}^2 \; , \\ & \int \mathsf{q}(x) \, \mathrm{d}x = 1 \; . \end{aligned} \tag{2.24}$$

i.e., the differential entropy is maximized while restricting the second order moment to some value σ_{x}^2 and the integral over $\mathsf{q}(x)$ to 1, in order to obtain a valid pdf. The usual approach to bring together the optimization task with the constraints is the combination in a so-called Lagrangian, which reads in this case

$$\mathcal{L}(\mathsf{q}(x),\, \mu,\, \nu) = -\mathrm{h}(\mathsf{q}(x)) + \mu \left(\mathrm{E}_{\mathsf{x}\sim\mathsf{q}}\left\{\mathsf{x}^2\right\} - \sigma_{\mathsf{x}}^2 \right) + \nu \left(\int \mathsf{q}(x) \, \mathrm{d}x - 1 \right) \; . \tag{2.25}$$

In order to find (arguments of) extreme points in this functional, the functional derivative (for a detailed explanation, see Appendix C) of the Lagrangian is necessary. With

$$\frac{\partial}{\partial \mathsf{q}} \int x^2 \mathsf{q}(x) \, \mathrm{d}x = x^2 \tag{2.26}$$

[6] For a summary of the notions from optimization theory, which are used within this thesis, the reader is referred to Appendix B.

and (C.8) it is

$$\frac{\partial \mathcal{L}}{\partial \mathsf{q}} = 1 + \log \mathsf{q}(x) + \mu x^2 + \nu \stackrel{!}{=} 0 \,, \tag{2.27}$$

which can be solved for $\mathsf{q}(x)$ to obtain

$$\mathsf{q}(x) = \exp\Big(-\mu x^2 - 1 - \nu\Big) \ . \tag{2.28}$$

The second constraint in (2.24) causes normalization via the multiplier ν, i.e., it forces $\exp(1+\nu) = \int \exp(-\mu x^2)\,\mathrm{d}x = \sqrt{\pi/\mu}$. Hence, the result is a Gaussian distribution with variance $\sigma^2 = 1/(2\mu)$. Since the mean was not specified, it ended up to be zero. Additionally specifying the mean would similarly have resulted in a Gaussian distribution with respective mean. Of course, for non-zero mean, the value of $\mathrm{E}_{\mathsf{x}\sim\mathsf{q}}\{\mathsf{x}^2\}$ would not be the variance but the second-order non-centralized moment instead.

2.2.2.3 Constrained Optimization

On the way to motivating the use of the Kullback–Leibler divergence, a constrained optimization problem for the task of approximating $\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})$ with a substitute distribution $\mathsf{q}(\boldsymbol{x})$ under a given moment constraint (and a normalization constraint) can now be stated

$$\begin{aligned} \text{minimize} \quad & -\mathrm{h}(\mathsf{q}(\boldsymbol{x})) \\ \text{subject to} \quad & \mathrm{E}_{\mathbf{x}\sim\mathsf{q}}\{U(\mathbf{x})\} = \mathrm{E}_{\mathbf{x}\sim\mathsf{f}_{\mathsf{x}|\mathsf{y}}}\{U(\mathbf{x})\} \,, \\ & \int \mathsf{q}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = 1 \,. \end{aligned} \tag{2.29}$$

Since the entropy is concave in its corresponding distribution, the optimization problem is convex and therefore uniquely solvable, as long as there are feasible points. Although this approach seems as infeasible as the ones before, it gives an interesting viewpoint on the problem. Indeed, the point of view on the problem can be changed by considering the so-called *Lagrange dual problem*, that is obtained by minimizing the *Lagrangian*, which additively combines the above minimization problem (2.29) with the constraints, over $\boldsymbol{x}$. This new point of view yields a representation that leads to the task of minimizing the Kullback–Leibler divergence, if the statistic $U(\boldsymbol{x})$ is chosen accordingly, which will get clear below. Before turning to that, the mathematical derivation of the dual problem is considered.

With Lagrangian multipliers β and ν, the respective Lagrangian reads

$$\mathcal{L}(\mathsf{q}(\boldsymbol{x}),\, \beta,\, \nu) = -\mathrm{h}(\mathsf{q}(\boldsymbol{x})) + \beta\left(\mathrm{E}_{\mathbf{x}\sim\mathsf{q}}\{U(\mathbf{x})\} - \mathrm{E}_{\mathbf{x}\sim\mathsf{f}_{\mathsf{x}|\mathsf{y}}}\{U(\mathbf{x})\}\right) + \nu\left(\int \mathsf{q}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} - 1\right) \ . \tag{2.30}$$

Differentiation with respect to $\mathsf{q}(\boldsymbol{x})$ yields a functional derivative, see Appendix C

$$\frac{\partial}{\partial \mathsf{q}(\boldsymbol{x})}\mathcal{L}(\mathsf{q}(\boldsymbol{x}),\, \beta,\, \nu) = 1 + \log \mathsf{q}(\boldsymbol{x}) + \beta U(\boldsymbol{x}) + \nu \stackrel{!}{=} 0 \tag{2.31}$$

$$\Leftrightarrow \qquad \mathsf{q}(\boldsymbol{x}) = \frac{1}{Z_U(\beta)} \exp(-\beta U(\boldsymbol{x})) \ , \tag{2.32}$$

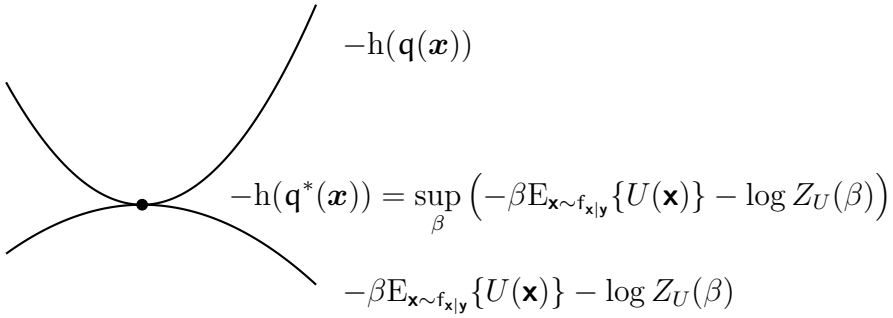


Figure 2.2: Schematic representation of the inequality (2.35).

where $Z_U(\beta) = \exp(1+\nu) = \int \exp(-\beta U(\boldsymbol{x}))\ \mathrm{d}\boldsymbol{x}$ fulfills the normalization constraint. The Lagrange dual function is then given by

$$\mathcal{L}_\mathrm{D}(\beta) = -\log Z_U(\beta) - \beta \mathrm{E}_{\mathbf{x}\sim\mathsf{f}_{\mathsf{x}|\mathsf{y}}}\{U(\mathbf{x})\}\ . \tag{2.33}$$

The dual problem is specified by the maximization of the Lagrange dual function, thus, another approach to obtain the best substitute distribution $\mathsf{q}^*(\boldsymbol{x})$ is to maximize the right-hand side of (2.33), i.e.,

$$-\mathrm{h}(\mathsf{q}^*(\boldsymbol{x})) = \sup_\beta \left(-\beta \mathrm{E}_{\mathbf{x}\sim\mathsf{f}_{\mathsf{x}|\mathsf{y}}}\{U(\mathbf{x})\} - \log Z_U(\beta)\right)\ , \tag{2.34}$$

where $\mathsf{q}^*(\boldsymbol{x})$ is the optimizer[7]. This means especially that for feasible $\mathsf{q}(\boldsymbol{x})$ and arbitrary β it is

$$-\mathrm{h}(\mathsf{q}(\boldsymbol{x})) \geq -\beta \mathrm{E}_{\mathbf{x}\sim\mathsf{f}_{\mathsf{x}|\mathsf{y}}}\{U(\mathbf{x})\} - \log Z_U(\beta)\ , \tag{2.35}$$

To verify this intuitively, consider that, on the one hand, the entropy is concave in its distribution, i.e., $-\mathrm{h}(\mathsf{q}(\boldsymbol{x}))$ is convex. On the other hand, the supremum on the right-hand side of (2.34) is obviously an upper bound for its argument. Hence, the optimizer $\mathsf{q}^*(\boldsymbol{x})$ defines the point, where both functionals touch. Moving away from that point yields the inequality. This is depicted schematically in Fig. 2.2. Noteworthy, the argument of the supremum is concave in β since the integral over the convex but positive $\exp(-\beta U(\boldsymbol{x}))$ is convex, as well as $\beta \mathrm{E}_{\mathbf{x}\sim\mathsf{f}_{\mathsf{x}|\mathsf{y}}}\{U(\mathbf{x})\}$, which is linear in β, i.e., both concave and convex, so the negative sum of both terms is concave.

Noteworthy, the optimization problem can as well be solved by considering any reformulation of the inequality (2.35) and pulling it towards equality. For feasible $\mathsf{q}(\boldsymbol{x})$, which fulfill the moment constraint, $\mathrm{E}_{\mathbf{x}\sim\mathsf{f}_{\mathsf{x}|\mathsf{y}}}\{U(\mathbf{x})\}$ can be replaced by $\mathrm{E}_{\mathbf{x}\sim\mathsf{q}}\{U(\mathbf{x})\}$. This way, the dual representation (2.33) of the optimization problem can be reformulated to the inequality

$$-\log Z_U(\beta) \leq \beta \mathrm{E}_{\mathbf{x}\sim\mathsf{q}}\{U(\mathbf{x})\} - \mathrm{h}(\mathsf{q}(\boldsymbol{x}))\ , \tag{2.36}$$

i.e., an equivalent optimization problem is obtained by minimizing the right-hand side of (2.36). Since, the inequality (2.36) holds for any β, one may deliberately specify it to a certain value, say $\beta = 1$. This specification will be kept throughout the thesis.

[7]The optimizer is indeed achieved. Since the optimization problem does not inherit inequality constraints, *Slater's condition* holds whenever the constraints are fulfilled, i.e., *strong duality* [BV04] holds.

2.2.2.4 Specifying the Moment Constraint

So far, an arbitrary statistic $U(\boldsymbol{x})$ has been considered that should tie the moments of $\mathsf{q}(\boldsymbol{x})$ and $\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})$ together. Recalling that the differential entropy can be written as

$$\mathrm{h}(\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})) = -\int \mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x}) \log \mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} = \mathrm{E}_{\mathbf{x}\sim\mathsf{f}_{\mathbf{x}|\mathbf{y}}}\Big\{-\log \mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})\Big\} \,, \tag{2.37}$$

a suitable statistic may be given by

$$U(\boldsymbol{x}) = -\log \mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x}) \,. \tag{2.38}$$

This specification means that the moments specified by $-\log \mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})$ of the substitute distribution $\mathsf{q}(\boldsymbol{x})$ should match the entropy of the posterior $\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})$. With $U(\boldsymbol{x})$ as defined in (2.38) and $\beta = 1$, the right-hand side of (2.36) can be rewritten to [YFW05][8]

$$\begin{aligned}
\mathrm{E}_{\mathbf{x}\sim\mathsf{q}}\{U(\mathbf{x})\} - \mathrm{h}(\mathsf{q}(\boldsymbol{x})) &= \int U(\boldsymbol{x})\mathsf{q}(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} + \int \mathsf{q}(\boldsymbol{x}) \log \mathsf{q}(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} \\
&= \int \mathsf{q}(\boldsymbol{x}) \log \mathsf{q}(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} - \int \mathsf{q}(\boldsymbol{x}) \log \mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} \\
&= \int \mathsf{q}(\boldsymbol{x}) \log \frac{\mathsf{q}(\boldsymbol{x})}{\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})} \,\mathrm{d}\boldsymbol{x} \\
&= \mathrm{D}_{\mathrm{KL}}\Big(\mathsf{q}(\boldsymbol{x}) \;||\; \mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})\Big) \,,
\end{aligned} \tag{2.39}$$

with $\mathrm{D}_{\mathrm{KL}}(\cdot||\cdot)$ being the *Kullback–Leibler divergence* [KL51] or *cross entropy*, which was defined in (2.21). The path behind clarifies that the Kullback–Leibler divergence measures the "closeness" between two densities on the basis of the differential entropy. This shows again the actual aim in this procedure to approximate the target distribution $\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})$, since the minimum of the Kullback–Leibler divergence ($\mathrm{D}_{\mathrm{KL}}(\mathsf{q}(\boldsymbol{x})||\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})) = 0$) is achieved exactly, if $\mathsf{q}(\boldsymbol{x}) = \mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})$. When minimizing the Kullback–Leibler divergence, one should always keep in mind that the basic idea behind the procedure is to maximize the entropy of a substitute distribution $\mathsf{q}(\boldsymbol{x})$ while staying consistent to the differential entropy of the original density $\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})$, which means the goal is to find the most freely chosen distribution $\mathsf{q}(\boldsymbol{x})$ that fits to prior knowledge of $\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})$ [SJ80].

2.2.2.5 Connection to Statistical Mechanics

In statistical mechanics, the distribution in (2.32) is known as *Boltzmann-Gibbs distribution*, which describes the distribution of particles of a system in thermal equilibrium, i.e., when there is no energy flow [Ber20]. Indeed, the function $U(\boldsymbol{x})$ is hereby the *energy* or *Hamiltonian* of the system and β is the inverse temperature. All in all, the result of the derivations coincides with the second law of thermodynamics, stating that under the constraint of conserving energy, the entropy of a system is always maximized, which results in the distribution of particles according to Boltzmann's law (2.32). It can be shown that when the system freezes

[8] When the energy is only defined for the part that depends on $\boldsymbol{x}$, i.e., by separating the posterior according to $\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x}) = \frac{1}{Z} f(\boldsymbol{x})$ and defining $U(\boldsymbol{x}) = -\log f(\boldsymbol{x})$, it is $F_{\mathrm{V}}(\mathsf{q}(\boldsymbol{x})) = -\log Z + \mathrm{D}_{\mathrm{KL}}(\mathsf{q}(\boldsymbol{x})||\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x}))$.

($\beta \to \infty$), the distribution bundles up around the state $\boldsymbol{x}$ that minimizes $U(\boldsymbol{x})$, i.e., the system converges to a state of minimum energy. If the energy represents the underlying system via the specification of $U(\boldsymbol{x})$ as in (2.38) the minimization of the energy

$$\hat{\boldsymbol{x}} = \operatorname*{argmin}_{\boldsymbol{x}} U(\boldsymbol{x}) \tag{2.40}$$

is equivalent to the MAP estimate (2.6).

Noteworthy, by defining the energy of the problem via

$$U(\boldsymbol{x}) = -\log\Big(\mathsf{f}_{\mathsf{x}}(\boldsymbol{x}) \cdot \mathsf{f}_{\mathsf{y}|\mathsf{x}}(\boldsymbol{x})\Big) \tag{2.41}$$

the problem of minimizing the energy becomes

$$\min_{\boldsymbol{x}} \frac{1}{2\sigma_{\mathsf{n}}^2} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 - \log \mathsf{f}_{\mathsf{x}}(\boldsymbol{x}) \ , \tag{2.42}$$

which itself for Laplacian prior with variance σ^2, given by $\mathsf{f}_{\mathsf{x}}(x) = \frac{1}{\sqrt{2}\sigma} \exp\Big(-\frac{\sqrt{2}}{\sigma}|x|\Big)$, yields so-called basis pursuit denoising (BPDN) [CDS01], also known as LASSO [Tib96][9]

$$\min_{\boldsymbol{x}} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \lambda \|\boldsymbol{x}\|_1 \tag{2.43}$$

with tuning parameter $\lambda = 2\sqrt{2}\frac{\sigma_{\mathsf{n}}^2}{\sigma}$ and $\|\cdot\|_1$ being the so-called ℓ_1-norm, for the definition, see Appendix A.

2.3 Structure of the Compressed Sensing Problem

Now the conditional pdf (2.7) is considered, which can be factorized in different ways. With

$$\mathsf{f}_{\mathsf{x}|\mathsf{y}}(\boldsymbol{x}) = \frac{1}{Z} \mathsf{f}_{\mathsf{x}}(\boldsymbol{x}) \cdot \mathsf{f}_{\mathsf{y}|\mathsf{x}}(\boldsymbol{x}) \ , \tag{2.44}$$

where

$$\mathsf{f}_{\mathsf{x}}(\boldsymbol{x}) = \prod_{j=1}^{N} \mathsf{f}_{\mathsf{x}}(x_j) \tag{2.45}$$

is the signal part,

$$\begin{aligned} \mathsf{f}_{\mathsf{y}|\mathsf{x}}(\boldsymbol{x}) &= \frac{1}{\sqrt{(2\pi\sigma_{\mathsf{n}}^2)^M}} \exp\left(-\frac{1}{2\sigma_{\mathsf{n}}^2} \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2\right) \\ &= \prod_{i=1}^{M} \frac{1}{\sqrt{2\pi\sigma_{\mathsf{n}}^2}} \exp\left(-\frac{1}{2\sigma_{\mathsf{n}}^2}(y_i - \boldsymbol{a}_i^\top \boldsymbol{x})^2\right) \overset{\text{def}}{=} \prod_{i=1}^{M} \mathsf{f}_{\mathsf{y}_i|\mathsf{x}}(\boldsymbol{x}) \ , \end{aligned} \tag{2.46}$$

the channel part, and $Z = \mathsf{f}_{\mathsf{y}}(\boldsymbol{y})$ serves as normalization, the problem can be split into two parts that are connected via the signal $\boldsymbol{x}$. Note that signal and channel part can itself be factorized as given.

[9] LASSO stands for least absolute shrinkage and selection operator.

2.3.1 Factor Graph Representations

A graphical representation of the structure of the posterior is given as *factor graph* [KFL01] in Fig. 2.3. In the upper left, the detailed structure of the problem with the factors broken down to the most fragmented factorization, representing $\mathsf{f}_{\mathsf{x}|\mathsf{y}}(\boldsymbol{x}) = \frac{1}{Z}\prod_{i=1}^{M}\mathsf{f}_{\mathsf{y}_i|\mathsf{x}}(\boldsymbol{x})\prod_{j=1}^{N}\mathsf{f}_{\mathsf{x}}(x_j)$, is shown. Factors are represented by squares, the *variable nodes* by circles. The connections, represented by lines between the nodes, indicate dependencies between the nodes, i.e., variable nodes connected to a factor node mean that the respective variables are arguments of the factor. One can see that each factor of the channel part is connected to all variables. On the other hand, the signal factors only depend on one corresponding variable. The lower right factor graph shows the splitting according to (2.44), which focuses on the fact that there are only two different kinds of factors. In order to emphasize that the factor nodes here are combined versions of the factors in the upper left, a different representation is used. Furthermore, combined variable nodes are represented by a double circle.

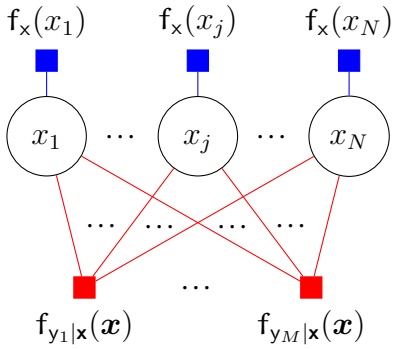
Additionally, mixtures of both extreme cases are possible. The lower left shows a representation that combines the channel parts into $\mathsf{f}_{\mathsf{y}|\mathsf{x}}(\boldsymbol{x})$, but keeps the signal parts as factors, thereby causing distinct variable nodes. Whereas, on the upper right, the opposite is shown with combined signal parts and individual channel parts $\mathsf{f}_{\mathsf{y}_i|\mathsf{x}}(\boldsymbol{x})$ $(i \in \{1, \ldots, M\})$. Noteworthy, in this last case, the variable nodes are combined because the channel parts depend on the entire signal vector $\boldsymbol{x}$. According to the denomination we may refer to the respective nodes as channel- or signal-constrained (factor) nodes.

2.3.2 Representatives for the Factors

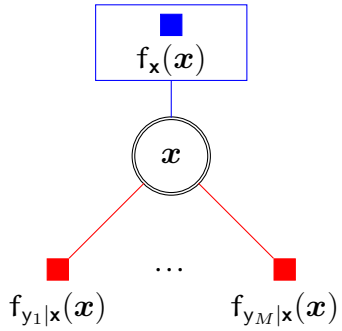
The approximation of $\mathsf{f}_{\mathsf{x}|\mathsf{y}}(\boldsymbol{x})$ by the substitute distribution $\mathsf{q}(\boldsymbol{x})$ needs to represent the structure of the problem suitably. To that end, distributions are defined that represent the substitute distribution at certain nodes. The representation for the entire channel part $\mathsf{f}_{\mathsf{y}|\mathsf{x}}(\boldsymbol{x})$ is called $\mathsf{q}_{\mathrm{c}}(\boldsymbol{x})$, whereas if only the dependency of $\boldsymbol{x}$ on a single observation, i.e., $\mathsf{f}_{\mathsf{y}_i|\mathsf{x}}(\boldsymbol{x})$, is considered the representative is denoted by $\mathsf{q}_{\mathrm{c},i}(\boldsymbol{x})$. The signal part $\mathsf{f}_{\mathsf{x}}(\boldsymbol{x})$ is represented by $\mathsf{q}_{\mathrm{s}}(\boldsymbol{x})$ and the combined variable node by $\mathsf{q}_{\mathrm{v}}(\boldsymbol{x})$, respectively. In contrast to $\mathsf{q}_{\mathrm{c}}(\boldsymbol{x})$ and $\mathsf{q}_{\mathrm{s}}(\boldsymbol{x})$, the representative of the combined variable node $\mathsf{q}_{\mathrm{v}}(\boldsymbol{x})$ cannot be found in the formula for the posterior $\mathsf{f}_{\mathsf{x}|\mathsf{y}}(\boldsymbol{x})$, indeed its existence is only motivated by the factor graph representation. The task of the (combined) variable node $\boldsymbol{x}$ is to force equality between the arguments of the factors $\mathsf{f}_{\mathsf{y}|\mathsf{x}}(\boldsymbol{x})$ and $\mathsf{f}_{\mathsf{x}}(\boldsymbol{x})$ in (2.44), i.e., if they are treated individually, one has to make sure that the argument is the same. This behavior can be described by the Dirac delta function, i.e., one can write [LDH+07]

$$\mathsf{f}_{\mathsf{x}|\mathsf{y}}(\boldsymbol{x}) = \text{const.} \cdot \int \mathsf{f}_{\mathsf{x}}(\tilde{\boldsymbol{x}})\mathsf{f}_{\mathsf{y}|\mathsf{x}}(\boldsymbol{x})\delta(\boldsymbol{x}-\tilde{\boldsymbol{x}})\,\mathrm{d}\tilde{\boldsymbol{x}}\ . \tag{2.47}$$

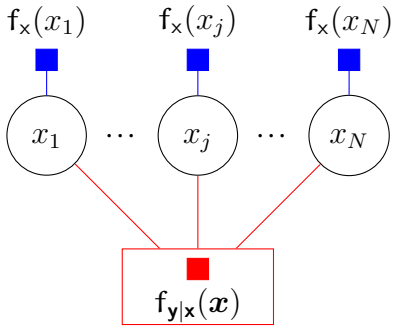
This means, the Dirac delta function can be seen as the function of the factor behind the variable node. Nevertheless, we will keep the denomination for variable nodes for the distinguishability to the actual factor nodes. The derivations in subsequent chapters will show that the representative for the variable nodes $\mathsf{q}_{\mathrm{v}}(\boldsymbol{x})$ softens the equality constraint by allowing also variations in the factors.



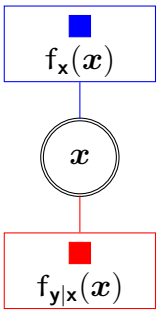
(a) Detailed view on all individual factors.



(b) Combined variable node, detailed view on channel-constrained factor nodes.

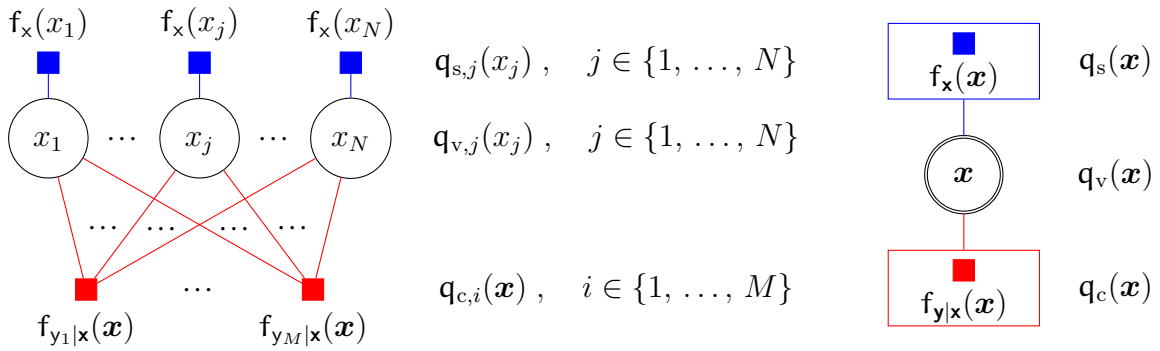


(c) Detailed view on variable nodes, combined channel-constrained factor nodes.



(d) Focus on the two different kinds of factors.

Figure 2.3: Several representations of the factor graph for the conditional pdf of the CS problem (2.7).



(a) Detailed factorization. (b) Two-part factorization.

Figure 2.4: Factor graphs for different factorizations with respective representatives.

Considering the individual variable nodes and individual signal factor nodes as, e.g., in Fig. 2.3(a), requires also representatives for these (sub)nodes. As these representatives do not depend on all variables anymore, one needs to get rid of them. This is done by *marginalization* [KFL01, LDH+07]. A marginal of any distribution is in general obtained for a variable x_j by integrating (or summing in case of discrete random variables) over all elements of $\boldsymbol{x}$ that are not x_j, i.e., to obtain a marginal for a single variable at a variable node, one needs to compute

$$\mathsf{q}_{\mathrm{v},j}(x_j) = \int \cdots \int \mathsf{q}_{\mathrm{v}}(\boldsymbol{x})\, \mathrm{d}x_1 \ldots \mathrm{d}x_{j-1} \mathrm{d}x_{j+1} \ldots \mathrm{d}x_N\,, \quad j \in \{1, \ldots, N\}\,. \tag{2.48}$$

Analogously, the marginal at a single signal factor $\mathsf{f}_{\mathsf{x}}(x_j)$ is

$$\mathsf{q}_{\mathrm{s},j}(x_j) = \int \cdots \int \mathsf{q}_{\mathrm{s}}(\boldsymbol{x})\, \mathrm{d}x_1 \ldots \mathrm{d}x_{j-1} \mathrm{d}x_{j+1} \ldots \mathrm{d}x_N\,, \quad j \in \{1, \ldots, N\}\,. \tag{2.49}$$

With the marginals $\mathsf{q}_{\mathrm{v},j}(x_j)$ and $\mathsf{q}_{\mathrm{s},j}(x_j)$ the connection between x_j and $\mathsf{f}_{\mathsf{x}}(x_j)$ can be expressed, as will be seen later in Chapter 3.

Together with $\mathsf{q}_{\mathrm{v},j}(x_j)$, the marginals of $\mathsf{q}_{\mathrm{c},i}(\boldsymbol{x})$ can be used to specify the connections between x_j and the channel part $\mathsf{f}_{\mathsf{y}_i|\mathsf{x}}(\boldsymbol{x})$. They are denoted and calculated by

$$\mathsf{q}_{\mathrm{c},i}(x_j) = \int \cdots \int \mathsf{q}_{\mathrm{c},i}(\boldsymbol{x})\, \mathrm{d}x_1 \ldots \mathrm{d}x_{j-1} \mathrm{d}x_{j+1} \ldots \mathrm{d}x_N\,, \quad j \in \{1, \ldots, N\}\,. \tag{2.50}$$

In this last case, it is clearly visible that the factor marginals are edge-dependent, as one marginal specifies the connecting edge between corresponding factor and variable node.

To get the denomination of the different marginals straight, two of the factor graph figures are repeated in Fig. 2.4 to show the correspondences. Furthermore, the effect of marginalizing a distribution is shown in Ex. 2.1.

Example 2.1:

As an example, $\mathsf{f}_{\mathsf{y}|\mathsf{x}}(\boldsymbol{x})$ from (2.46) of a two-dimensional CS scenario is considered as a function of $\boldsymbol{x}$ only. To that end, let us define

$$\mathsf{p}_{\boldsymbol{y}}(\boldsymbol{x}) \overset{\text{def}}{=} \frac{1}{\sqrt{(2\pi\sigma_{\mathsf{n}}^2)^2}} \exp\left(-\frac{1}{2\sigma_{\mathsf{n}}^2} \left\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\right\|_2^2\right)\,, \tag{2.51}$$

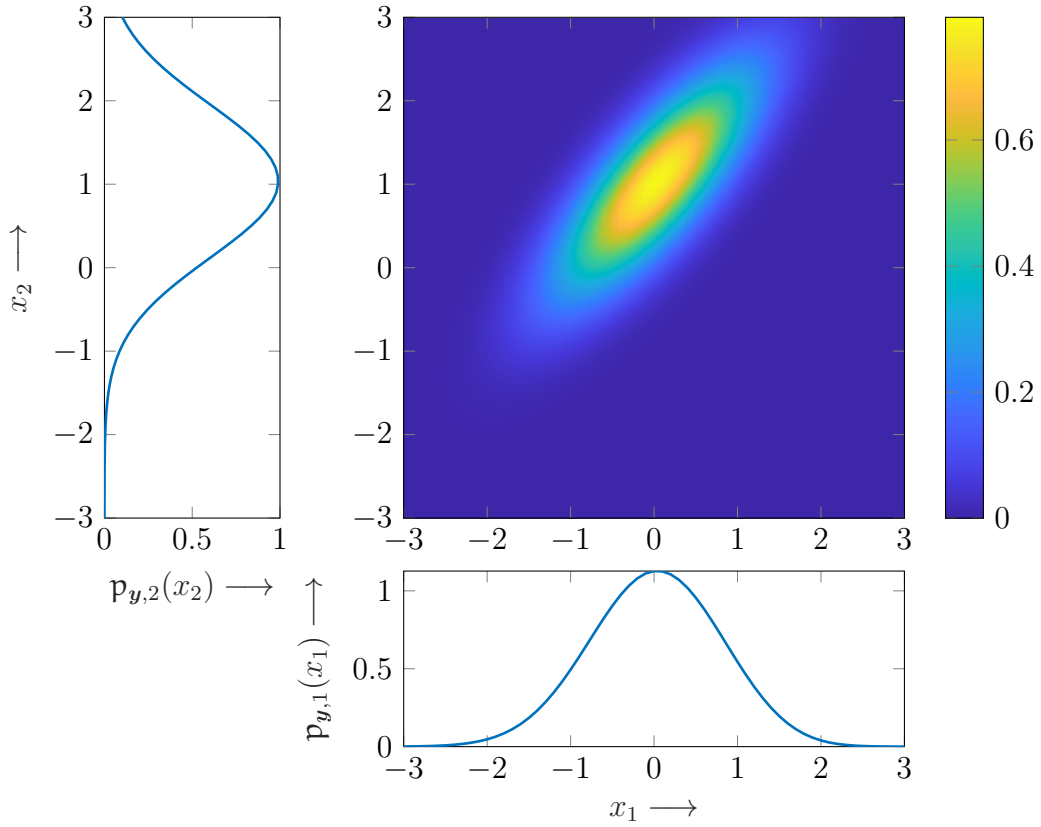


Figure 2.5: Contour plot of $\mathrm{p}_{\boldsymbol{y}}(\boldsymbol{x})$ of Ex. 2.1 and respective marginals.

where $\boldsymbol{y} = [-0.23,\, 0.76]^\top$ is obtained from compressing the signal $\boldsymbol{x} = [0,\, 1]^\top$, with the sensing matrix

$$\boldsymbol{A} = \begin{bmatrix} 0.75 & -0.25 \\ -0.5 & 0.75 \end{bmatrix} \tag{2.52}$$

and adding Gaussian noise with variance $\sigma_{\mathsf{n}}^2 = 0.2$. The two-dimensional pdf $\mathrm{p}_{\boldsymbol{y}}(\boldsymbol{x})$ is shown as a contour plot in Fig. 2.5, surrounded by the marginals which are here computed by

$$\mathrm{p}_{\boldsymbol{y},1}(x_1) = \int \mathrm{p}_{\boldsymbol{y}}(\boldsymbol{x})\,\mathrm{d}x_2 \tag{2.53}$$

and

$$\mathrm{p}_{\boldsymbol{y},2}(x_2) = \int \mathrm{p}_{\boldsymbol{y}}(\boldsymbol{x})\,\mathrm{d}x_1\ , \tag{2.54}$$

respectively. The integration removes the dependency of the variable that is integrated over. Noteworthy, the pdf $\mathrm{p}_{\boldsymbol{y}}(\boldsymbol{x})$ has a unique maximum, since the sensing matrix is square. In a scenario, where the signal is actually compressed, e.g., by considering only $y_1 = \boldsymbol{a}_1^\top \boldsymbol{x} + \mathsf{n}_1$ with $y_1 = -0.23$, $\boldsymbol{a}_1 = [0.75,\, -0.25]^\top$ and $\mathsf{n}_1 \sim \mathcal{N}(0,\, \sigma_{\mathsf{n}}^2)$, the contour plot shows a Gaussian distribution around the line defined by

$$y_1 = -0.23 = \boldsymbol{a}_1^\top \boldsymbol{x} = a_{11}x_1 + a_{12}x_2 = 0.75x_1 - 0.25x_2 \tag{2.55}$$

$$\Leftrightarrow \quad x_2 = -4 \cdot (y_1 - 0.75x_1) = 3x_1 + 0.92\ . \tag{2.56}$$

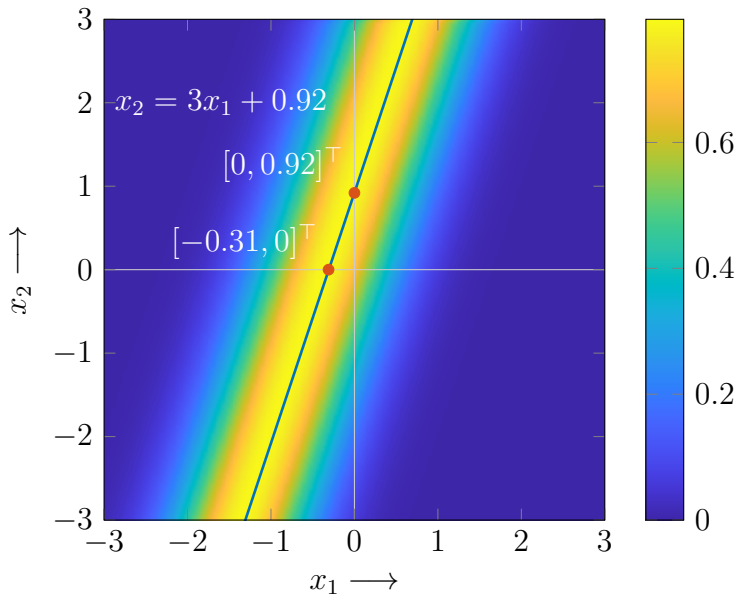


Figure 2.6: Contour plot of $\mathrm{p}_{y_1}(\boldsymbol{x})$ of Ex. 2.1, the line $x_2 = 3x_1 + 0.92$, specifying the maximum of the pdf, and two possible sparse solutions.

All points on that line maximize the pdf and may therefore be seen as solution. Setting either of the signal values x_1 or x_2 to zero, recovers a sparse solution, e.g., $x_1 = 0$ leads to $x_2 = 0.92$. The pdf that corresponds to the single measurement scenario

$$\mathrm{p}_{y_1}(\boldsymbol{x}) = \frac{1}{\sqrt{2\pi\sigma_{\mathsf{n}}^2}} \exp\left(-\frac{1}{2\sigma_{\mathsf{n}}^2}(y_1 - \boldsymbol{a}_1^\top \boldsymbol{x})^2\right) \tag{2.57}$$

is depicted in Fig. 2.6 as contour plot. Additionally the line following the maximum and the two sparse solutions are shown.

3. Inference on Factor Graphs

The use of factor graphs [Loe04, For01] and other graphical models for inference was exploited in several fields of engineering already. There is to mention the decoding of LDPC codes via Tanner graphs [Tan81], or algebraic codes via Ising-type models [For18], other decoding strategies, such as the BCJR or the Viterbi algorithm [Loe04], multiuser detection in CDMA [GW06] and several aspects in signal processing [LDH+07], as, e.g., Kalman filtering or recursive least-squares algorithms. The underlying algorithm is called *belief propagation* (BP) [Pea88] or *message passing* (MP) and may adopt certain forms; for an overview see [KFL01].

In the following, an optimization problem with constraints that are imposed by the structure of the considered factor graph is set up. Based on that, a form of (Gaussian) message passing for the CS problem with factor graph representation in Fig. 2.3(a) as basis is derived. The resulting algorithm is equal to the one stated in [KMS+12, Sec. 3.1], however, the derivation differs from the one considered here.

An optimization-based view has been used in [YFW05] to show the connection between MP and the so-called *Bethe free energy* [Bet35]. A similar derivation is given in [HOW+05], where approximate inference with expectation constraints is considered for general inference problems. An overview over signal processing strategies encompassing, next to MP and the Bethe approach, also AMP and the EP framework for general inference is given in [PSC+15]. Here, we follow the approaches given in [HOW+05, YFW05], but focus on the CS problem.

Noteworthy, the optimization-based derivation reveals that the notion of *extrinsic* [BGT93, HOP96] is inherently given in this approach to the solution of the problem.

3.1 Structure of the Factor Graph

In Chapter 2 it has been pointed out that the high dimensionality of $\boldsymbol{x}$ makes it difficult to recover the signal from the observations. As a first approach, the computation is simplified as much as possible by breaking down the structure to its basic components. To that end, the representation for the CS problem that factorizes the problem as much as possible, cf. Fig. 2.3(a), is regarded. With this fine-granular view, the multi-dimensional estimation is split

into several one-dimensional ones, which simplifies the estimations the most. In the following, this structure will be used to derive a suitable representation of the approximation for the factorization.

3.1.1 Factorization of the Substitute Distribution

First, it is important to find a suitable form for the substitute $\mathsf{q}(\boldsymbol{x})$ that should be worked with. The substitute distribution $\mathsf{q}(\boldsymbol{x})$, which is used to approximate the posterior $\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})$, should inherit the structure of the factor graph in Fig. 2.3(a). To that end, the corresponding factorization is considered as a function of $\boldsymbol{x}$. In order to allow variations in the factors, the product is expanded by functions of $\boldsymbol{x}$ for each factor, i.e.,

$$\begin{aligned}
\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x}) &= \text{const.} \cdot \prod_{j=1}^{N} \mathsf{f}_{\mathsf{x}}(x_j) \prod_{i=1}^{M} \mathsf{f}_{\mathsf{y}_i|\mathbf{x}}(\boldsymbol{x}) \\
&= \text{const.} \cdot \prod_{j=1}^{N} \mathsf{f}_{\mathsf{x}}(x_j) \prod_{i=1}^{M} \mathsf{f}_{\mathsf{y}_i|\mathbf{x}}(\boldsymbol{x}) \cdot \frac{\prod_{j=1}^{N} s_j(x_j) \prod_{i=1}^{M} c_i(\boldsymbol{x})}{\prod_{j=1}^{N} s_j(x_j) \prod_{i=1}^{M} c_i(\boldsymbol{x})} \\
&= \text{const.} \cdot \frac{\prod_{j=1}^{N} \mathsf{f}_{\mathsf{x}}(x_j) s_j(x_j) \prod_{i=1}^{M} \mathsf{f}_{\mathsf{y}_i|\mathbf{x}}(\boldsymbol{x}) c_i(\boldsymbol{x})}{\prod_{j=1}^{N} s_j(x_j) \prod_{i=1}^{M} c_i(\boldsymbol{x})} \,. \qquad (3.1)
\end{aligned}$$

The association of the factors and the respective functions may now be interpreted as the corresponding representatives from Sec. 2.3.2, i.e.,

$$\mathsf{q}_{\mathrm{c},i}(\boldsymbol{x}) = \mathsf{f}_{\mathsf{y}_i|\mathbf{x}}(\boldsymbol{x}) \cdot c_i(\boldsymbol{x}) \,, \qquad i \in \{1, \ldots, M\} \,, \qquad (3.2)$$

$$\mathsf{q}_{\mathrm{s},j}(x_j) = \mathsf{f}_{\mathsf{x}}(x_j) \cdot s_j(x_j) \,, \qquad j \in \{1, \ldots, N\} \,. \qquad (3.3)$$

Assuming that the function for the channel-constrained approximation split as follows

$$c_i(\boldsymbol{x}) = \prod_{j=1}^{N} c_{ij}(x_j) \,, \qquad i \in \{1, \ldots, M\} \,, \qquad (3.4)$$

one can write

$$\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x}) = \text{const.} \cdot \frac{\prod_{j=1}^{N} \mathsf{q}_{\mathrm{s},j}(x_j) \prod_{i=1}^{M} \mathsf{q}_{\mathrm{c},i}(\boldsymbol{x})}{\prod_{j=1}^{N} \left(s_j(x_j) \prod_{i=1}^{M} c_{ij}(x_j) \right)} \,. \qquad (3.5)$$

This way, the sorting in the denominator is chosen such that functions of the same variable x_j ($j \in \{1, \ldots, N\}$) are packed together. These clustered variable-dependent functions can now be set in relation with the representatives $\mathsf{q}_{\mathrm{v},j}(x_j)$ ($j \in \{1, \ldots, N\}$) that were chosen for the variable nodes.

The expansion of the product by the introduced deviations may be visualized by attaching them as factors (in numerator and denominator) to the corresponding variable nodes, cf. Fig. 3.1. The approximations $\mathsf{q}_{\mathrm{c},i}(\boldsymbol{x})$ and $\mathsf{q}_{\mathrm{s},j}(x_j)$ are then obtained by grouping the factors correspondingly.

Recall, that the distributions representing the variable nodes are Dirac delta functions, cf. (2.47), i.e., the equality between the arguments of the different functions is obtained by the

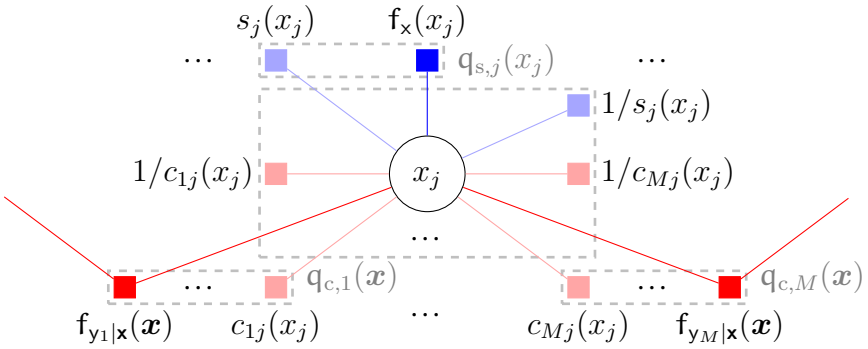


Figure 3.1: Visualization of the grouping of the factors to allow for variations.

integration over respective Dirac functions. The central box in Fig. 3.1 can, thus, be described by

$$\frac{\text{const.}}{s_j(x_j)\prod_{i=1}^{M} c_{ij}(x_j)} = \text{const.} \cdot \int \cdots \int \frac{\prod_{i=1}^{M} \delta(x_j - x_{i,j})}{s_j(x_j)\prod_{i=1}^{M} c_{ij}(x_{i,j})} \, \mathrm{d}x_{1,j} \ldots x_{M,j} \;, \tag{3.6}$$

where the slack variables $x_{1,j}, \ldots, x_{M,j}$ are introduced to indicate the arguments of the different deviations. The argument of $s_j(x_j)$ is chosen to be x_j for convenience, so that the usual argument is recovered after the integration.

It is important to note that the M-fold integral on the right-hand side of (3.6) does not describe the effect of the variable node on other factors as in (2.47) but rather represents an internal behavior. In Chapter 2, variable nodes have been defined by their (integral) effect of the Dirac delta function (cf. (2.47)) on other factors, i.e., they do not have an internal behavior. Therefore, the internal behavior that is present on the right-hand side in (3.6) needs to be compensated. If the integral were separable, it would constitute an M-fold multiplication, i.e., the multiplication of M functions. However, the marginal $\mathsf{q}_{\mathrm{v},j}(x_j)$ for the variable node x_j represents one function. When drawing the connection between marginal $\mathsf{q}_{\mathrm{v},j}(x_j)$ and (3.6), this inherent multiplication is therefore compensated by taking the Mth root, i.e.,

$$\mathsf{q}_{\mathrm{v},j}(x_j) = \text{const.} \cdot \sqrt[M]{s_j(x_j)\prod_{i=1}^{M} c_{ij}(x_j)} \;, \qquad j \in \{1, \ldots, N\} \;. \tag{3.7}$$

Note that the denominator is represented directly for better readability. Interestingly, the approach of allowing variations in the factors led to the effect of the Dirac delta function being replaced by corresponding variations, which means that the equality between the arguments of different factors is not as strict as before. This way, the entire substitute reads [YFW05]

$$\mathsf{q}(\boldsymbol{x}) = \text{const.} \cdot \frac{\prod_{j=1}^{N} \mathsf{q}_{\mathrm{s},j}(x_j) \prod_{i=1}^{M} \mathsf{q}_{\mathrm{c},i}(\boldsymbol{x})}{\left(\prod_{j=1}^{N} \mathsf{q}_{\mathrm{v},j}(x_j)\right)^{M}} \;. \tag{3.8}$$

Note that since the interest lies in the argument $\boldsymbol{x}$ of the function, the constant factor in the formula does not harm the processing [LDH+07]; it is therefore not stated in subsequent derivations.

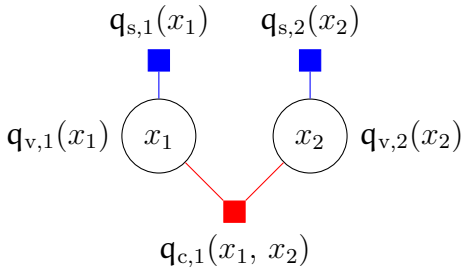


Figure 3.2: Factor graph and representatives for Ex. 2.1 with one observation y_1.

This factorization represents the most fine-granular view possible, as every individual factor and variable node is represented by its own marginal distribution. According to [YFW05], this factorization is the underlying principle of the Bethe free energy, that was shown to be connected to the BP algorithm. The procedure amounts to the simple rule that for each variable x_j ($j \in \{1, \ldots, N\}$) that occurs "too often" in the numerator, one representative $\mathsf{q}_{\mathrm{v},j}(x_j)$ occurs in the denominator as compensation. In the given case, each variable x_j is part of $\mathsf{q}_{\mathrm{s},j}(x_j)$ as well as the M channel-constrained representatives, i.e., the node x_j has $M+1$ neighbors. Therefore, $(M+1)-1 = M$ instances of $\mathsf{q}_{\mathrm{v},j}(x_j)$ appear in the denominator.

An example computation is conducted in Ex. 3.1.

Example 3.1:

For Ex. 2.1, when considering only one observation y_1 for the two signal components x_1 and x_2, the factor graph, shown in Fig. 3.2, consists of three factors and two variable nodes, which are connected by the single channel-constrained factor, i.e., the observation. Instead of the factors, the representatives of the notation above are placed at the respective positions. Since the multiplication of $\mathsf{q}_{\mathrm{s},1}(x_1)$, $\mathsf{q}_{\mathrm{s},2}(x_1)$, and $\mathsf{q}_{\mathrm{c},1}(x_1,\, x_2)$ contains each of the variables x_1 and x_2 two times, one needs to divide by $\mathsf{q}_{\mathrm{v},1}(x_1)$ and $\mathsf{q}_{\mathrm{v},2}(x_2)$, which results in an overall approximation

$$\mathsf{q}(\boldsymbol{x}) = \frac{\mathsf{q}_{\mathrm{s},1}(x_1)\mathsf{q}_{\mathrm{s},2}(x_2)\mathsf{q}_{\mathrm{c},1}(x_1,\, x_2)}{\mathsf{q}_{\mathrm{v},1}(x_1)\mathsf{q}_{\mathrm{v},2}(x_2)} \,. \tag{3.9}$$

3.1.2 The Moment-Matching Constraint

As already mentioned, a suitable solution $\mathsf{q}(\boldsymbol{x})$ must represent the structure of the problem correctly. In this fine-granular view, this means that the marginals of respective nodes must be consistent, when following the connecting edges. Consistency, here, means that there is no contradiction to neighboring factor nodes, when marginalizing the respective pdfs to the variable under consideration. In [HOW+05] this constraint is called *local consistency condition*. For the marginals at hand, this reads

$$\mathsf{q}_{\mathrm{c},i}(x_j) \overset{!}{=} \mathsf{q}_{\mathrm{v},j}(x_j) \,, \quad j \in \{1, \ldots, N\} \,, \quad i \in \{1, \ldots, M\} \,, \tag{3.10}$$

$$\mathsf{q}_{\mathrm{s},j}(x_j) \overset{!}{=} \mathsf{q}_{\mathrm{v},j}(x_j) \,, \quad j \in \{1, \ldots, N\} \,. \tag{3.11}$$

The resulting constrained optimization problem leads to the famous *message passing* (MP) algorithm, also known as (loopy) *belief propagation* (BP)[1] [Pea88]. The algorithm propagates pdfs as messages to update the so-called *beliefs*, which are approximates for the exact marginals of the signal $\boldsymbol{x}$ [YFW05].

Since this is a rather unhandy representation, this step is skipped and it is directly worked with moments instead of the distributions, which is equal to restricting the marginals to some *exponential family* [Bro86][2]. This is a class of distributions that is entirely determined by certain moments; the respective moments are specified by the so-called *sufficient statistics* $\boldsymbol{g}(\boldsymbol{x})$. For a given sufficient statistics, an exponential family is parameterized by the so-called *natural parameters*, which is here denoted by $\boldsymbol{\theta}$. Noteworthy, there is a direct dependency between the moments specified by the sufficient statistics and the natural parameters; examples are given in Appendix D. An exponential family has, consequently, the beneficial property of allowing to transfer the computations w.r.t. distributions into a parameter space, i.e., instead of handling the distribution, one can compute estimates and process them. Restricting to the Gaussian distribution, which requires only first and second order moments for its statistic to be specified, will lead to so-called *Gaussian BP* [KMS+12]. For the derivation thereof, the sufficient statistics $\boldsymbol{g}(\boldsymbol{x})$ of the respective exponential family are fixed to (D.11), i.e.,

$$\boldsymbol{g}(\boldsymbol{x}) = [\boldsymbol{g}_{\boldsymbol{\lambda}}(\boldsymbol{x})^\top, \boldsymbol{g}_{\boldsymbol{\Lambda}}(\boldsymbol{x})^\top]^\top \tag{3.12}$$

with

$$\boldsymbol{g}_{\boldsymbol{\lambda}}(\boldsymbol{x}) = [x_1, \ldots, x_N]^\top \,, \quad \boldsymbol{g}_{\boldsymbol{\Lambda}}(\boldsymbol{x}) = -\frac{1}{2}[x_1^2, \ldots, x_N^2]^\top \,, \tag{3.13}$$

and respective natural parameter $\boldsymbol{\theta} = [\boldsymbol{\lambda}^\top, \boldsymbol{\Lambda}^\top]^\top$. In order to specify the moment of a single variable, we may use the notation $\boldsymbol{g}(x_j) = [g_{\lambda_j}(x_j), g_{\Lambda_j}(x_j)]^\top = [x_j, -x_j^2/2]^\top$.

The respective constraints, ensuring the representation of the structure of the CS posterior $\mathsf{f}_{\mathsf{x}|\mathsf{y}}(\boldsymbol{x})$, is called *weak consistency condition* in [HOW+05] and can be stated by using the marginals and the specified sufficient statistics for $j \in \{1, \ldots, N\}$

$$\mathrm{E}_{\mathsf{x}_j \sim \mathsf{q}_{\mathrm{v},j}}\{\boldsymbol{g}(\mathsf{x}_j)\} \stackrel{!}{=} \mathrm{E}_{\mathsf{x}_j \sim \mathsf{q}_{\mathrm{c},i}}\{\boldsymbol{g}(\mathsf{x}_j)\} \,, \quad i \in \{1, \ldots, M\} \,, \tag{3.14}$$

respectively,

$$\mathrm{E}_{\mathsf{x}_j \sim \mathsf{q}_{\mathrm{v},j}}\{\boldsymbol{g}(\mathsf{x}_j)\} \stackrel{!}{=} \mathrm{E}_{\mathsf{x}_j \sim \mathsf{q}_{\mathrm{s},j}}\{\boldsymbol{g}(\mathsf{x}_j)\} \,. \tag{3.15}$$

These requirements will be referred to as the *moment-matching constraints*.

3.2 Constrained Optimization

In the following, the task of approximating $\mathsf{f}_{\mathsf{x}|\mathsf{y}}(\boldsymbol{x})$ suitably is described as constrained optimization problem for the given structure and the stationary points of the respective Lagrangian is sought, which eventually is used to derive a recovery algorithm for the CS problem.

[1] In the given case the BP algorithm is loopy because the factor graph is not a tree, i.e., has cycles.

[2] The necessary knowledge about exponential families is summarized in Appendix D.

To that end, the idea of minimizing the Kullback–Leibler divergence, which was motivated in Sec. 2.2.2, is seized and combined with the moment-matching constraints (3.14) and (3.15) of the fine-granular view on the factor graph. This way, a constrained optimization problem is obtained that reads

$$\begin{aligned}
&\text{minimize} && \mathrm{D}_{\mathrm{KL}}\Big(\mathsf{q}(\boldsymbol{x}) \,||\, \mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})\Big) \\
&\text{subject to} && \mathrm{E}_{\mathsf{x}_j\sim\mathsf{q}_{\mathrm{v},j}}\{\boldsymbol{g}(\mathsf{x}_j)\} - \mathrm{E}_{\mathsf{x}_j\sim\mathsf{q}_{\mathrm{c},i}}\{\boldsymbol{g}(\mathsf{x}_j)\} = \mathbf{0}\,, \quad && j \in \{1,\,\ldots,\,N\}\,,\, i \in \{1,\,\ldots,\,M\}\,, \\
& && \mathrm{E}_{\mathsf{x}_j\sim\mathsf{q}_{\mathrm{v},j}}\{\boldsymbol{g}(\mathsf{x}_j)\} - \mathrm{E}_{\mathsf{x}_j\sim\mathsf{q}_{\mathrm{s},j}}\{\boldsymbol{g}(\mathsf{x}_j)\} = \mathbf{0}\,, \quad && j \in \{1,\,\ldots,\,N\}\,, \\
& && \int \mathsf{q}_{\mathrm{v},j}(x_j)\,\mathrm{d}x_j = 1\,,\ \int \mathsf{q}_{\mathrm{s},j}(x_j)\,\mathrm{d}x_j = 1\,, \quad && j \in \{1,\,\ldots,\,N\}\,, \\
& && \int \mathsf{q}_{\mathrm{c},i}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = 1\,, && i \in \{1,\,\ldots,\,M\}\,.
\end{aligned} \tag{3.16}$$

3.2.1 Cost Function for Compressed Sensing

Before considering the optimization itself, the Kullback–Leibler divergence to be minimized is stated, utilizing the structure imposed by CS. Inserting the factorization (3.8) into the Kullback–Leibler divergence yields a certain form, because the term disintegrates to a sum of parts depending solely on the factors, respectively variable marginals due to the normalization property of pdfs. The result to be minimized in the above optimization problem reads

$$\begin{aligned}
\mathrm{D}_{\mathrm{KL}}\Big(\mathsf{q}(\boldsymbol{x}) \,||\, \mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})\Big) = &\sum_{j=1}^{N} \int \mathsf{q}_{\mathrm{s},j}(x_j) \log \mathsf{q}_{\mathrm{s},j}(x_j)\,\mathrm{d}x_j + \sum_{i=1}^{M} \int \mathsf{q}_{\mathrm{c},i}(\boldsymbol{x}) \log \mathsf{q}_{\mathrm{c},i}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \\
&- \sum_{j=1}^{N} \int \mathsf{q}_{\mathrm{s},j}(x_j) \log \mathsf{f}_{\mathsf{x}}(x_j)\,\mathrm{d}x_j - \sum_{i=1}^{M} \int \mathsf{q}_{\mathrm{c},i}(\boldsymbol{x}) \log \mathsf{f}_{\mathsf{y}_i|\mathbf{x}}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \\
&- M \sum_{j=1}^{N} \int \mathsf{q}_{\mathrm{v},j}(x_j) \log \mathsf{q}_{\mathrm{v},j}(x_j)\,\mathrm{d}x_j\,.
\end{aligned} \tag{3.17}$$

This form of the Kullback–Leibler divergence is called *Bethe free energy* in [YFW05].

3.2.2 Stationary Points of the Lagrangian

Constrained optimization problems can be dealt with by considering the so-called *Lagrangian*, which combines the optimization task and the constraints in one function(al), cf. Appendix B. As motivated in Appendix B, the optimizer of the problem needs to fulfill a stationary condition of the Lagrangian, which means the stationary points of the Lagrangian are of interest for us to be achieved. By introducing the Lagrangian Multipliers $\boldsymbol{\nu}_{\mathrm{c},ij}$ and $\boldsymbol{\nu}_{\mathrm{s},jj}$ for the moment-matching constraints, as well as ν_j, $\nu_{\mathrm{c},i}$, and $\nu_{\mathrm{s},j}$ as normalization constraint to ensure valid marginals, the Lagrangian reads (omitting the Lagrangian multipliers in the argument

for brevity)

$$\begin{aligned}\mathcal{L}(\mathsf{q}(\boldsymbol{x})) = \mathrm{D}_{\mathrm{KL}}\Big(\mathsf{q}(\boldsymbol{x}) \mid\mid \mathsf{f}_{\mathsf{x}|\mathsf{y}}(\boldsymbol{x})\Big) &+ \sum_{i=1}^{M}\sum_{j=1}^{N} \boldsymbol{\nu}_{\mathrm{c},ij}^{\top}\Big(\mathrm{E}_{\mathsf{x}_j\sim\mathsf{q}_{\mathrm{v},j}}\{\boldsymbol{g}(\mathsf{x}_j)\} - \mathrm{E}_{\mathsf{x}_j\sim\mathsf{q}_{\mathrm{c},i}}\{\boldsymbol{g}(\mathsf{x}_j)\}\Big) \\ &+ \sum_{j=1}^{N} \boldsymbol{\nu}_{\mathrm{s},jj}^{\top}\Big(\mathrm{E}_{\mathsf{x}_j\sim\mathsf{q}_{\mathrm{v},j}}\{\boldsymbol{g}(\mathsf{x}_j)\} - \mathrm{E}_{\mathsf{x}_j\sim\mathsf{q}_{\mathrm{s},j}}\{\boldsymbol{g}(\mathsf{x}_j)\}\Big) \\ &+ \sum_{j=1}^{N} \Big(\nu_j\big(\int \mathsf{q}_{\mathrm{v},j}(x_j)\,\mathrm{d}x_j - 1\big) + \nu_{\mathrm{s},j}\big(\int \mathsf{q}_{\mathrm{s},j}(x_j)\,\mathrm{d}x_j - 1\big)\Big) \\ &+ \sum_{i=1}^{M} \nu_{\mathrm{c},i}\big(\int \mathsf{q}_{\mathrm{c},i}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} - 1\big)\ . \end{aligned} \tag{3.18}$$

The stationary points can be obtained by differentiating the Lagrangian with respect to its parameters and setting the term to zero. When differentiating w.r.t. one of the Lagrangian multipliers, this leads to the respective constraint, while differentiation w.r.t. the marginals specify the form of the respective marginal, because the stationarity constraint on the derivative fixates the form of the marginal in the optimal point. This form will help us to derive algorithms for the solution to the problem, since the stationary points are the targets to aim for. In the latter case, the differentiation is again a functional derivative, for more details see Appendix C.

The derivatives, that subsequently are set to zero, read

$$\frac{\partial}{\partial \mathsf{q}_{\mathrm{v},j}}\mathcal{L}(\mathsf{q}(\boldsymbol{x})) = -M\log \mathsf{q}_{\mathrm{v},j}(x_j) + \left(\boldsymbol{\nu}_{\mathrm{s},jj}^{\top} + \sum_{i=1}^{M}\boldsymbol{\nu}_{\mathrm{c},ij}^{\top}\right)\boldsymbol{g}(x_j) + \mathrm{const.} \stackrel{!}{=} 0\ , \tag{3.19}$$

$$\frac{\partial}{\partial \mathsf{q}_{\mathrm{c},i}}\mathcal{L}(\mathsf{q}(\boldsymbol{x})) = \log \mathsf{q}_{\mathrm{c},i}(\boldsymbol{x}) - \log \mathsf{f}_{\mathsf{y}_i|\mathsf{x}}(\boldsymbol{x}) - \sum_{j=1}^{N}\boldsymbol{\nu}_{\mathrm{c},ij}^{\top}\boldsymbol{g}(x_j) + \mathrm{const.} \stackrel{!}{=} 0\ , \tag{3.20}$$

$$\frac{\partial}{\partial \mathsf{q}_{\mathrm{s},j}}\mathcal{L}(\mathsf{q}(\boldsymbol{x})) = \log \mathsf{q}_{\mathrm{s},j}(x_j) - \log \mathsf{f}_{\mathsf{x}}(x_j) - \boldsymbol{\nu}_{\mathrm{s},jj}^{\top}\boldsymbol{g}(x_j) + \mathrm{const.} \stackrel{!}{=} 0\ , \tag{3.21}$$

where parts that do not depend on any element of $\boldsymbol{x}$ are summarized to a constant addend.

The resulting terms, solved for the marginals, read [HOW+05]

$$\mathsf{q}_{\mathrm{v},j}(x_j) = \frac{1}{Z_{\mathrm{v},j}}\exp\left(\frac{1}{M}(\boldsymbol{\nu}_{\mathrm{s},jj} + \sum_{i=1}^{M}\boldsymbol{\nu}_{\mathrm{c},ij})^{\top}\boldsymbol{g}(x_j)\right)\ , \tag{3.22}$$

$$\mathsf{q}_{\mathrm{c},i}(\boldsymbol{x}) = \frac{1}{Z_{\mathrm{c},i}}\mathsf{f}_{\mathsf{y}_i|\mathsf{x}}(\boldsymbol{x})\exp\left(\sum_{j=1}^{N}\boldsymbol{\nu}_{\mathrm{c},ij}^{\top}\boldsymbol{g}(x_j)\right)\ , \tag{3.23}$$

$$\mathsf{q}_{\mathrm{s},j}(x_j) = \frac{1}{Z_{\mathrm{s},j}}\mathsf{f}_{\mathsf{x}}(x_j)\exp\Big(\boldsymbol{\nu}_{\mathrm{s},jj}^{\top}\boldsymbol{g}(x_j)\Big)\ , \tag{3.24}$$

where $Z_{\mathrm{v},j}$, $Z_{\mathrm{c},i}$, and $Z_{\mathrm{s},j}$ are introduced to fulfill the normalization constraint.

As expected, the marginals at the stationary points are members of exponential families. Noteworthy, the Lagrangian multipliers turn out to work as natural parameters. We define

$$\boldsymbol{\theta}_j \stackrel{\mathrm{def}}{=} \frac{1}{M}(\boldsymbol{\nu}_{\mathrm{s},jj} + \sum_{i=1}^{M}\boldsymbol{\nu}_{\mathrm{c},ij})\ , \tag{3.25}$$

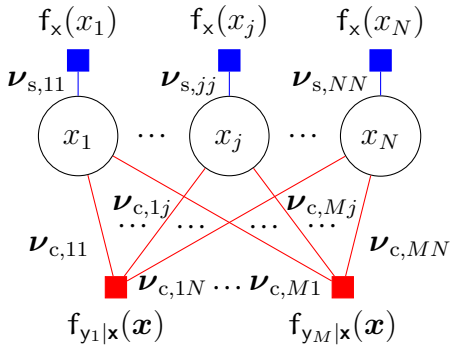


Figure 3.3: Factor graph for compressed sensing and edge-dependent multipliers.

in order to specify the natural parameter at the variable node. The computational connection of the variable node to the factor nodes, i.e., the marginals $\mathsf{q}_{\mathrm{c},i}(\boldsymbol{x})$, respectively $\mathsf{q}_{\mathrm{s},j}(x_j)$ is given by the Lagrangian multipliers $\boldsymbol{\nu}_{\mathrm{c},ij}$, respectively $\boldsymbol{\nu}_{\mathrm{s},jj}$ since the multipliers appear in these connected marginals as well. Hence, each multiplier belongs to one edge, i.e., they are *edge-dependent*, cf. Fig. 3.3. The variable node marginal $\mathsf{q}_{\mathrm{v},j}(x_j)$ is entirely specified by the respective Lagrangian multipliers and specifies a Gaussian distribution with the given sufficient statistics $\boldsymbol{g}(\boldsymbol{x})$. The factor marginals $\mathsf{q}_{\mathrm{c},i}(\boldsymbol{x})$ and $\mathsf{q}_{\mathrm{s},j}(x_j)$, however, include also the factors $\mathsf{f}_{\mathsf{y}_i|\mathsf{x}}(\boldsymbol{x})$ and $\mathsf{f}_{\mathsf{x}}(x_j)$, respectively.

3.3 Algorithms

The stationary points derived above can be seen as targets to aim for, because they solve the given problem. A recovery algorithm for CS should therefore end up with the forms for the marginals (3.22)–(3.24) above. In order to get there, the algorithm needs to make use of the structural connections incorporated in the Lagrangian multipliers. Since these multipliers are natural parameters of an exponential family, they express the entire knowledge about the corresponding distribution. As they are, furthermore, edge-dependent, they can be seen as "messages" to be broadcasted to other nodes. The connection between natural parameters and moments of exponential families, cf. Appendix D.2, thus, makes it possible to shift the processing from the distributions to the parameter domain, which comes in handy, as the interest lies in the conditional mean (2.9). In the following, it is important to find out, how the processing is done. At first, the moments that are computed at the factor nodes are considered.

3.3.1 Expectations

Since the above specified sufficient statistic represents a Gaussian distribution, the natural parameters of $\mathsf{q}_{\mathrm{s},j}(x_j)$ can be identified as $\boldsymbol{\nu}_{\mathrm{s},jj} = [\tilde{x}_{\mathrm{s},j}/\tilde{\sigma}^2_{\mathrm{s},j},\ 1/\tilde{\sigma}^2_{\mathrm{s},j}]^\top$ with $\tilde{x}_{\mathrm{s},j}$ and $\tilde{\sigma}^2_{\mathrm{s},j}$ being mean and variance of the Gaussian distribution. With this identification, it is obvious that the

moments to be computed relate to the non-linear estimators (2.19) and (2.20), i.e.,

$$\mathrm{E}_{\mathsf{x}_j\sim\mathsf{q}_{\mathrm{s},j}}\left\{g_{\lambda_j}(\mathsf{x}_j)\right\} = \mathrm{E}_{\mathsf{x}_j}\{\mathsf{x}_j \mid \tilde{x}_{\mathrm{s},j}\} = m_{\mathrm{s},j} \ , \tag{3.26}$$

$$\mathrm{E}_{\mathsf{x}_j\sim\mathsf{q}_{\mathrm{s},j}}\left\{g_{\Lambda_j}(\mathsf{x}_j)\right\} = -\frac{1}{2}\mathrm{E}_{\mathsf{x}_j}\left\{\mathsf{x}_j^2 \mid \tilde{x}_{\mathrm{s},j}\right\} = -\frac{1}{2}\left(\sigma_{\mathrm{s},j}^2 + m_{\mathrm{s},j}^2\right) \ . \tag{3.27}$$

For the channel-constrained factor, this is similar. First note that by stacking, it is possible to define a new natural parameter, such that

$$\sum_{j=1}^{N} \boldsymbol{\nu}_{\mathrm{c},ij}^{\top}\boldsymbol{g}(x_j) = \boldsymbol{\nu}_{\mathrm{c},i}^{\top}\boldsymbol{g}(\boldsymbol{x}) \ . \tag{3.28}$$

Using the identification

$$\boldsymbol{\nu}_{\mathrm{c},i} = [m_{\mathrm{x},1}/\sigma_{\mathrm{x},1}^2, \ \ldots, m_{\mathrm{x},N}/\sigma_{\mathrm{x},N}^2, \ 1/\sigma_{\mathrm{x},1}^2, \ \ldots, \ 1/\sigma_{\mathrm{x},N}^2]^{\top} \ , \tag{3.29}$$

the moment computations (for the entire vector) relate to (2.13) and (2.14) by

$$\mathrm{E}_{\mathbf{x}\sim\mathsf{q}_{\mathrm{c},i}}\{\boldsymbol{g}_{\boldsymbol{\lambda}}(\mathbf{x})\} = \mathrm{E}_{\mathbf{x}}\{\mathbf{x} \mid y_i\} = \boldsymbol{m}_{\mathrm{c},i} \ , \tag{3.30}$$

$$\mathrm{E}_{\mathbf{x}\sim\mathsf{q}_{\mathrm{c},i}}\{\boldsymbol{g}_{\boldsymbol{\Lambda}}(\mathbf{x})\} = -\frac{1}{2}\mathbf{diag}\Big(\mathrm{E}_{\mathbf{x}}\Big\{\mathbf{x}\mathbf{x}^{\top} \mid y_i\Big\}\Big) = -\frac{1}{2}\mathbf{diag}\Big(\boldsymbol{\Phi}_{\mathrm{c},i} + \boldsymbol{m}_{\mathrm{c},i}\boldsymbol{m}_{\mathrm{c},i}^{\top}\Big) \ . \tag{3.31}$$

The operator $\mathbf{diag}(\cdot)$ shall denote the vector of diagonal entries of the matrix, which is input to the operator. Otherwise, if a single element with an index, usually j is inserted, it shall denote the diagonal matrix with diagonal entries specified by the indexed input variable, e.g., if $j \in \{1, \ \ldots, \ N\}$ then the resulting matrix is a $N \times N$ matrix.

The estimates for a single variable, thus, relate to the respective jth components in (2.16) and (2.17) via

$$\mathrm{E}_{\mathsf{x}_j\sim\mathsf{q}_{\mathrm{c},i}}\left\{g_{\lambda_j}(\mathsf{x}_j)\right\} = m_{\mathrm{c},i,j} \ , \tag{3.32}$$

$$\mathrm{E}_{\mathsf{x}_j\sim\mathsf{q}_{\mathrm{c},i}}\left\{g_{\Lambda_j}(\mathsf{x}_j)\right\} = -\frac{1}{2}\left(\sigma_{\mathrm{c},i,j}^2 + m_{\mathrm{c},i,j}^2\right) \ . \tag{3.33}$$

As intended, all estimations are one-dimensional for the given consideration.

3.3.2 Projection to Exponential Family

Having the moments computed at the factor nodes, it is necessary to find out, how to process them for the exchange between the nodes, i.e., we have to get back to the edges. Since the Lagrangian multipliers $\boldsymbol{\nu}_{\mathrm{c},ij}$ and $\boldsymbol{\nu}_{\mathrm{s},jj}$ specify the connections in the factor graph, they can be interpreted as "messages" on an edge. To that end, the two possible sending directions on the edge have to be distinguished. Consider the variable node with $\mathsf{q}_{\mathrm{v},j}(x_j)$ in (3.22). Here, as already mentioned, the distribution is entirely specified by the respective Lagrangian multipliers, i.e., they tell something about the distribution and can therefore be interpreted as *outgoing* messages. For the distinction to the outgoing messages, the *incoming* messages are denoted by $\tilde{\boldsymbol{\nu}}_{\mathrm{c},ij}$ and $\tilde{\boldsymbol{\nu}}_{\mathrm{s},jj}$, the denomination is exemplary shown for one node in Fig. 3.4.

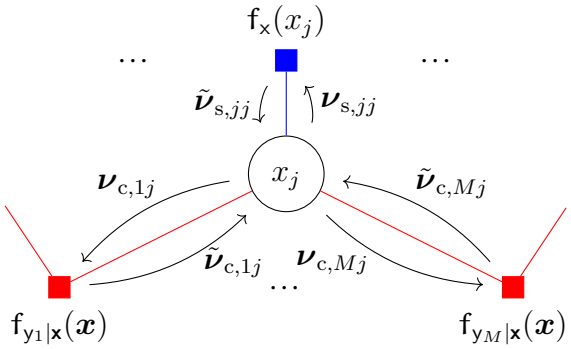


Figure 3.4: Detailed view on the factor graph for CS with edge- and direction-dependent messages.

In order to close the gap between the moments on the nodes and the natural parameters that serve as messages on the edges, we have a look at the moment-matching constraints (3.14) and (3.15). These were enforced between factor nodes and connected variable nodes. Since the marginals at the variable nodes represent members of an exponential family, i.e., a Gaussian distribution for the sufficient statistics $\boldsymbol{g}(\boldsymbol{x})$ as specified above, the moments can be mapped according to Ex. D.2 from Appendix D to the natural parameter of the marginal at the variable node to obtain for a channel-constrained factor node ($j \in \{1, \ldots, N\}$)

$$\boldsymbol{\theta}_{\mathrm{c},i,j} = [m_{\mathrm{c},i,j}/\sigma^2_{\mathrm{c},i,j},\, 1/\sigma^2_{\mathrm{c},i,j}]^\top \,, \quad i \in \{1, \ldots, M\} \,, \tag{3.34}$$

respectively,

$$\boldsymbol{\theta}_{\mathrm{s},j} = [m_{\mathrm{s},j}/\sigma^2_{\mathrm{s},j},\, 1/\sigma^2_{\mathrm{s},j}]^\top \,, \tag{3.35}$$

for a signal-constrained factor node.

3.3.3 Processing for Stationarity

The mapping in (3.34) and (3.35) above shows a way how to represent the signal component x_j after the corresponding processing. However, it does not state what messages have to be sent on the respective edges. The edges are currently described by messages that pass it in the two possible directions. For example in Fig. 3.4 the edge between variable x_j and (signal-constrained) factor $\mathsf{f}_{\mathsf{x}}(x_j)$ transports the messages $\tilde{\boldsymbol{\nu}}_{\mathrm{s},jj}$ and $\boldsymbol{\nu}_{\mathrm{s},jj}$. Since this edge is connected to the signal component x_j, it should provide knowledge about this variable. As already mentioned, this knowledge is (after processing at the factor node) given by $\boldsymbol{\theta}_{\mathrm{s},j}$. In the following, the connection between these three parameters is sought.

To that end, consider the following partitioning of the posterior

$$\mathsf{f}_{\mathsf{x}|\mathsf{y}}(\boldsymbol{x}) = \text{const.} \cdot \mathsf{f}_{\mathsf{x}}(x_j) \cdot \left(\prod_{\substack{j'=1 \\ j' \neq j}}^{N} \mathsf{f}_{\mathsf{x}}(x_{j'}) \prod_{i=1}^{M} \mathsf{f}_{\mathsf{y}_i|\mathsf{x}}(\boldsymbol{x}) \right) \,, \tag{3.36}$$

i.e., it is split into the factor $\mathsf{f}_{\mathsf{x}}(x_j)$ on the one side and everything else on the other side. This extracts the edge under consideration as the combining element between the two parts. It also

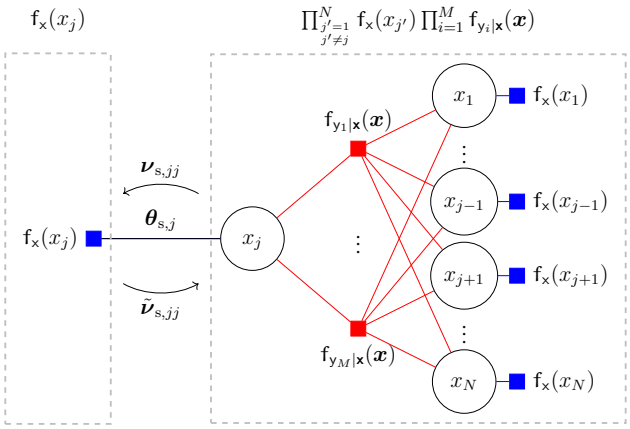


Figure 3.5: Depiction of the partitioning of the factor graph with the messages between the parts and the representation of the knowledge $\boldsymbol{\theta}_{\mathrm{s},j}$ on the connecting edge.

shows that a message from $\mathsf{f}_{\mathsf{x}}(x_j)$ must represent knowledge from the respective factor, while a message towards $\mathsf{f}_{\mathsf{x}}(x_j)$ must carry the knowledge from everywhere else. Obviously, the multiplication of both parts yields (up to a constant factor) the posterior again. The splitting is displayed in Fig. 3.5 with $\mathsf{f}_{\mathsf{x}}(x_j)$ on the left, the other part on the right, and the corresponding representation of the knowledge $\boldsymbol{\theta}_{\mathrm{s},j}$, as well as the messages in between.

In this consideration, the interest lies in the variable x_j only, which means that mathematically

$$\mathsf{f}_{\mathsf{x}_j|\mathsf{y}}(x_j) \overset{\text{def}}{=} \int \cdots \int \mathsf{f}_{\mathsf{x}|\mathsf{y}}(\boldsymbol{x}) \, \mathrm{d}x_1 \ldots \mathrm{d}x_{j-1} \mathrm{d}x_{j+1} \ldots \mathrm{d}x_N \tag{3.37}$$

should be approximated. Since $\mathsf{f}_{\mathsf{x}}(x_j)$ does not carry an argument that is used in the integral, it is

$$\mathsf{f}_{\mathsf{x}_j|\mathsf{y}}(x_j) = \text{const.} \cdot \mathsf{f}_{\mathsf{x}}(x_j) \cdot \int \cdots \int \prod_{\substack{j'=1\\j'\neq j}}^{N} \mathsf{f}_{\mathsf{x}}(x_{j'}) \prod_{i=1}^{M} \mathsf{f}_{\mathsf{y}_i|\mathsf{x}}(\boldsymbol{x}) \, \mathrm{d}x_1 \ldots \mathrm{d}x_{j-1} \mathrm{d}x_{j+1} \ldots \mathrm{d}x_N \ , \tag{3.38}$$

i.e., the same separation as before can be obtained. This can now be set into relation with the messages. The messages used here are natural parameters specifying a member of an exponential family. The distributions corresponding to the messages therefore express approximations for the two separated parts. Following the thought that the message $\tilde{\boldsymbol{\nu}}_{\mathrm{s},jj}$ must represent the factor $\mathsf{f}_{\mathsf{x}}(x_j)$ and $\boldsymbol{\nu}_{\mathrm{s},jj}$ the other part leads to an entire approximation, which reads

$$\mathsf{f}_{\mathsf{x}_j|\mathsf{y}}(x_j) \approx \text{const.} \cdot \exp\left(\tilde{\boldsymbol{\nu}}_{\mathrm{s},jj}^{\top} \boldsymbol{g}(x_j)\right) \cdot \exp\left(\boldsymbol{\nu}_{\mathrm{s},jj}^{\top} \boldsymbol{g}(x_j)\right) \ . \tag{3.39}$$

The knowledge about x_j that is associated to this edge is, as already mentioned, given by $\boldsymbol{\theta}_{\mathrm{s},j}$, i.e.,

$$\mathsf{f}_{\mathsf{x}_j|\mathsf{y}}(x_j) \approx \text{const.} \cdot \exp\left(\boldsymbol{\theta}_{\mathrm{s},j}^{\top} \boldsymbol{g}(x_j)\right) \ . \tag{3.40}$$

Hence, the approximation of $\mathsf{f}_{\mathsf{x}_j|\mathsf{y}}(x_j)$, which should fit to the respective incoming and outgoing messages, is given by the exponential family carrying $\boldsymbol{\theta}_{\mathrm{s},j}$ as natural parameter. The

multiplication of the members of this exponential family transforms to an addition of the natural parameters, meaning that the natural parameter $\boldsymbol{\theta}_{\mathrm{s},j}$ representing the entire knowledge is given by the summation of the natural parameters describing the messages, i.e., [LDH+07, Eqs. (54), (55)]

$$\boldsymbol{\theta}_{\mathrm{s},j} = \tilde{\boldsymbol{\nu}}_{\mathrm{s},jj} + \boldsymbol{\nu}_{\mathrm{s},jj} \ , \quad j \in \{1, \ldots, N\} \ . \tag{3.41}$$

Note that in [LDH+07] this connection is derived from the *sum-product rule* [KFL01] for Gaussian distributions, which is one of the instances of a message passing algorithm.

Neglecting the loops in the graph, this holds similarly for the channel-constrained factors $\mathrm{f}_{\mathsf{y}_i|\mathsf{x}}(\boldsymbol{x})$, i.e.,

$$\boldsymbol{\theta}_{\mathrm{c},i,j} = \tilde{\boldsymbol{\nu}}_{\mathrm{c},ij} + \boldsymbol{\nu}_{\mathrm{c},ij} \ , \quad j \in \{1, \ldots, N\} \ , \quad i \in \{1, \ldots, M\} \ . \tag{3.42}$$

Having these connections, it is now possible to turn to the necessary processing in the algorithm. To that end, recall that the stationary points of the optimization problem are already given in (3.22)–(3.24). These are the goals an iterative algorithm should achieve. In order to derive the processing for the algorithms, it is therefore suitable to stick to updates that fulfill these equations, i.e., stay in a stationary point once reached. In particular (3.25), i.e.,

$$\boldsymbol{\theta}_j = \frac{1}{M} \left(\boldsymbol{\nu}_{\mathrm{s},jj} + \sum_{i=1}^{M} \boldsymbol{\nu}_{\mathrm{c},ij} \right) \ , \quad j \in \{1, \ldots, N\} \ , \tag{3.43}$$

is of importance for that matter as it encodes the structure of the factor graph in itself. In a stationary point, the representation of x_j given in $\boldsymbol{\theta}_j$ from (3.25) should not change with the processing, which means that eventually

$$\boldsymbol{\theta}_j \stackrel{!}{=} \boldsymbol{\theta}_{\mathrm{s},j} \stackrel{!}{=} \boldsymbol{\theta}_{\mathrm{c},1,j} \stackrel{!}{=} \cdots \stackrel{!}{=} \boldsymbol{\theta}_{\mathrm{c},M,j} \ , \quad j \in \{1, \ldots, N\} \ , \tag{3.44}$$

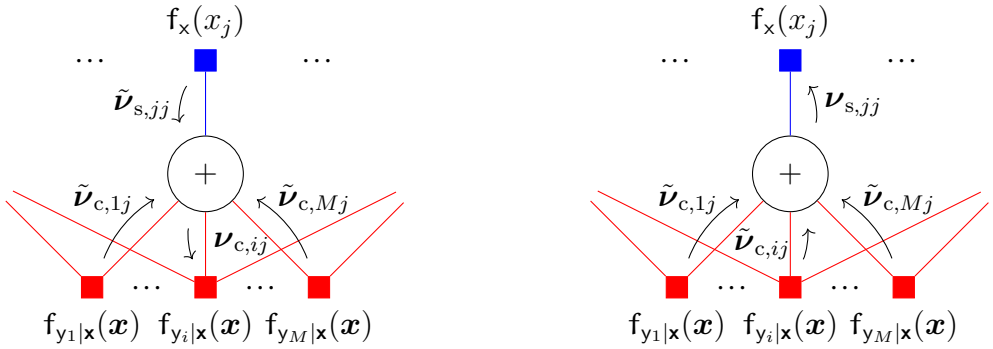
must hold. With this equality, (3.41) and (3.42) can be inserted into the stationary condition (3.25), which yields

$$\begin{aligned} \boldsymbol{\theta}_j &\stackrel{(3.25)}{=} \frac{1}{M} \left(\boldsymbol{\nu}_{\mathrm{s},jj} + \sum_{i=1}^{M} \boldsymbol{\nu}_{\mathrm{c},ij} \right) \\ &= \frac{1}{M} \left(\boldsymbol{\theta}_j - \tilde{\boldsymbol{\nu}}_{\mathrm{s},jj} + \sum_{i=1}^{M} (\boldsymbol{\theta}_j - \tilde{\boldsymbol{\nu}}_{\mathrm{c},ij}) \right) \\ &= \frac{M+1}{M} \boldsymbol{\theta}_j - \frac{1}{M} \left(\tilde{\boldsymbol{\nu}}_{\mathrm{s},jj} + \sum_{i=1}^{M} \tilde{\boldsymbol{\nu}}_{\mathrm{c},ij} \right) \end{aligned} \tag{3.45}$$

$$\Leftrightarrow \quad \left(\frac{M+1}{M} - 1 \right) \boldsymbol{\theta}_j = \frac{1}{M} \boldsymbol{\theta}_j = \frac{1}{M} \left(\tilde{\boldsymbol{\nu}}_{\mathrm{s},jj} + \sum_{i=1}^{M} \tilde{\boldsymbol{\nu}}_{\mathrm{c},ij} \right) \ , \tag{3.46}$$

i.e.,

$$\boldsymbol{\theta}_j = \tilde{\boldsymbol{\nu}}_{\mathrm{s},jj} + \sum_{i=1}^{M} \tilde{\boldsymbol{\nu}}_{\mathrm{c},ij} \ . \tag{3.47}$$



(a) Visualization fo the computation of the message to channel factor $\mathsf{f}_{\mathsf{y}_i|\mathsf{x}}(\boldsymbol{x})$.

(b) Visualization of the computation of the message to signal factor $\mathsf{f}_{\mathsf{x}}(x_j)$.

Figure 3.6: Visualizations for the connections between incoming and outgoing messages, i.e., Eqs. (3.49) on the left and (3.48) on the right.

Comparing again with (3.41) and (3.42), this means that the connection between messages leaving the variable node and adjacent incoming messages must be given by

$$\boldsymbol{\nu}_{\mathrm{s},jj} = \sum_{i=1}^{M} \tilde{\boldsymbol{\nu}}_{\mathrm{c},ij} \ , \tag{3.48}$$

respectively

$$\boldsymbol{\nu}_{\mathrm{c},ij} = \tilde{\boldsymbol{\nu}}_{\mathrm{s},jj} + \sum_{\substack{i'=1 \\ i' \neq i}}^{M} \tilde{\boldsymbol{\nu}}_{\mathrm{c},i'j} \ . \tag{3.49}$$

The processing is depicted in Fig. 3.6.

The character of the separation in (3.41) and (3.42) is known as *extrinsic* [BG96], i.e., the a-priori knowledge (from before) is distinguished from the knowledge obtained by the processing. From (3.48) and (3.49) it becomes clear that the procedure is characterized by excluding knowledge of the incoming edge on which the next message should be sent. From an algorithmic point of view, especially for graphs with loops, broadcasting extrinsic is crucial to the success of the algorithm, because sending something on an edge that has been received there before, would lead to feedback loops and overemphasizing of certain aspects, which is precisely what was pursued to prevent by the approach based on the maximum entropy method, which was motivated in Sec. 2.2.2. This concept has been proven worthwhile not only in the context of the message passing algorithm, but also, e.g., in Turbo decoding [HOP96] and detection [GH11].

On the one hand, the connections in (3.41) and (3.42) reveal a simple relationship between outgoing and incoming messages as well as the moments computed at the nodes, i.e., the input has to be subtracted from the result $\boldsymbol{\theta}_j$ to obtain the returning message on the respective edge. The Equations (3.48) and (3.49), on the other hand, indicate that when computing the message for a certain edge, it is necessary to leave out the incoming message of that very

edge. Hence, there are two ways of processing extrinsic. In the following, the relation is examined in detail for a channel-constrained node $\mathrm{f}_{\mathrm{y}_i|\mathsf{x}}(\boldsymbol{x})$, where the input parameter is given by $\boldsymbol{\nu}_{\mathrm{c},ij} = [m_{\mathrm{x},j}/\sigma_{\mathrm{x},j}^2,\, 1/\sigma_{\mathrm{x},j}^2]^\top$ as above, Eq. (3.34) is used for the mapping between moments and natural parameters and $\tilde{\boldsymbol{\nu}}_{\mathrm{c},ij} \overset{\text{def}}{=} [m_{\mathrm{c},i,j}^{\setminus j}/\sigma_{\mathrm{c},i,j}^{2,\setminus j},\, 1/\sigma_{\mathrm{c},i,j}^{2,\setminus j}]^\top$ is defined for $i \in \{1, \ldots, M\}$ and $j \in \{1, \ldots, N\}$. Then Eq. (3.42) can be rewritten as

$$\tilde{\boldsymbol{\nu}}_{\mathrm{c},ij} = \boldsymbol{\theta}_{\mathrm{c},i,j} - \boldsymbol{\nu}_{\mathrm{c},ij} \tag{3.50}$$

so that the elements compute to

$$\begin{aligned}
\frac{m_{\mathrm{c},i,j}^{\setminus j}}{\sigma_{\mathrm{c},i,j}^{2,\setminus j}} &= \frac{m_{\mathrm{c},i,j}}{\sigma_{\mathrm{c},i,j}^2} - \frac{m_{\mathrm{x},j}}{\sigma_{\mathrm{x},j}^2} \\
&\overset{(2.16)}{=} \frac{m_{\mathrm{x},j}\sigma_{\mathsf{n}}^2 + m_{\mathrm{x},j}\sum_{\substack{j'=1\\j'\neq j}}^{N} \sigma_{\mathrm{x},j'}^2 a_{ij'}^2 + \sigma_{\mathrm{x},j}^2 a_{ij}\left(y_i - \sum_{\substack{j'=1\\j'\neq j}}^{N} a_{ij'} m_{\mathrm{x},j'}\right)}{\sigma_{\mathrm{x},j}^2\left(\sigma_{\mathsf{n}}^2 + \sum_{\substack{j'=1\\j'\neq j}}^{N} \sigma_{\mathrm{x},j'}^2 a_{ij'}^2\right)} - \frac{m_{\mathrm{x},j}}{\sigma_{\mathrm{x},j}^2} \\
&= \frac{a_{ij}\left(y_i - \sum_{\substack{j'=1\\j'\neq j}}^{N} a_{ij'} m_{\mathrm{x},j'}\right)}{\sigma_{\mathsf{n}}^2 + \sum_{\substack{j'=1\\j'\neq j}}^{N} \sigma_{\mathrm{x},j'}^2 a_{ij'}^2}
\end{aligned} \tag{3.51}$$

and

$$\begin{aligned}
\frac{1}{\sigma_{\mathrm{c},i,j}^{2,\setminus j}} &= \frac{1}{\sigma_{\mathrm{c},i,j}^2} - \frac{1}{\sigma_{\mathrm{x},j}^2} \overset{(2.17)}{=} \frac{\sigma_{\mathsf{n}}^2 + \sum_{j=1}^{N} \sigma_{\mathrm{x},j}^2 a_{ij}^2}{\sigma_{\mathrm{x},j}^2\left(\sigma_{\mathsf{n}}^2 + \sum_{\substack{j'=1\\j'\neq j}}^{N} \sigma_{\mathrm{x},j'}^2 a_{ij'}^2\right)} - \frac{1}{\sigma_{\mathrm{x},j}^2} \\
&= \frac{\sigma_{\mathsf{n}}^2 + \sum_{j=1}^{N} \sigma_{\mathrm{x},j}^2 a_{ij}^2 - \sigma_{\mathsf{n}}^2 - \sum_{\substack{j'=1\\j'\neq j}}^{N} \sigma_{\mathrm{x},j'}^2 a_{ij'}^2}{\sigma_{\mathrm{x},j}^2\left(\sigma_{\mathsf{n}}^2 + \sum_{\substack{j'=1\\j'\neq j}}^{N} \sigma_{\mathrm{x},j'}^2 a_{ij'}^2\right)} \\
&= \frac{a_{ij}^2}{\sigma_{\mathsf{n}}^2 + \sum_{\substack{j'=1\\j'\neq j}}^{N} \sigma_{\mathrm{x},j'}^2 a_{ij'}^2} \,.
\end{aligned} \tag{3.52}$$

This reveals that in the calculation for $\tilde{\boldsymbol{\nu}}_{\mathrm{c},ij}$, one can directly leave out $\boldsymbol{\nu}_{\mathrm{c},ij} = [m_{\mathrm{x},j}/\sigma_{\mathrm{x},j}^2,\, 1/\sigma_{\mathrm{x},j}^2]^\top$ as input parameter, i.e., the calculation of the extrinsic can be done as *post-processing* via the subtraction imposed by (3.42) or as *pre-processing* by omitting the respective input parameter. The respective computations are contrasted in the upper row of Fig. 3.8 as block diagrams, where the notation of the GMP algorithm (stated below) is used. On the left (Fig. 3.8(a)), the pre-processed extrinsic is shown, whereas on the right (Fig. 3.8(b)) the extrinsic is post-processed. Both philosophies, pre- and post-processed extrinsic, are used in the literature. For example, estimation theoretic extrinsic [GH11] in the context of Gaussian noise uses the subtractions as specified above. The pre-processing is used for example in the decoding procedure for low-density parity-check (LDPC) codes [Gal62]. In the message passing context, usually the pre-processing, i.e., incident parameter omitting strategy is considered [KMS+12, Pea88].

In case of a signal factor $\mathsf{f}_{\mathsf{x}}(x_j)$, the relation (3.48) indicates to compute the estimates (2.19) and (2.20) based on all messages coming from the channel-constrained nodes. Via the projection (3.35) and connection (3.42), an outgoing message $\boldsymbol{\nu}_{\mathrm{c},ij}$ is computed. Since the estimators (2.19) and (2.20) are usually non-linear, it is not possible to move the effect of the subtraction to the input of the estimator. Thus, the equivalence of post- and pre-processed extrinsic only holds if the factor marginal is Gaussian. This is usually not the case for the signal factors in CS. Hence, both procedures yield different results. The different processings are depicted in the lower row of Fig. 3.8; pre-processed extrinsic can be found on the left side, post-processed extrinsic on the right side.

3.3.4 Message Passing

Following the notion of pre-processing the extrinsic, i.e., omitting the incoming message of the edge for which the message is computed yields the *Gaussian belief propagation* (GBP) or *Gaussian message passing* (GMP) as stated in [KMS+12, Sec. 3.1]. In order to get there, recall that an outgoing message $\boldsymbol{\nu}_{\mathrm{c},ij}$ carries the information of the signal-constrained factor stimulated by all incoming messages $\tilde{\boldsymbol{\nu}}_{\mathrm{c},i'j}$ with $i' \in \{1, \ldots, M\} \setminus \{i\}$, i.e., all except the message of the considered edge. This means that the input parameters to the corresponding estimates (2.19) and (2.20) are computed via

$$\frac{\tilde{x}_{\mathrm{s},j}^{\setminus i}}{\tilde{\sigma}_{\mathrm{s},j}^{2,\setminus i}} = \sum_{\substack{i'=1 \\ i' \neq i}}^{M} \frac{m_{\mathrm{c},i',j}^{\setminus j}}{\sigma_{\mathrm{c},i',j}^{2,\setminus j}} \,, \qquad \text{and} \qquad \frac{1}{\tilde{\sigma}_{\mathrm{s},j}^{2,\setminus i}} = \sum_{\substack{i'=1 \\ i' \neq i}}^{M} \frac{1}{\sigma_{\mathrm{c},i',j}^{2,\setminus j}} \,. \tag{3.53}$$

With these inputs, one can define

$$m_{\mathrm{s},j}^{\setminus i} \overset{\text{def}}{=} \mathrm{E}_{\mathsf{x}_j}\left\{\mathsf{x}_j \mid \tilde{x}_{\mathrm{s},j}^{\setminus i}\right\} = \frac{\int x_j \mathsf{f}_{\mathsf{x}}(x_j) \exp\left(-\frac{(\tilde{x}_{\mathrm{s},j}^{\setminus i} - x_j)^2}{2\tilde{\sigma}_{\mathrm{s},j}^{2,\setminus i}}\right) \mathrm{d}x_j}{\int \mathsf{f}_{\mathsf{x}}(x_j) \exp\left(-\frac{(\tilde{x}_{\mathrm{s},j}^{\setminus i} - x_j)^2}{2\tilde{\sigma}_{\mathrm{s},j}^{2,\setminus i}}\right) \mathrm{d}x_j} \tag{3.54}$$

and

$$\sigma_{\mathrm{s},j}^{2,\setminus i} \overset{\text{def}}{=} \mathrm{E}_{\mathsf{x}_j}\left\{(\mathsf{x}_j - m_{\mathrm{s},j}^{\setminus i})^2 \mid \tilde{x}_{\mathrm{s},j}^{\setminus i}\right\} = \frac{\int (x_j - m_{\mathrm{s},j})^2 \mathsf{f}_{\mathsf{x}}(x_j) \exp\left(-\frac{(\tilde{x}_{\mathrm{s},j}^{\setminus i} - x_j)^2}{2\tilde{\sigma}_{\mathrm{s},j}^{2,\setminus i}}\right) \mathrm{d}x_j}{\int \mathsf{f}_{\mathsf{x}}(x_j) \exp\left(-\frac{(\tilde{x}_{\mathrm{s},j}^{\setminus i} - x_j)^2}{2\tilde{\sigma}_{\mathrm{s},j}^{2,\setminus i}}\right) \mathrm{d}x_j} \,. \tag{3.55}$$

The resulting estimates, mapped to the natural parameters, serve as input to the computation of the channel-constrained estimate. Therefore, defining $\boldsymbol{\nu}_{\mathrm{c},ij} = [m_{\mathrm{s},j}^{\setminus i}/\sigma_{\mathrm{s},j}^{2,\setminus i},\, 1/\sigma_{\mathrm{s},j}^{2,\setminus i}]^{\top}$, the computations at the channel-constrained nodes become

$$\tilde{\boldsymbol{\nu}}_{\mathrm{c},ij} = \left[\frac{m_{\mathrm{c},i,j}^{\setminus j}}{\sigma_{\mathrm{c},i,j}^{2,\setminus j}},\, \frac{1}{\sigma_{\mathrm{c},i',j}^{2,\setminus j}}\right]^{\top} = \left[\frac{a_{ij}\left(y_i - \sum_{\substack{j'=1 \\ j' \neq j}}^{N} a_{ij'} m_{\mathrm{s},j'}^{\setminus i}\right)}{\sigma_{\mathsf{n}}^2 + \sum_{\substack{j'=1 \\ j' \neq j}}^{N} \sigma_{\mathrm{s},j'}^{2,\setminus i} a_{ij'}^2},\, \frac{a_{ij}^2}{\sigma_{\mathsf{n}}^2 + \sum_{\substack{j'=1 \\ j' \neq j}}^{N} \sigma_{\mathrm{s},j'}^{2,\setminus i} a_{ij'}^2}\right]^{\top} \,. \tag{3.56}$$

The resulting algorithm iterates between the nodes by choosing the indices $j \in \{1, \ldots, N\}$ and $i \in \{1, \ldots, M\}$ in a suitable manner. The entire procedure is stated in Algorithm 3.1 below and in Algorithm E.1 in Appendix E. The notation uses only means and variances for comprehensibility, however, for the implementation it is beneficial to resort to scaled means and inverse variances (precisions), so that, e.g., the summations in Lines 7 and 8 spare the inverses, and scalings.

Algorithm 3.1: $\boldsymbol{m}_{\mathrm{s}} = \mathtt{GMP}(\boldsymbol{y},\, \boldsymbol{A},\, \sigma_{\mathsf{n}}^2)$

1 $m_{\mathrm{s},j}^{\backslash i} = 0$, $\sigma_{\mathrm{s},j}^{2,\backslash i} = s/N \;\forall\, j \in \{1, \ldots, N\}, i \in \{1, \ldots, M\}$ `// initialization`
2 $m_{\mathrm{c},i,j}^{\backslash j} = 0$, $\sigma_{\mathrm{c},i,j}^{2,\backslash j} = s/N \;\forall\, j \in \{1, \ldots, N\}, i \in \{1, \ldots, M\}$
3 **while** *stopping criterion not met* **do**
4 Choose $j \in \{1, \ldots, N\}$ and $i \in \{1, \ldots, M\}$
5 $m_{\mathrm{c},i,j}^{\backslash j} = (y_i - \sum_{\substack{j'=1 \\ j' \neq j}}^{N} a_{ij'} m_{\mathrm{s},j'}^{\backslash i})/a_{ij}$ `// "upward" messages`
6 $\sigma_{\mathrm{c},i,j}^{2,\backslash j} = (\sigma_{\mathsf{n}}^2 + \sum_{\substack{j'=1 \\ j' \neq j}}^{N} \sigma_{\mathrm{s},j'}^{2,\backslash i} a_{ij'}^2)/a_{ij}^2$
7 $\tilde{\sigma}_{\mathrm{s},j}^{2,\backslash i} = \left(\sum_{\substack{i'=1 \\ i' \neq i}}^{M} 1/\sigma_{\mathrm{c},i',j}^{2,\backslash j} \right)^{-1}$ `// input to NLMMSE (edge-dependent)`
8 $\tilde{x}_{\mathrm{s},j}^{\backslash i} = \tilde{\sigma}_{\mathrm{s},j}^{2,\backslash i} \sum_{\substack{i'=1 \\ i' \neq i}}^{M} m_{\mathrm{c},i',j}^{\backslash j} / \sigma_{\mathrm{c},i',j}^{2,\backslash j}$
9 $m_{\mathrm{s},j}^{\backslash i} = \mathrm{E}_{\mathsf{x}_j}\{\mathsf{x}_j \mid \tilde{x}_{\mathrm{s},j}^{\backslash i},\, \tilde{\sigma}_{\mathrm{s},j}^{2,\backslash i}\}$ `// "downward" messages`
10 $\sigma_{\mathrm{s},j}^{2,\backslash i} = \mathrm{E}_{\mathsf{x}_j}\{(\mathsf{x}_j - m_{\mathrm{s},j}^{\backslash i})^2 \mid \tilde{x}_{\mathrm{s},j}^{\backslash i},\, \tilde{\sigma}_{\mathrm{s},j}^{2,\backslash i}\}$
11 $\tilde{\sigma}_{\mathrm{s},j}^2 = \left(\sum_{i=1}^{M} 1/\sigma_{\mathrm{c},i',j}^{2,\backslash j} \right)^{-1} \quad \forall\, j \in \{1, \ldots, N\}$
12 $\tilde{x}_{\mathrm{s},j} = \tilde{\sigma}_{\mathrm{s},j}^2 \sum_{i=1}^{M} m_{\mathrm{c},i',j}^{\backslash j} / \sigma_{\mathrm{c},i',j}^{2,\backslash j} \quad \forall\, j \in \{1, \ldots, N\}$
13 $\boldsymbol{m}_{\mathrm{s}} = \mathrm{E}_{\mathbf{x}}\{\mathbf{x} \mid \tilde{\boldsymbol{x}}_{\mathrm{s}},\, \tilde{\sigma}_{\mathrm{s},1}^2,\, \ldots,\, \tilde{\sigma}_{\mathrm{s},N}^2\}$ `// NLMMSE`

The statement of GMP in Algorithm 3.1 implies that the messages are sent from one factor node $\mathsf{f}_{\mathsf{y}_i|\mathbf{x}}(\boldsymbol{x})$ to one variable node x_j ("upwards") and then the messages from variable node x_j (by employing factor node $\mathsf{f}_{\mathsf{x}}(x_j)$) to factor node $\mathsf{f}_{\mathsf{y}_i|\mathbf{x}}(\boldsymbol{x})$ ("downwards"); subsequently choosing the next indices to be processed. This notation is used to simplify readability; for the simulations in Chapter 6, a schedule that iterates between computing all upward messages, followed by computing all downward messages, is used. Of course also other schedules are possible.

The "upward" and "downward" messages as used in the statement of the algorithm are also visualized in Fig. 3.7. The input message for the signal factor $\mathsf{f}_{\mathsf{x}}(x_j)$ is not shown, because it depends for which outgoing edge (downwards) the estimate is computed; here it may be either $\tilde{x}_{\mathrm{s},j}^{\backslash 1}$, together with $\tilde{\sigma}_{\mathrm{s},j}^{2,\backslash 1}$, or $\tilde{x}_{\mathrm{s},j}^{\backslash M}$ and $\tilde{\sigma}_{\mathrm{s},j}^{2,\backslash M}$.

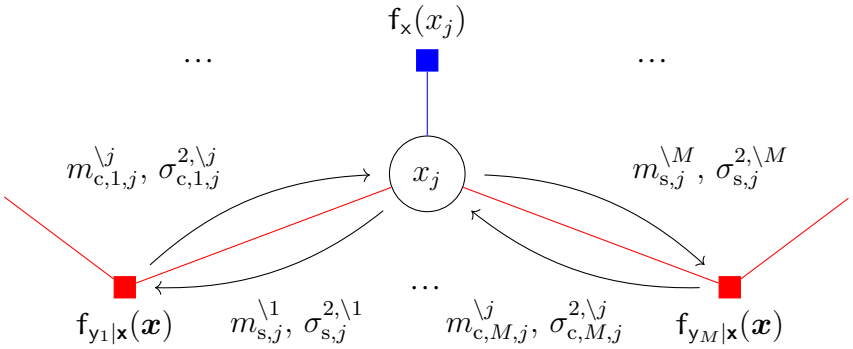


Figure 3.7: Visualization of "upward" and "downward" messages in terms of means and variances in the factor graph for CS.

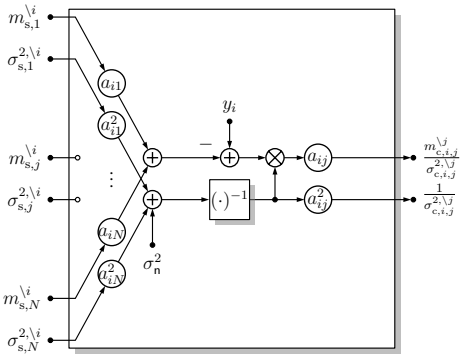
3.3.5 Related Algorithms

Despite simplifying to two scalar moments instead of computing arbitrary marginals or multi-dimensional estimates, the resulting algorithm still inherits high computational complexity. The reason lies in the fine-granular view on the factor graph that causes edge-dependent estimates (and messages) that have to be computed, stored and propagated and the fact that the variable nodes are connected to all channel-constrained factors, i.e., the sensing matrix $\boldsymbol{A}$ is not sparse. The edge-dependent processing therefore impedes scalability, i.e., the algorithm can only be used for tasks with small dimensions.

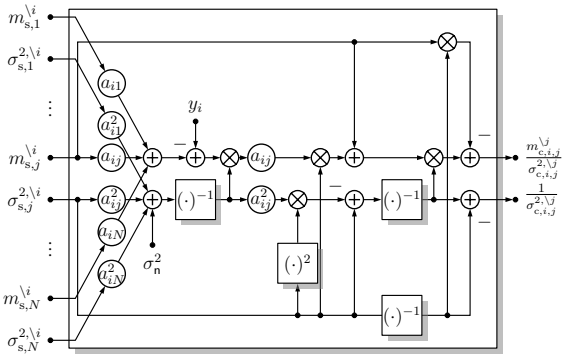
One way to reduce the computation overhead is to utilize post-processed extrinsic, instead of pre-processed extrinsic as in Algorithm 3.1. In Eqs. (3.51) and (3.52), it has been shown that for the channel-constrained estimation both processings are mathematically the same. The post-processed extrinsic has in combination with the schedule that computes all "upward" messages together the advantage that the expectations get node-dependent, i.e., have to be performed M times, instead of $M \cdot N$ times, when they are edge-dependent as it is the case when the extrinsic is pre-processed. Of course, afterwards the computation of the extrinsic is again edge-dependent, but this involves only scalings and one subtraction (per edge and parameter), as opposed to the multiple summations of (almost) all parameters in the computation of the expectations. This form of the algorithm is not stated in detail, but referred to as GMPpostExtLin for GMP with post-processed extrinsic at the linear estimation.

If it is assumed that the non-linear estimation in the signal-constrained factor behaves similarly for pre- and post-processed extrinsic, then the same strategy can be applied again to the computation of the respective estimates, i.e., one can switch from pre- to post-processed extrinsic. The resulting algorithm is shown in Appendix E.2 as Algorithm E.2—it is called GMPpostExt—and benefits from the fact that the input parameters for the signal-constrained estimation do not have to be computed edge-dependently. Therefore, the signal-constrained estimation can be computed once for all variables $x_1, \ldots, x_N$ and the "downward" message is obtained by subtracting only the message from the incoming edge.

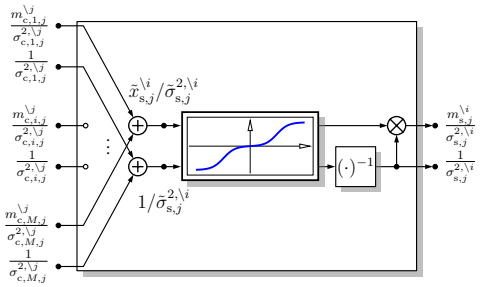
GMP and its variants can be composed from the block diagrams shown in Fig. 3.8. The GMP algorithm itself utilizes pre-processed extrinsic, i.e., only blocks from the left side (and perhaps



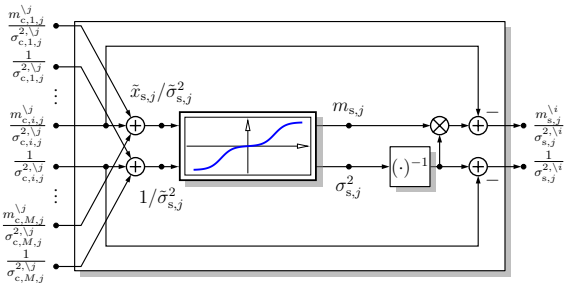
(a) Pre-processing of extrinsic at channel-constrained node; messages $m_{\mathrm{s},j}^{\backslash i}$ and $\sigma_{\mathrm{s},j}^{2,\backslash i}$ are left out.



(b) Post-processing of extrinsic at channel-constrained node; all entries are summed up, $m_{\mathrm{s},j}^{\backslash i}$ and $\sigma_{\mathrm{s},j}^{2,\backslash i}$ are scaled and subtracted afterwards.



(c) Pre-processed extrinsic at signal-constrained node, i.e., the input values of the desired edge are left out.



(d) Extrinsic at signal-constrained node is post-processed, i.e., the respective input is subtracted in the end.

Figure 3.8: Computations at the channel-constrained (upper row) and the signal-constrained nodes (lower row). The left side shows pre-processed, the right side shows post-processed extrinsic.

a suitable transformation from output to input parameters between the blocks). In contrast, GMPpostExt only uses the right-side blocks; and GMPpostExtLin mixes the approaches by taking the upper right and lower left blocks for its processing. Note that, for simplicity, the block representing the signal-constrained estimation accepts here scaled mean and inverse variance (similar to the implementation in (6.1)–(6.3)) as inputs and returns conditional mean and variance.

In the literature, one can also find various approaches to mitigate the scalability problem by simplifying the strategy. In [KMS+12] the complexity is reduced by restricting to signal-constrained and channel-constrained estimates for each variable x_j while also keeping two estimates (mean and variance) per observation y_i, the resulting algorithm is called *generalized approximate message passing* (GAMP) [Ran11]. Further simplifications, e.g., the step from individual variances to average variances and the insertion of the observation dependent parts into the calculation of the residuum, lead to the regular form of *approximate message passing* (AMP) [DMM09, Mal11]. Noteworthy, the LMMSE solution in the linear estimation is simplified to a matched filter (MF), i.e., the transpose of the sensing matrix is used without the inverse that is computed in the LMMSE estimator. The AMP and the GAMP algorithm are stated in the notation used here in Appendices E.3 and E.4, respectively. AMP works well for Gaussian sensing matrices and possesses very low computational complexity, but deteriorates for more general classes of sensing matrices [CZK14].

With the focus on factor graphs and its various representations of the CS problem, we will not follow this path but instead consider the opposite approach to the fine-granular view that is considering only two factors.

4. Turbo-type Inference

The notion of Turbo-type processing was established in [BGT93] in the form of Turbo decoding. The idea is to consider codes composed of two component codes. Decoding can then be done for the component codes, which is less computationally costly, with the results being passed in a suitable way between the decoders, i.e., an iterative scheme is present; for an overview, see [HOP96]. The overall goal is, thus, the exchange of *messages* between two major parts constituting to the problem. The suitable way to process the message for sending leads again to the notion of *extrinsic* [BG96, GH11].

The Turbo principle has also been used in detection for digital transmission schemes and is known as *Turbo equalization* [DJB+95] there, for an overview see [TS11].

In this chapter, the processing of Turbo-type algorithms for CS is derived and connections to the famous VAMP algorithm [RSF19] are revealed. To that end, the same procedure of considering a constrained optimization problem for the approximation of the CS posterior—only this time for a different form of the factorization—as in the chapter before is employed. This shall show that the underlying principles are the same. For inference in general, the optimization approach has in [HOW+05] been shown to be equivalent to expectation-consistent (EC) approximate inference [OW05], which coincides with the VAMP algorithm in the given case.

For the case of Turbo codes, the connection to message passing (MP) has already been shown in [MMC98], [KF98], and [Wib96]. Noteworthy, VAMP is also derived from a message-passing-type view in [RSF19], however, the exposition here puts everything into one notation, states the derivation in detail and draws attention to the optimization point of view.

Other literature that considers optimization-based approaches, usually adds the optimization on top, so that a *double-loop algorithm* results [OW05, RFSK16]. Here, the focus lies on the optimization within the factor graph, yielding *single-loop algorithms* only.

The connection of Turbo-type processing and minimization of the Kullback–Leibler divergence is investigated in [MG98] for iterative decoding and in [Moh97a, Moh97b] for multi-user detection in digital transmission scenarios.

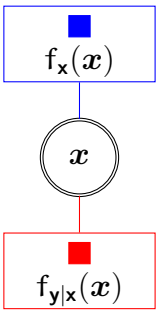


Figure 4.1: Factor graph for CS separating the two different kinds of factors.

4.1 Structure of the Factor Graph

The Turbo approach implies two opposing parts of a problem. In the compressed sensing case, this is given by considering the factorization

$$\mathsf{f}_{\mathsf{x}|\mathsf{y}}(\boldsymbol{x}) = \text{const.} \cdot \mathsf{f}_{\mathsf{y}|\mathsf{x}}(\boldsymbol{x}) \cdot \mathsf{f}_{\mathsf{x}}(\boldsymbol{x}) \tag{4.1}$$

of the posterior, which combines factors of the same kind in one factor, thereby explicitly separating channel and signal part while keeping the different components of each part together. The corresponding factor graph is shown in Fig. 2.3(d) and for convenience plotted here in Fig. 4.1 again.

The approach can be motivated by the fact that the parts, which complicate the overall estimation, are separated. In order to comprehend this, recall that the computation of the conditional mean (2.9) is infeasible, since the multi-dimensional integral depends on the signal prior $\mathsf{f}_{\mathsf{x}}(\boldsymbol{x})$ and can in general not be solved analytically. Further recall, that the estimators in Sec. 2.2.1.1 and Sec. 2.2.1.3 have been derived by replacing either of the factors with a Gaussian distribution, respectively, which resulted in feasible MMSE estimators. On the one hand, the multi-dimensionality of the estimator (2.10) can be handled, since the estimator is linear, which is optimal under the assumption of a Gaussian prior $\mathsf{f}_{\mathsf{x}}(\boldsymbol{x})$. If one or more signal components x_j were supposed to be distributed differently, the linear estimator would not be optimal and the corresponding multi-dimensional non-linear estimator could not be expressed, i.e., the computational effort would increase disproportionately. On the other hand, the non-linear estimators (2.19) can be handled because they are separated into one-dimensional integrals that can be solved, i.e., the multi-dimensionality is broken down, since the dependencies induced by the observations $\boldsymbol{y}$ and the sensing matrix $\boldsymbol{A}$ are ignored.

Hence, the approach in this chapter aims to separate what needs to be separated without going further into detail, i.e., what can be considered together is considered as one factor. Thereby, the issue of scalability, which arose for MP from the huge number of edges in the factor graph, is also addressed since the number of edges is decreased drastically. However, one has to keep in mind that instead of two scalars, the messages on the edges are now N-dimensional vectors.

4.1.1 Factorization of the Substitute Distribution

This more distanced view with less focus on the details has direct consequences on the form of the approximating distribution $\mathsf{q}(\boldsymbol{x})$. For the given factorization, the partial approximations $\mathsf{q}_{\mathrm{s}}(\boldsymbol{x})$ are used for $\mathsf{f}_{\mathbf{x}}(\boldsymbol{x})$, as well as $\mathsf{q}_{\mathrm{c}}(\boldsymbol{x})$ for $\mathsf{f}_{\mathbf{y}|\mathbf{x}}(\boldsymbol{x})$. Comparing to Chapter 3, the number of approximations is drastically reduced. Here, the adaptions by the variations may be written as follows

$$\begin{aligned}\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x}) &= \text{const.} \cdot \mathsf{f}_{\mathbf{x}}(\boldsymbol{x}) \cdot \mathsf{f}_{\mathbf{y}|\mathbf{x}}(\boldsymbol{x}) \cdot \frac{s(\boldsymbol{x}) \cdot c(\boldsymbol{x})}{s(\boldsymbol{x}) \cdot c(\boldsymbol{x})} \\ &= \text{const.} \cdot \frac{\mathsf{q}_{\mathrm{s}}(\boldsymbol{x}) \cdot \mathsf{q}_{\mathrm{c}}(\boldsymbol{x})}{s(\boldsymbol{x}) \cdot c(\boldsymbol{x})} \ , \end{aligned} \tag{4.2}$$

i.e., the representatives are defined as

$$\mathsf{q}_{\mathrm{s}}(\boldsymbol{x}) = \mathsf{f}_{\mathbf{x}}(\boldsymbol{x}) \cdot s(\boldsymbol{x}) \ , \tag{4.3}$$

$$\mathsf{q}_{\mathrm{c}}(\boldsymbol{x}) = \mathsf{f}_{\mathbf{y}|\mathbf{x}}(\boldsymbol{x}) \cdot c(\boldsymbol{x}) \ . \tag{4.4}$$

Since, only two deviations are present, the manipulated variable node obeys

$$\frac{\text{const.}}{s(\boldsymbol{x}) \cdot c(\boldsymbol{x})} = \text{const.} \int \frac{\delta(\boldsymbol{x} - \tilde{\boldsymbol{x}})}{s(\boldsymbol{x}) \cdot c(\tilde{\boldsymbol{x}})} \, \mathrm{d}\tilde{\boldsymbol{x}} \ , \tag{4.5}$$

so that the representative distribution for the variable node is defined as

$$\mathsf{q}_{\mathrm{v}}(\boldsymbol{x}) = \text{const.} \cdot s(\boldsymbol{x}) \cdot c(\boldsymbol{x}) \ . \tag{4.6}$$

All in all, the substitute distribution is given by

$$\mathsf{q}(\boldsymbol{x}) = \frac{\mathsf{q}_{\mathrm{s}}(\boldsymbol{x}) \cdot \mathsf{q}_{\mathrm{c}}(\boldsymbol{x})}{\mathsf{q}_{\mathrm{v}}(\boldsymbol{x})} \ . \tag{4.7}$$

4.1.2 The Moment-Matching Constraint

The connection between the approximated factors is routed via the (combined) variable node. Since the variable node has an edge to both factors, the consistency is now required on these edges. Our interest in moments, especially in the conditional mean (2.9), justifies the use of a moment constraint in this case as well. Therefore, exponential families are used again, i.e., a certain statistic is specified on which to rely the estimation and which should be matched in order to achieve consistency. That is, for an arbitrary sufficient statistic $\boldsymbol{g}(\boldsymbol{x})$ of the exponential family,

$$\mathrm{E}_{\mathbf{x} \sim \mathsf{q}_{\mathrm{v}}}\{\boldsymbol{g}(\mathbf{x})\} \overset{!}{=} \mathrm{E}_{\mathbf{x} \sim \mathsf{q}_{\mathrm{s}}}\{\boldsymbol{g}(\mathbf{x})\} \ , \tag{4.8}$$

$$\mathrm{E}_{\mathbf{x} \sim \mathsf{q}_{\mathrm{v}}}\{\boldsymbol{g}(\mathbf{x})\} \overset{!}{=} \mathrm{E}_{\mathbf{x} \sim \mathsf{q}_{\mathrm{c}}}\{\boldsymbol{g}(\mathbf{x})\} \tag{4.9}$$

is required. Note that depending on the sufficient statistic that specifies the form of consistency, the requirements and thereby the resulting messages are based on multi-dimensional vectors. For example, in Chapter 3, the *individual variances* case was employed, which leads to $2N$-dimensional messages per edge. Another possibility introduced in Appendix D is the *average variance* case yielding $N + 1$-dimensional messages. For now, this may stay open; later it will be specified for the respective algorithms.

4.2 Constrained Optimization

As in Chapter 3, the minimization of the Kullback–Leibler divergence is considered for the given structure to derive stationary points that are aspired to be achieved by recovery algorithms. For the structure at hand, there are vector-wise consistency constraints for the matching of moments and two approximating factors given. The constrained optimization problem reads

$$\begin{aligned} \text{minimize} \quad & \mathrm{D}_{\mathrm{KL}}\Big(\mathsf{q}(\boldsymbol{x}) \,||\, \mathsf{f}_{\mathsf{x}|\mathsf{y}}(\boldsymbol{x})\Big) \\ \text{subject to} \quad & \mathrm{E}_{\mathsf{x}\sim\mathsf{q}_\mathrm{v}}\{\boldsymbol{g}(\mathsf{x})\} - \mathrm{E}_{\mathsf{x}\sim\mathsf{q}_\mathrm{c}}\{\boldsymbol{g}(\mathsf{x})\} = \boldsymbol{0} \,, \\ & \mathrm{E}_{\mathsf{x}\sim\mathsf{q}_\mathrm{v}}\{\boldsymbol{g}(\mathsf{x})\} - \mathrm{E}_{\mathsf{x}\sim\mathsf{q}_\mathrm{s}}\{\boldsymbol{g}(\mathsf{x})\} = \boldsymbol{0} \,, \\ & \int \mathsf{q}_\mathrm{v}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} - 1 = 0 \,, \quad \int \mathsf{q}_\mathrm{s}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} - 1 = 0 \,, \int \mathsf{q}_\mathrm{c}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} - 1 = 0 \,. \end{aligned} \tag{4.10}$$

4.2.1 Cost Function for Compressed Sensing

For the given case, the Kullback–Leibler divergence to be minimized splits directly into

$$\begin{aligned} \mathrm{D}_{\mathrm{KL}}\Big(\mathsf{q}(\boldsymbol{x}) \,||\, \mathsf{f}_{\mathsf{x}|\mathsf{y}}(\boldsymbol{x})\Big) = & \int \mathsf{q}_\mathrm{s}(\boldsymbol{x}) \log \mathsf{q}_\mathrm{s}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} + \int \mathsf{q}_\mathrm{c}(\boldsymbol{x}) \log \mathsf{q}_\mathrm{c}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \\ & - \int \mathsf{q}_\mathrm{s}(\boldsymbol{x}) \log \mathsf{f}_{\mathsf{x}}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} - \int \mathsf{q}_\mathrm{c}(\boldsymbol{x}) \log \mathsf{f}_{\mathsf{y}|\mathsf{x}}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \\ & - \int \mathsf{q}_\mathrm{v}(\boldsymbol{x}) \log \mathsf{q}_\mathrm{v}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \,. \end{aligned} \tag{4.11}$$

For this form of the Kullback–Leibler divergence, the stationary points of the Lagrangian incorporating additionally the moment-matching constraints are derived. The stationary points are of interest, because they minimize the cost function under the given constraints, and will later be used to derive recovery algorithms.

4.2.2 Stationary Points of the Lagrangian

As before, the Lagrangian, cf. Appendix B, is used to insert the moment-matching constraint into the optimization problem. By introducing the Lagrangian multipliers $\boldsymbol{\nu}_\mathrm{c}$ and $\boldsymbol{\nu}_\mathrm{s}$ for the moment-matching constraints and ν_v, ν_c, and ν_s for the normalization constraints, the Lagrangian reads (again without multipliers in the argument)

$$\begin{aligned} \mathcal{L}(\mathsf{q}(\boldsymbol{x})) = & \, \mathrm{D}_{\mathrm{KL}}\Big(\mathsf{q}(\boldsymbol{x}) \,||\, \mathsf{f}_{\mathsf{x}|\mathsf{y}}(\boldsymbol{x})\Big) \\ & + \boldsymbol{\nu}_\mathrm{s}^\top \left(\mathrm{E}_{\mathsf{x},\mathsf{q}_\mathrm{v}}\{\boldsymbol{g}(\mathsf{x})\} - \mathrm{E}_{\mathsf{x},\mathsf{q}_\mathrm{s}}\{\boldsymbol{g}(\mathsf{x})\}\right) + \boldsymbol{\nu}_\mathrm{c}^\top \left(\mathrm{E}_{\mathsf{x},\mathsf{q}_\mathrm{v}}\{\boldsymbol{g}(\mathsf{x})\} - \mathrm{E}_{\mathsf{x},\mathsf{q}_\mathrm{c}}\{\boldsymbol{g}(\mathsf{x})\}\right) \\ & + \nu_\mathrm{v} \left(\int \mathsf{q}_\mathrm{v}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} - 1\right) + \nu_\mathrm{s} \left(\int \mathsf{q}_\mathrm{s}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} - 1\right) + \nu_\mathrm{c} \left(\int \mathsf{q}_\mathrm{c}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} - 1\right) \,. \end{aligned} \tag{4.12}$$

The optimal form for the distributions is given by the stationary points of the Lagrangian.

Hence, the Lagrangian is differentiated w.r.t. the distributions, which yields

$$\frac{\partial}{\partial \mathsf{q}_\mathrm{v}} \mathcal{L}(\mathsf{q}(\boldsymbol{x})) = -\log \mathsf{q}_\mathrm{v}(\boldsymbol{x}) + (\boldsymbol{\nu}_\mathrm{s} + \boldsymbol{\nu}_\mathrm{c})^\top \boldsymbol{g}(\boldsymbol{x}) + \text{const.} \,, \tag{4.13}$$

$$\frac{\partial}{\partial \mathsf{q}_\mathrm{s}} \mathcal{L}(\mathsf{q}(\boldsymbol{x})) = \log \mathsf{q}_\mathrm{s}(\boldsymbol{x}) - \log \mathsf{f}_{\mathbf{x}}(\boldsymbol{x}) - \boldsymbol{\nu}_\mathrm{s}^\top \boldsymbol{g}(\boldsymbol{x}) + \text{const.} \,, \tag{4.14}$$

$$\frac{\partial}{\partial \mathsf{q}_\mathrm{c}} \mathcal{L}(\mathsf{q}(\boldsymbol{x})) = \log \mathsf{q}_\mathrm{c}(\boldsymbol{x}) - \log \mathsf{f}_{\mathbf{y}|\mathbf{x}}(\boldsymbol{x}) - \boldsymbol{\nu}_\mathrm{c}^\top \boldsymbol{g}(\boldsymbol{x}) + \text{const.} \,. \tag{4.15}$$

By setting the above terms to zero, the respective stationary points, thus, calculate to

$$\mathsf{q}_\mathrm{v}(\boldsymbol{x}) = \frac{1}{Z_\mathrm{v}(\boldsymbol{\nu}_\mathrm{s}, \boldsymbol{\nu}_\mathrm{c})} \exp\Big((\boldsymbol{\nu}_\mathrm{s} + \boldsymbol{\nu}_\mathrm{c})^\top \boldsymbol{g}(\boldsymbol{x})\Big) \,, \tag{4.16}$$

$$\mathsf{q}_\mathrm{s}(\boldsymbol{x}) = \frac{1}{Z_\mathrm{s}(\boldsymbol{\nu}_\mathrm{s})} \mathsf{f}_{\mathbf{x}}(\boldsymbol{x}) \exp\Big(\boldsymbol{\nu}_\mathrm{s}^\top \boldsymbol{g}(\boldsymbol{x})\Big) \,, \tag{4.17}$$

$$\mathsf{q}_\mathrm{c}(\boldsymbol{x}) = \frac{1}{Z_\mathrm{c}(\boldsymbol{\nu}_\mathrm{c})} \mathsf{f}_{\mathbf{y}|\mathbf{x}}(\boldsymbol{x}) \exp\Big(\boldsymbol{\nu}_\mathrm{c}^\top \boldsymbol{g}(\boldsymbol{x})\Big) \,, \tag{4.18}$$

with normalizing partition functions[1]

$$Z_\mathrm{v}(\boldsymbol{\nu}_\mathrm{s}, \boldsymbol{\nu}_\mathrm{c}) = \int \exp\Big((\boldsymbol{\nu}_\mathrm{s} + \boldsymbol{\nu}_\mathrm{c})^\top \boldsymbol{g}(\boldsymbol{x})\Big) \, \mathrm{d}\boldsymbol{x} \,, \tag{4.19}$$

$$Z_\mathrm{s}(\boldsymbol{\nu}_\mathrm{s}) = \int \mathsf{f}_{\mathbf{x}}(\boldsymbol{x}) \exp\Big(\boldsymbol{\nu}_\mathrm{s}^\top \boldsymbol{g}(\boldsymbol{x})\Big) \, \mathrm{d}\boldsymbol{x} \,, \tag{4.20}$$

$$Z_\mathrm{c}(\boldsymbol{\nu}_\mathrm{c}) = \int \mathsf{f}_{\mathbf{y}|\mathbf{x}}(\boldsymbol{x}) \exp\Big(\boldsymbol{\nu}_\mathrm{c}^\top \boldsymbol{g}(\boldsymbol{x})\Big) \, \mathrm{d}\boldsymbol{x} \,. \tag{4.21}$$

These forms show directly the connection between variable and factor nodes, as the natural parameters of the variable node are given by

$$\boldsymbol{\theta} \overset{\text{def}}{=} \boldsymbol{\nu}_\mathrm{c} + \boldsymbol{\nu}_\mathrm{s} \,. \tag{4.22}$$

Considering that the two summands are the input parameters for the estimations $\mathrm{E}_{\mathbf{x}\sim\mathsf{q}_\mathrm{c}}\{\boldsymbol{g}(\mathbf{x})\}$, and $\mathrm{E}_{\mathbf{x}\sim\mathsf{q}_\mathrm{s}}\{\boldsymbol{g}(\mathbf{x})\}$, respectively, it is already visible how the extrinsic calculations are performed here. Which is to say, the extrinsic is obtained via *post-processing* of the parameter $\boldsymbol{\theta}$, i.e., the input for the other estimation is achieved by subtracting the input of the currently considered estimation from the result of the estimation. This is usual for Turbo-type approaches, e.g., in coding related publications [HOP96] a-priori log-likelihood ratios (LLRs) are subtracted from the decoding results of component codes to obtain the extrinsic values.

4.2.3 The Lagrange Dual Function

Before continuing with the derivation of algorithms for this approach, we take a step back and consider the *Lagrange dual function* of the problem. It is defined as infimum of the Lagrangian w.r.t. the optimization parameter, which is $\mathsf{q}(\boldsymbol{x})$ in the given case, i.e.,

$$\mathcal{L}_\mathrm{D}(\boldsymbol{\nu}_\mathrm{s}, \boldsymbol{\nu}_\mathrm{c}) = \inf_{\mathsf{q}(\boldsymbol{x})} \mathcal{L}(\mathsf{q}(\boldsymbol{x})) \,. \tag{4.23}$$

[1]The term *partition function* stems from exponential families, for details see Appendix D.

Therefore, it is a function of the Lagrangian multipliers only. It offers a different way in approaching the optimal value, namely from below. The so-called *dual problem* is the task of maximizing the Lagrange dual function. Since *strong duality* holds, the maximum value of $\mathcal{L}_\mathrm{D}(\boldsymbol{\nu}_\mathrm{s},\, \boldsymbol{\nu}_\mathrm{c})$ is indeed the desired optimum. For more information, see Appendix B. Noteworthy, the switch from primal to dual problem, also shifts the focus to the Lagrangian multipliers as parameters to be optimized.

Since the stationary points which minimize the Lagrangian are already given, the dual function is obtained by inserting the solution for the distributions $\mathrm{q}_\mathrm{v}(\boldsymbol{x})$, $\mathrm{q}_\mathrm{s}(\boldsymbol{x})$, and $\mathrm{q}_\mathrm{c}(\boldsymbol{x})$ into the Lagrangian. Note that at the stationary points the constraints must be fulfilled, i.e., they vanish. The insertion leads to

$$\begin{aligned}
\mathcal{L}_\mathrm{D}(\boldsymbol{\nu}_\mathrm{s},\, \boldsymbol{\nu}_\mathrm{c}) &= \int \mathrm{q}_\mathrm{s}(\boldsymbol{x}) \left(\log \mathrm{f}_{\mathbf{x}}(\boldsymbol{x}) + \boldsymbol{\nu}_\mathrm{s}^\top \boldsymbol{g}(\boldsymbol{x}) - \log Z_\mathrm{s}(\boldsymbol{\nu}_\mathrm{s})\right) \mathrm{d}\boldsymbol{x} \\
&\quad + \int \mathrm{q}_\mathrm{c}(\boldsymbol{x}) \left(\log \mathrm{f}_{\mathbf{y}|\mathbf{x}}(\boldsymbol{x}) + \boldsymbol{\nu}_\mathrm{c}^\top \boldsymbol{g}(\boldsymbol{x}) - \log Z_\mathrm{c}(\boldsymbol{\nu}_\mathrm{c})\right) \mathrm{d}\boldsymbol{x} \\
&\quad - \int \mathrm{q}_\mathrm{s}(\boldsymbol{x}) \log \mathrm{f}_{\mathbf{x}}(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} - \int \mathrm{q}_\mathrm{c}(\boldsymbol{x}) \log \mathrm{f}_{\mathbf{y}|\mathbf{x}}(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} \\
&\quad - \int \mathrm{q}_\mathrm{v}(\boldsymbol{x}) \left((\boldsymbol{\nu}_\mathrm{s} + \boldsymbol{\nu}_\mathrm{c})^\top \boldsymbol{g}(\boldsymbol{x}) - \log Z_\mathrm{v}(\boldsymbol{\nu}_\mathrm{s},\, \boldsymbol{\nu}_\mathrm{c})\right) \mathrm{d}\boldsymbol{x} \\
&= -\log Z_\mathrm{s}(\boldsymbol{\nu}_\mathrm{s}) \int \mathrm{q}_\mathrm{s}(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} - \log Z_\mathrm{c}(\boldsymbol{\nu}_\mathrm{c}) \int \mathrm{q}_\mathrm{c}(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} \\
&\quad + \log Z_\mathrm{v}(\boldsymbol{\nu}_\mathrm{s},\, \boldsymbol{\nu}_\mathrm{c}) \int \mathrm{q}_\mathrm{v}(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} \\
&\quad + \boldsymbol{\nu}_\mathrm{s}^\top \left(\mathrm{E}_{\mathbf{x},\mathrm{q}_\mathrm{s}}\{\boldsymbol{g}(\mathbf{x})\} - \mathrm{E}_{\mathbf{x},\mathrm{q}_\mathrm{v}}\{\boldsymbol{g}(\mathbf{x})\}\right) \\
&\quad + \boldsymbol{\nu}_\mathrm{c}^\top \left(\mathrm{E}_{\mathbf{x},\mathrm{q}_\mathrm{c}}\{\boldsymbol{g}(\mathbf{x})\} - \mathrm{E}_{\mathbf{x},\mathrm{q}_\mathrm{v}}\{\boldsymbol{g}(\mathbf{x})\}\right) \,. \qquad (4.24)
\end{aligned}$$

Since the equality constraints are fulfilled [HOW+05]

$$\mathcal{L}_\mathrm{D}(\boldsymbol{\nu}_\mathrm{s},\, \boldsymbol{\nu}_\mathrm{c}) = -\log Z_\mathrm{s}(\boldsymbol{\nu}_\mathrm{s}) - \log Z_\mathrm{c}(\boldsymbol{\nu}_\mathrm{c}) + \log Z_\mathrm{v}(\boldsymbol{\nu}_\mathrm{s},\, \boldsymbol{\nu}_\mathrm{c}) \qquad (4.25)$$

results. This means that alternatively to minimizing the Kullback–Leibler divergence between densities, an approach to solve the CS problem is to adjust the parameters $\boldsymbol{\nu}_\mathrm{s}$ and $\boldsymbol{\nu}_\mathrm{c}$ such that the cost function (4.25) is maximized. Noteworthy, this cost function already incorporates the moment-matching and normalization constraints in itself due to the definition of the partition functions $Z_\mathrm{v}(\boldsymbol{\nu}_\mathrm{s},\, \boldsymbol{\nu}_\mathrm{c})$, $Z_\mathrm{s}(\boldsymbol{\nu}_\mathrm{s})$, and $Z_\mathrm{c}(\boldsymbol{\nu}_\mathrm{c})$.

Equation (4.25) is the cost function that the expectation-consistent (EC) approximate inference framework [OW05] builds upon. There, so-called *single-loop algorithms* are derived, which are in the given case equivalent to VAMP or variants of it. Furthermore, it is suggested to optimize the Lagrange dual (4.25) partially, which leads to so-called *double-loop algorithms*.

The connection between EC, as well as EP (expectation propagation) [Min01] and convex optimization has already been shown in [HOW+05] for general inference problems.

4.3 Turbo-Type Algorithms—Optimization-Based Derivation

Having the stationary points of the Lagrangian, we can now turn to the derivation of algorithms by iteratively following the restrictions imposed by the structure of the densities.

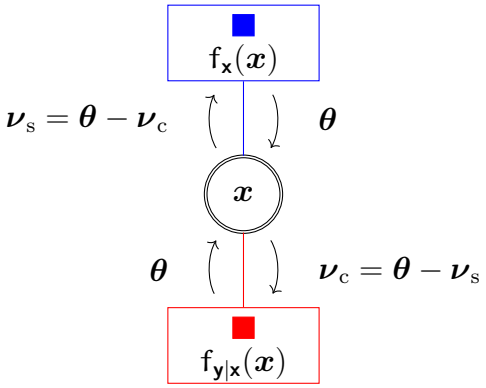


Figure 4.2: Factor graph for CS of the Turbo kind with respective messages.

The optimization-based view on the problem gives us already strict connections between the (natural) parameters, cf. (4.22). In Chapter 3, the use of this computation was called *post-processed* extrinsic, because an outgoing message on an edge could be obtained by subtracting the influence of the incoming message on the same edge from the result of the estimation that was obtained from all incoming messages. In the given case, each factor node is connected with only one edge to the variable node. Hence, for the given structure, the extrinsic computed at one factor node is directly the input to the other factor as indicated in (4.22). The respective forwarding of the messages is depicted in Fig. 4.2.

The mapping of the estimates $\mathrm{E}_{\mathbf{x}\sim q_{\mathrm{s}}}\{\boldsymbol{g}(\mathbf{x})\}$, respectively $\mathrm{E}_{\mathbf{x}\sim q_{\mathrm{c}}}\{\boldsymbol{g}(\mathbf{x})\}$ to the natural parameters $\boldsymbol{\theta}$ depend on the form of the exponential family, i.e., on the definition of the sufficient statistics $\boldsymbol{g}(\boldsymbol{x})$. In the following, two cases for the specification of $\boldsymbol{g}(\boldsymbol{x})$ are considered, starting with average variances.

4.3.1 Average Variances—VAMP

The aspired estimate (2.9) is a conditional *mean* that specifies the value of the components of the signal vector $\boldsymbol{x}$ for the given observations $\boldsymbol{y}$. The advantage of working with a Gaussian density as exponential family is that it additionally specifies the variance or second order moment, thereby introducing a measure of reliability for the random variable, thus, also for the estimate.

In Chapter 3, the individual variance case was considered, where each estimate had its own variance, i.e., reliability. Due to the detailed view on the factor graph and the resulting scalar estimations this specification was necessary. In the given case, where always the entire vector is processed, this constraint can be relaxed to considering a variance that represents the average reliability of the entire vector, which yields a complexity reduction in terms of the number of messages to be sent.

The average variance case of a Gaussian density is specified in (D.11), i.e.,

$$\boldsymbol{g}(\boldsymbol{x}) = [\boldsymbol{g}_{\boldsymbol{\lambda}}(\boldsymbol{x})^{\top},\, g_{\Lambda}(\boldsymbol{x})^{\top}]^{\top} \tag{4.26}$$

with

$$\boldsymbol{g}_{\boldsymbol{\lambda}}(\boldsymbol{x}) = \boldsymbol{x} \ , \quad g_{\Lambda}(\boldsymbol{x}) = -\frac{1}{2}\boldsymbol{x}^{\top}\boldsymbol{x} \ . \tag{4.27}$$

This specification will eventually lead to the VAMP algorithm [RSF19].

Let us consider the moment computed at the factor nodes first. Obviously, the estimation at the channel factor is related to the LMMSE estimation in Sec. 2.2.1.1. To compare with those estimators, regard that an average variance at the input is equivalent to $\boldsymbol{\Phi}_{\mathrm{x}} = \sigma_{\mathrm{x}}^2 \boldsymbol{I}_N$. With the identification $\boldsymbol{\nu}_{\mathrm{c}} = [\boldsymbol{m}_{\mathrm{x}}^{\top}/\sigma_{\mathrm{x}}^2, \, 1/\sigma_{\mathrm{x}}^2]^{\top}$, it is ($\mathrm{trace}(\boldsymbol{M}) = \sum_{j=1}^{N}[\boldsymbol{M}]_{jj}$)

$$\mathrm{E}_{\mathbf{x}\sim \mathrm{q_c}}\{\boldsymbol{g}_{\boldsymbol{\lambda}}(\mathbf{x})\} = \boldsymbol{m}_{\mathrm{c}} \ , \tag{4.28}$$

$$\mathrm{E}_{\mathbf{x}\sim \mathrm{q_c}}\{g_{\Lambda}(\mathbf{x})\} = -\frac{1}{2}\mathrm{trace}\Big(\mathrm{E}_{\mathbf{x}}\Big\{\mathbf{x}\mathbf{x}^{\top} \mid \boldsymbol{y}\Big\}\Big) = -\frac{1}{2}\mathrm{trace}\Big(\boldsymbol{\Phi}_{\mathrm{c}} + \boldsymbol{m}_{\mathrm{c}}\boldsymbol{m}_{\mathrm{c}}^{\top}\Big) \ . \tag{4.29}$$

The resulting average variance is then given by

$$\sigma_{\mathrm{c}}^2 \overset{\mathrm{def}}{=} \frac{1}{N}\mathrm{trace}(\boldsymbol{\Phi}_{\mathrm{c}}) = -\frac{2}{N}\mathrm{E}_{\mathbf{x}\sim \mathrm{q_c}}\{g_{\Lambda}(\mathbf{x})\} - \boldsymbol{m}_{\mathrm{c}}^{\top}\boldsymbol{m}_{\mathrm{c}} \ . \tag{4.30}$$

The mapping to the natural parameter $\boldsymbol{\theta}$ of the variable node is then obtained via

$$\boldsymbol{\theta} = [\boldsymbol{m}_{\mathrm{c}}^{\top}/\sigma_{\mathrm{c}}^2, \, 1/\sigma_{\mathrm{c}}^2]^{\top} \ , \tag{4.31}$$

and the extrinsic (which serves as input for the signal factor) is obtained via (4.22), i.e.,

$$\boldsymbol{\nu}_{\mathrm{s}} = \boldsymbol{\theta} - \boldsymbol{\nu}_{\mathrm{c}} \ . \tag{4.32}$$

For the signal factor, the estimator similarly relates to (2.19) and (2.20) from Sec. 2.2.1.3 with $\boldsymbol{\nu}_{\mathrm{s}} = [\tilde{\boldsymbol{x}}_{\mathrm{s}}^{\top}/\tilde{\sigma}_{\mathrm{s}}^2, \, 1/\tilde{\sigma}_{\mathrm{s}}^2]^{\top}$, i.e., in the average variance case it is $\tilde{\sigma}_{\mathrm{s},j}^2 = \tilde{\sigma}_{\mathrm{s}}^2$ for all $j \in \{1, \, \ldots, \, N\}$, as follows

$$\mathrm{E}_{\mathbf{x}\sim \mathrm{q_s}}\{\boldsymbol{g}_{\boldsymbol{\lambda}}(\mathbf{x})\} = \mathrm{E}_{\mathbf{x}}\{\mathbf{x} \mid \tilde{\boldsymbol{x}}_{\mathrm{s}}\} = \boldsymbol{m}_{\mathrm{s}} \ , \tag{4.33}$$

$$\mathrm{E}_{\mathbf{x}\sim \mathrm{q_s}}\{g_{\Lambda}(\mathbf{x})\} = -\frac{1}{2}\mathrm{trace}\Big(\mathrm{E}_{\mathbf{x}}\Big\{\mathbf{x}\mathbf{x}^{\top} \mid \tilde{\boldsymbol{x}}_{\mathrm{s}}\Big\}\Big) = -\frac{1}{2}\sum_{j=1}^{N}(\sigma_{\mathrm{s},j}^2 + m_{\mathrm{s},j}^2) \ , \tag{4.34}$$

and a respective average variance is obtained via

$$\sigma_{\mathrm{s}}^2 \overset{\mathrm{def}}{=} \frac{1}{N}\sum_{j=1}^{N}\sigma_{\mathrm{s},j}^2 = -\frac{2}{N}\mathrm{E}_{\mathbf{x}\sim \mathrm{q_s}}\{g_{\Lambda}(\mathbf{x})\} - \boldsymbol{m}_{\mathrm{s}}^{\top}\boldsymbol{m}_{\mathrm{s}} \ . \tag{4.35}$$

The mapping is analogously $\boldsymbol{\theta} = [\boldsymbol{m}_{\mathrm{s}}^{\top}/\sigma_{\mathrm{s}}^2, 1/\sigma_{\mathrm{s}}^2]^{\top}$ and the extrinsic calculation $\boldsymbol{\nu}_{\mathrm{c}} = \boldsymbol{\theta} - \boldsymbol{\nu}_{\mathrm{s}}$, respectively.

Given a suitable initialization, the iterative processing of the steps above yields a recovery algorithm for the CS problem, which is termed VAMP in the literature. The entire algorithm is stated in pseudo code in Appendix E.5. Note that the use of natural parameters is omitted in the description by directly computing mean and variance via the respective transformations.

Noteworthy, an algorithm called *orthogonal approximate message passing* (OAMP) [MP17] was proposed simultaneously to VAMP, essentially identical.

4.3.2 Individual Variances

The VAMP algorithm can simply be varied by specifying the sufficient statistics differently. Two cases have been considered in [FSRS16]: so-called *uniform diagonalization*, which equals the specification we termed *average variance*, and *vector-valued diagonalization*, leading to *individual variances*. This second case may lead to better insights into the estimates, as the reliability is evaluated on an individual level. However, the complexity increases in terms of messages—from $N+1$-dimensional to $2N$-dimensional vectors—and also the computation of the LMMSE estimate gets more difficult. In [FSRS16], generalizations of the EC framework [OW05], i.e., variants of VAMP, are considered for general inference problems, but not the compressed sensing problem itself.

By changing the specification, the basic processing steps stay the same as in VAMP; only the moment computation and the mapping from and to natural parameters differ.

Mathematically, the sufficient statistic need to be defined by $\boldsymbol{g}(\boldsymbol{x}) = [\boldsymbol{g}_{\boldsymbol{\lambda}}(\boldsymbol{x})^\top, \boldsymbol{g}_{\boldsymbol{\Lambda}}(\boldsymbol{x})^\top]^\top$ with

$$\boldsymbol{g}_{\boldsymbol{\lambda}}(\boldsymbol{x}) = \boldsymbol{x} \;, \tag{4.36}$$

$$\boldsymbol{g}_{\boldsymbol{\Lambda}}(\boldsymbol{x}) = -\frac{1}{2}\mathbf{diag}\Big(\boldsymbol{x}\boldsymbol{x}^\top\Big) \;. \tag{4.37}$$

The individual variances case for the estimates in (2.10) and (2.11) is obtained by specifying $\boldsymbol{\Phi}_{\mathrm{x}} = \mathbf{diag}(\sigma_{\mathrm{x},j}^2)$. To compare with the estimators from Sections 2.2.1.1 and 2.2.1.3, the input variables are identified by

$$\boldsymbol{\nu}_{\mathrm{c}} = [m_{\mathrm{x},1}/\sigma_{\mathrm{x},1}^2, \ldots, m_{\mathrm{x},N}/\sigma_{\mathrm{x},N}^2, 1/\sigma_{\mathrm{x},1}^2, \ldots, 1/\sigma_{\mathrm{x},N}^2]^\top \;, \tag{4.38}$$

respectively,

$$\boldsymbol{\nu}_{\mathrm{s}} = [\tilde{x}_{\mathrm{s},1}/\tilde{\sigma}_{\mathrm{s},1}^2, \ldots, \tilde{x}_{\mathrm{s},N}/\tilde{\sigma}_{\mathrm{s},N}^2, 1/\tilde{\sigma}_{\mathrm{s},1}^2, \ldots, 1/\tilde{\sigma}_{\mathrm{s},N}^2]^\top \;. \tag{4.39}$$

The expectations w.r.t. $\boldsymbol{g}_{\boldsymbol{\lambda}}(\boldsymbol{x})$ are the same as for the average variances case. For the second order moment, the channel-constrained estimation reads

$$\mathrm{E}_{\mathbf{x}\sim\mathrm{q_c}}\{\boldsymbol{g}_{\boldsymbol{\Lambda}}(\mathbf{x})\} = -\frac{1}{2}\mathbf{diag}\Big(\boldsymbol{\Phi}_{\mathrm{c}} + \boldsymbol{m}_{\mathrm{c}}\boldsymbol{m}_{\mathrm{c}}^\top\Big) \;, \tag{4.40}$$

respectively

$$\mathrm{E}_{\mathbf{x}\sim\mathrm{q_s}}\{\boldsymbol{g}_{\boldsymbol{\Lambda}}(\mathbf{x})\} = -\frac{1}{2}\mathbf{diag}\Big(\mathbf{diag}\Big(\sigma_{\mathrm{s},j}^2 + m_{\mathrm{s},j}^2\Big)\Big) \;, \tag{4.41}$$

for the signal-constrained estimation. The mapping to the natural parameters is then given by

$$\boldsymbol{\theta} = [m_{\bullet,1}/\sigma_{\bullet,1}^2, \ldots, m_{\bullet,N}/\sigma_{\bullet,N}^2, 1/\sigma_{\bullet,1}^2, \ldots, 1/\sigma_{\bullet,N}^2]^\top \tag{4.42}$$

with $\bullet \in \{\mathrm{c}, \mathrm{s}\}$, respectively. The extrinsic calculation is the same as before, i.e., subtraction of the natural parameters. The entire algorithm is stated in Algorithm E.6; we call it VAMPind, for VAMP with individual variances. Again, the representation with means and variances is used in the statement of the algorithm.

4.4 Estimation-Theoretic Bias Compensation

The simple additive connection (4.22) between the natural parameters of variable and factor nodes turns out to be crucial for the processing in the algorithms. So far, it was called the computation of *extrinsic*, since previously known values are subtracted [BG96, GH11].

In [SF17, SF18] a connection to *bias compensation* [For03] from an estimation theoretical point of view was drawn. This already shows that different perspectives can yield the same results, but also open the view for more possibilities, as will be visible soon.

Bias compensation describes the task of removing the *bias*, which is a systematic offset in the estimate, from the estimate. The procedure usually degrades the reliability of the estimate [For03, Fis02]. MMSE estimates are always biased, since they are by definition orthogonal to the error. This causes a part of the signal (to be estimated) to be accounted for the error [For03]. The compensation is then usually realized by scaling the estimate adequately [Fis02].

In the following, forms of post-processing for the extrinsic based on the estimation-theoretical viewpoint of bias compensation are derived and compared to the extrinsic calculations obtained so far. Similar derivations can be found in [SF18, Spa19, FS22, SF22].

There are two kinds of estimations to be considered; the linear estimation which is used for the channel part of CS and the non-linear estimation as in the signal part; starting with the derivations for the channel-constrained estimations, i.e., $\boldsymbol{m}_{\mathrm{c}}$ as defined in (2.10).

4.4.1 Channel-Constrained Estimation

The bias in the channel-constrained estimate

$$\boldsymbol{m}_{\mathrm{c}} = \boldsymbol{m}_{\mathsf{x}} + \boldsymbol{\Phi}_{\mathsf{x}} \boldsymbol{A}^{\mathsf{T}} (\boldsymbol{A} \boldsymbol{\Phi}_{\mathsf{x}} \boldsymbol{A}^{\mathsf{T}} + \sigma_{\mathsf{n}}^2 \boldsymbol{I}_M)^{-1} (\boldsymbol{y} - \boldsymbol{A} \boldsymbol{m}_{\mathsf{x}}) \tag{4.43}$$

can be seen in the fact that the end-to-end cascade $\boldsymbol{K}$ between signal $\boldsymbol{x}$ and estimate $\boldsymbol{m}_{\mathrm{c}}$, which is given by [Spa19, FS22]

$$\begin{aligned} \boldsymbol{K} &= \boldsymbol{\Phi}_{\mathsf{x}} \boldsymbol{A}^{\mathsf{T}} (\boldsymbol{A} \boldsymbol{\Phi}_{\mathsf{x}} \boldsymbol{A}^{\mathsf{T}} + \sigma_{\mathsf{n}}^2 \boldsymbol{I}_M)^{-1} \boldsymbol{A} \\ &= (\boldsymbol{A}^{\mathsf{T}} \boldsymbol{A} + \sigma_{\mathsf{n}}^2 \boldsymbol{\Phi}_{\mathsf{x}}^{-1})^{-1} \boldsymbol{A}^{\mathsf{T}} \boldsymbol{A} \\ &= \boldsymbol{I}_N - \boldsymbol{\Phi}_{\mathrm{c}} \boldsymbol{\Phi}_{\mathsf{x}}^{-1} \,, \end{aligned} \tag{4.44}$$

is not an identity matrix, which can be easily verified reformulating $\boldsymbol{\Phi}_{\mathrm{c}}$ from (2.11) via the Sherman-Morrison-Woodbury identity [GV96]. A visualization of the end-to-end cascade by block diagrams is given in Fig. 4.3. The compensation for an unbiased estimate $\boldsymbol{m}_{\mathrm{c,u}}$ can be done by inserting a respective scaling matrix $\boldsymbol{W}$ in the block diagram (shown in Fig. 4.4) under various restrictions.

4.4.1.1 Average Unbiasing

First, a rather loose requirement is considered; the effect of the end-to-end cascade should be compensated *on average* [Fis16]. This requirement is suitable as long as the energies (squared

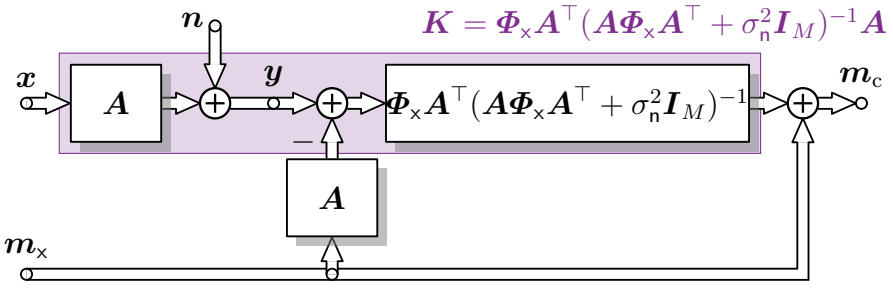


Figure 4.3: Visualization of the end-to-end cascade $\boldsymbol{K}$ between signal $\boldsymbol{x}$ and estimate $\boldsymbol{m}_{\mathrm{c}}$.

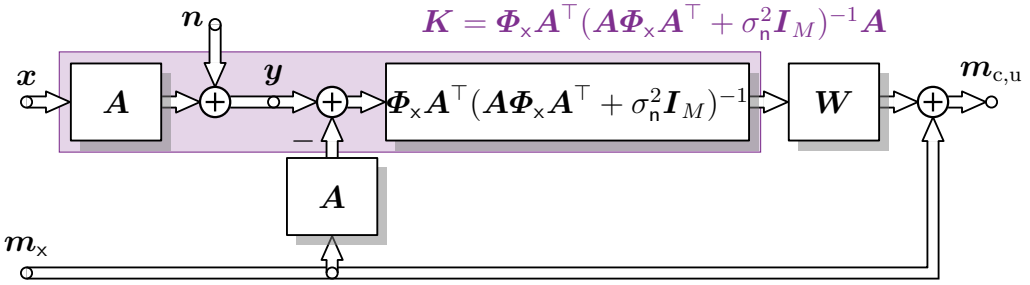


Figure 4.4: Visualization of the bias compensation procedure with scaling matrix $\boldsymbol{W}$.

ℓ_2-norms) of the columns of the sensing matrix as well as the variances of signal components of $\boldsymbol{x}$ do not vary much. For this procedure, it is therefore assumed that $\boldsymbol{\Phi}_{\mathsf{x}} = \sigma_{\mathsf{x}}^2\boldsymbol{I}_N$. Mathematically, the requirement means, a scaling matrix $\boldsymbol{W}$ must fulfill [Fis16, FS22]

$$\frac{1}{N}\mathrm{trace}(\boldsymbol{W}\cdot\boldsymbol{K}) \stackrel{!}{=} 1\ . \tag{4.45}$$

The solution is

$$\boldsymbol{W} = \frac{1}{\frac{1}{N}\mathrm{trace}(\boldsymbol{K})}\boldsymbol{I}_N\ , \tag{4.46}$$

i.e., there is a single unbiasing factor $w = N/\mathrm{trace}(\boldsymbol{K})$, which is used to (re-)scale all elements of $\boldsymbol{m}_{\mathrm{c}}$. The resulting unbiased estimate $\boldsymbol{m}_{\mathrm{c,u}}$ is then given by

$$\boldsymbol{m}_{\mathrm{c,u}} = \boldsymbol{m}_{\mathsf{x}} + \boldsymbol{W}\boldsymbol{\Phi}_{\mathsf{x}}\boldsymbol{A}^{\top}(\boldsymbol{A}\boldsymbol{\Phi}_{\mathsf{x}}\boldsymbol{A}^{\top} + \sigma_{\mathsf{n}}^2\boldsymbol{I}_M)^{-1}(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{m}_{\mathsf{x}})\ . \tag{4.47}$$

Noteworthy, with $\boldsymbol{\Phi}_{\mathsf{x}} = \sigma_{\mathsf{x}}^2\boldsymbol{I}_N$, one obtains from (4.44)

$$\begin{aligned} w &= \frac{N}{\mathrm{trace}(\boldsymbol{K})} = \frac{N}{\mathrm{trace}(\boldsymbol{I}_N - \boldsymbol{\Phi}_{\mathrm{c}}\boldsymbol{\Phi}_{\mathsf{x}}^{-1})} \\ &= \frac{1}{1 - \frac{1}{\sigma_{\mathsf{x}}^2}\frac{1}{N}\mathrm{trace}(\boldsymbol{\Phi}_{\mathrm{c}})} \\ &= \frac{1}{1 - \frac{\sigma_{\mathrm{c}}^2}{\sigma_{\mathsf{x}}^2}} = \frac{\sigma_{\mathsf{x}}^2}{\sigma_{\mathsf{x}}^2 - \sigma_{\mathrm{c}}^2}\ . \end{aligned} \tag{4.48}$$

The covariance matrix corresponding to estimate $\boldsymbol{m}_{\mathrm{c,u}}$ reads [Spa19]

$$\boldsymbol{\Phi}_{\mathrm{c,u}} = \mathrm{E}_{\mathsf{x}}\Big\{(\mathsf{x} - \boldsymbol{m}_{\mathrm{c,u}})(\mathsf{x} - \boldsymbol{m}_{\mathrm{c,u}})^{\top} \mid \boldsymbol{y}\Big\} = \boldsymbol{\Phi}_{\mathsf{x}} - \boldsymbol{\Phi}_{\mathsf{x}}\boldsymbol{K}^{\top}\boldsymbol{W}^{\top} - \boldsymbol{W}\boldsymbol{K}\boldsymbol{\Phi}_{\mathsf{x}} + \boldsymbol{W}\boldsymbol{\Phi}_{\mathsf{x}}\boldsymbol{K}^{\top}\boldsymbol{W}^{\top}\ . \tag{4.49}$$

The average unbiased variance therefore reads ($\mathrm{trace}(\boldsymbol{W}\cdot\boldsymbol{K}) = \mathrm{trace}(\boldsymbol{K}^\top\cdot\boldsymbol{W}^\top) = N$)

$$\begin{aligned}\sigma_{\mathrm{c,u}}^2 &= \frac{1}{N}\mathrm{trace}(\boldsymbol{\Phi}_{\mathrm{c,u}}) = \sigma_{\mathsf{x}}^2 - \sigma_{\mathsf{x}}^2 - \sigma_{\mathsf{x}}^2 + w\sigma_{\mathsf{x}}^2 = \sigma_{\mathsf{x}}^2(w-1)\\ &= \sigma_{\mathsf{x}}^2\left(\frac{\sigma_{\mathsf{x}}^2}{\sigma_{\mathsf{x}}^2-\sigma_{\mathrm{c}}^2}-1\right) = \sigma_{\mathsf{x}}^2\frac{\sigma_{\mathsf{x}}^2-\sigma_{\mathsf{x}}^2+\sigma_{\mathrm{c}}^2}{\sigma_{\mathsf{x}}^2-\sigma_{\mathrm{c}}^2} = \frac{\sigma_{\mathsf{x}}^2\sigma_{\mathrm{c}}^2}{\sigma_{\mathsf{x}}^2-\sigma_{\mathrm{c}}^2} = \left(\frac{1}{\sigma_{\mathrm{c}}^2}-\frac{1}{\sigma_{\mathsf{x}}^2}\right)^{-1}\quad. \end{aligned} \tag{4.50}$$

So, the unbiased estimate can be written as

$$\begin{aligned}\boldsymbol{m}_{\mathrm{c,u}} &= \boldsymbol{m}_{\mathsf{x}} + \frac{\sigma_{\mathsf{x}}^2}{\sigma_{\mathsf{x}}^2-\sigma_{\mathrm{c}}^2}\boldsymbol{\Phi}_{\mathsf{x}}\boldsymbol{A}^\top(\boldsymbol{A}\boldsymbol{\Phi}_{\mathsf{x}}\boldsymbol{A}^\top+\sigma_{\mathsf{n}}^2\boldsymbol{I}_M)^{-1}(\boldsymbol{y}-\boldsymbol{A}\boldsymbol{m}_{\mathsf{x}})\\ &= \left(1-\frac{\sigma_{\mathsf{x}}^2}{\sigma_{\mathsf{x}}^2-\sigma_{\mathrm{c}}^2}+\frac{\sigma_{\mathsf{x}}^2}{\sigma_{\mathsf{x}}^2-\sigma_{\mathrm{c}}^2}\right)\boldsymbol{m}_{\mathsf{x}} + \frac{\sigma_{\mathsf{x}}^2}{\sigma_{\mathsf{x}}^2-\sigma_{\mathrm{c}}^2}\boldsymbol{\Phi}_{\mathsf{x}}\boldsymbol{A}^\top(\boldsymbol{A}\boldsymbol{\Phi}_{\mathsf{x}}\boldsymbol{A}^\top+\sigma_{\mathsf{n}}^2\boldsymbol{I}_M)^{-1}(\boldsymbol{y}-\boldsymbol{A}\boldsymbol{m}_{\mathsf{x}})\\ &= \frac{\sigma_{\mathrm{c}}^2}{\sigma_{\mathrm{c}}^2-\sigma_{\mathsf{x}}^2}\boldsymbol{m}_{\mathsf{x}} + \frac{\sigma_{\mathsf{x}}^2}{\sigma_{\mathsf{x}}^2-\sigma_{\mathrm{c}}^2}\boldsymbol{m}_{\mathrm{c}} = \sigma_{\mathrm{c,u}}^2(\boldsymbol{m}_{\mathrm{c}}/\sigma_{\mathrm{c}}^2-\boldsymbol{m}_{\mathsf{x}}/\sigma_{\mathsf{x}}^2)\,. \end{aligned} \tag{4.51}$$

By slightly rearranging these equations to

$$\boldsymbol{m}_{\mathrm{c,u}}/\sigma_{\mathrm{c,u}}^2 = \boldsymbol{m}_{\mathrm{c}}/\sigma_{\mathrm{c}}^2 - \boldsymbol{m}_{\mathsf{x}}/\sigma_{\mathsf{x}}^2\,,\qquad 1/\sigma_{\mathrm{c,u}}^2 = 1/\sigma_{\mathrm{c}}^2 - 1/\sigma_{\mathsf{x}}^2\,, \tag{4.52}$$

one can make the connection $\tilde{\boldsymbol{x}}_{\mathrm{c}} = \boldsymbol{m}_{\mathsf{x}}$, respectively $\tilde{\sigma}_{\mathrm{c}}^2 = \sigma_{\mathsf{x}}^2$, as well as $\boldsymbol{m}_{\mathrm{c,u}} = \tilde{\boldsymbol{x}}_{\mathrm{s}}$ and $\sigma_{\mathrm{c,u}}^2 = \tilde{\sigma}_{\mathrm{s}}^2$, to see that the bias compensation procedure equals the exchange after the linear estimation in the VAMP algorithm. This means that VAMP performs (at this point) average bias compensation with an average variance.

4.4.1.2 Individual Unbiasing

A stricter requirement is the compensation of the individual scaling effects that $\boldsymbol{K}$ causes on the signal components. In this case, the scaling matrix $\boldsymbol{W}$ must fulfill [FS07, FS22]

$$\mathbf{diag}(\boldsymbol{W}\cdot\boldsymbol{K}) \stackrel{!}{=} \boldsymbol{I}_N\,. \tag{4.53}$$

This is obtained for scaling matrix

$$\boldsymbol{W} = \mathbf{diag}(1/[\boldsymbol{K}]_{jj})\,. \tag{4.54}$$

Being more careful in the compensation, allows us to handle variations in the energies (squared ℓ_2-norms) of both, columns of the sensing matrix, as well as the signal components itself. This procedure is, thus, suitable for the use of individual variances, i.e., the more general case $\boldsymbol{\Phi}_{\mathsf{x}} = \mathbf{diag}(\sigma_{\mathsf{x},j}^2)$ is considered. The diagonal entries of the compensation matrix can be calculated via (4.44) [Spa19]

$$[\boldsymbol{W}]_{jj} = \frac{1}{[\boldsymbol{K}]_{jj}} = \frac{1}{1-\sigma_{\mathrm{c},j}^2/\sigma_{\mathsf{x},j}^2} = \frac{\sigma_{\mathsf{x},j}^2}{\sigma_{\mathsf{x},j}^2-\sigma_{\mathrm{c},j}^2}\,, \tag{4.55}$$

which resembles the unbiasing factor before, only with the individual variances. The (individual) unbiased variances compute to ($[\boldsymbol{W}]_{jj}\cdot[\boldsymbol{K}]_{jj} = 1$)

$$\begin{aligned}\sigma_{\mathrm{c,u},j}^2 \stackrel{\mathrm{def}}{=} [\boldsymbol{\Phi}_{\mathrm{c,u}}]_{jj} &= \sigma_{\mathsf{x},j}^2 - \sigma_{\mathsf{x},j}^2 - \sigma_{\mathsf{x},j}^2 + [\boldsymbol{W}]_{jj}\sigma_{\mathsf{x},j}^2 = \sigma_{\mathsf{x},j}^2([\boldsymbol{W}]_{jj}-1)\\ &= \sigma_{\mathsf{x},j}^2\left(\frac{\sigma_{\mathsf{x},j}^2}{\sigma_{\mathsf{x},j}^2-\sigma_{\mathrm{c},j}^2}-1\right) = \frac{\sigma_{\mathsf{x},j}^2\sigma_{\mathrm{c},j}^2}{\sigma_{\mathsf{x},j}^2-\sigma_{\mathrm{c},j}^2} = \left(\frac{1}{\sigma_{\mathrm{c},j}^2}-\frac{1}{\sigma_{\mathsf{x},j}^2}\right)^{-1}\,, \end{aligned} \tag{4.56}$$

leading to an unbiased estimate of the form

$$\begin{aligned}\boldsymbol{m}_{\mathrm{c,u}} &= \boldsymbol{m}_{\mathsf{x}} + \mathbf{diag}\left(\frac{\sigma_{\mathrm{x},j}^2}{\sigma_{\mathrm{x},j}^2 - \sigma_{\mathrm{c},j}^2}\right)\boldsymbol{\Phi}_{\mathsf{x}}\boldsymbol{A}^\top(\boldsymbol{A}\boldsymbol{\Phi}_{\mathsf{x}}\boldsymbol{A}^\top + \sigma_{\mathsf{n}}^2\boldsymbol{I}_M)^{-1}(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{m}_{\mathsf{x}}) \\ &= \left(\boldsymbol{I}_N - \mathbf{diag}\left(\frac{\sigma_{\mathrm{x},j}^2}{\sigma_{\mathrm{x},j}^2 - \sigma_{\mathrm{c},j}^2}\right) + \mathbf{diag}\left(\frac{\sigma_{\mathrm{x},j}^2}{\sigma_{\mathrm{x},j}^2 - \sigma_{\mathrm{c},j}^2}\right)\right)\boldsymbol{m}_{\mathsf{x}} \\ &\quad + \mathbf{diag}\left(\frac{\sigma_{\mathrm{x},j}^2}{\sigma_{\mathrm{x},j}^2 - \sigma_{\mathrm{c},j}^2}\right)\boldsymbol{\Phi}_{\mathsf{x}}\boldsymbol{A}^\top(\boldsymbol{A}\boldsymbol{\Phi}_{\mathsf{x}}\boldsymbol{A}^\top + \sigma_{\mathsf{n}}^2\boldsymbol{I}_M)^{-1}(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{m}_{\mathsf{x}}) \\ &= \mathbf{diag}\left(\frac{\sigma_{\mathrm{c},j}^2}{\sigma_{\mathrm{c},j}^2 - \sigma_{\mathrm{x},j}^2}\right)\boldsymbol{m}_{\mathsf{x}} + \mathbf{diag}\left(\frac{\sigma_{\mathrm{x},j}^2}{\sigma_{\mathrm{x},j}^2 - \sigma_{\mathrm{c},j}^2}\right)\boldsymbol{m}_{\mathrm{c}}\ , \end{aligned} \tag{4.57}$$

i.e., the components of the estimate compute to ($j \in \{1, \ldots, N\}$)

$$\begin{aligned} m_{\mathrm{c,u},j} &= \frac{\sigma_{\mathrm{x},j}^2}{\sigma_{\mathrm{x},j}^2 - \sigma_{\mathrm{c},j}^2} m_{\mathrm{c},j} - \frac{\sigma_{\mathrm{c},j}^2}{\sigma_{\mathrm{x},j}^2 - \sigma_{\mathrm{c},j}^2} m_{\mathsf{x},j} = \frac{\sigma_{\mathrm{x},j}^2\sigma_{\mathrm{c},j}^2}{\sigma_{\mathrm{x},j}^2 - \sigma_{\mathrm{c},j}^2}\left(\frac{m_{\mathrm{c},j}}{\sigma_{\mathrm{c},j}^2} - \frac{m_{\mathsf{x},j}}{\sigma_{\mathrm{x},j}^2}\right) \\ &= \sigma_{\mathrm{c,u},j}^2\left(\frac{m_{\mathrm{c},j}}{\sigma_{\mathrm{c},j}^2} - \frac{m_{\mathsf{x},j}}{\sigma_{\mathrm{x},j}^2}\right)\ , \end{aligned} \tag{4.58}$$

or slightly rephrased

$$m_{\mathrm{c,u},j}/\sigma_{\mathrm{c,u},j}^2 = m_{\mathrm{c},j}/\sigma_{\mathrm{c},j}^2 - m_{\mathsf{x},j}/\sigma_{\mathrm{x},j}^2\ , \qquad 1/\sigma_{\mathrm{c,u},j}^2 = 1/\sigma_{\mathrm{c},j}^2 - 1/\sigma_{\mathrm{x},j}^2\ . \tag{4.59}$$

Hence, there is a connection to the recovery algorithm with individual variances from Sec. 4.3.2, when making the identifications $m_{\mathrm{c,u},j} = \tilde{x}_{\mathrm{s},j}$, $\sigma_{\mathrm{c,u},j}^2 = \tilde{\sigma}_{\mathrm{s},j}^2$ and $m_{\mathsf{x},j} = \tilde{x}_{\mathrm{c},j}$, respectively, $\sigma_{\mathrm{x},j}^2 = \tilde{\sigma}_{\mathrm{c},j}^2$.

The resulting average unbiased variance of individual unbiasing computes with the definition of the *arithmetic mean* as $\mathrm{M_A}(x_j) = \frac{1}{N}\sum_{j=1}^N x_j$ to [Fis16]

$$\sigma_{\mathrm{c,u,i}}^2 = \frac{1}{N}\sum_{j=1}^N \sigma_{\mathrm{c,u},j}^2 = \mathrm{M_A}\Big(\sigma_{\mathrm{x},j}^2[\boldsymbol{W}]_{jj}\Big) - \mathrm{M_A}\Big(\sigma_{\mathrm{x},j}^2\Big)\ . \tag{4.60}$$

In case of the specification by a single (average) variance, i.e., $\boldsymbol{\Phi}_{\mathsf{x}} = \sigma_{\mathsf{x}}^2\boldsymbol{I}_N$, with *harmonic mean* $\mathrm{M_H}(x_j) = \frac{N}{\sum_{j=1}^N 1/x_j} = \frac{1}{\mathrm{M_A}(1/x_j)}$ this simplifies to

$$\sigma_{\mathrm{c,u,i}}^2 = \sigma_{\mathsf{x}}^2\left(\mathrm{M_A}([\boldsymbol{W}]_{jj}) - 1\right) = \sigma_{\mathsf{x}}^2\left(\mathrm{M_A}(1/[\boldsymbol{K}]_{jj}) - 1\right) = \sigma_{\mathsf{x}}^2\left(\frac{1}{\mathrm{M_H}([\boldsymbol{K}]_{jj})} - 1\right)\ . \tag{4.61}$$

Comparing this with the average variance (4.50) for average bias compensation

$$\sigma_{\mathrm{c,u}}^2 = \sigma_{\mathsf{x}}^2(w-1) = \sigma_{\mathsf{x}}^2\left(\frac{N}{\mathrm{trace}(\boldsymbol{K})} - 1\right) = \sigma_{\mathsf{x}}^2\left(\frac{1}{\mathrm{M_A}([\boldsymbol{K}]_{jj})} - 1\right)\ , \tag{4.62}$$

shows that $\sigma_{\mathrm{c,u}}^2 < \sigma_{\mathrm{c,u,i}}^2$, since $\mathrm{M_A}([\boldsymbol{K}]_{jj}) > \mathrm{M_H}([\boldsymbol{K}]_{jj})$ for varying $[\boldsymbol{K}]_{jj}$.

Noteworthy, in the optimization-based view, average variances and average unbiasing is inherently connected and so are individual variances and individual unbiasing. From the

estimation-theoretic perspective, there is no need in restricting to these combinations. A respective consideration of individual variances with average unbiasing is unlikely to yield good results, as the "loose" requirement of compensating only the average effect of the end-to-end cascade $\boldsymbol{K}$ cannot capture the variability in the variances $\sigma^2_{\mathsf{x},j}$. This case will therefore not be considered here. However, at first sight, there is no reason not to use individual unbiasing, when an average variance is present (case $\boldsymbol{\Phi}_{\mathsf{x}} = \sigma^2_{\mathsf{x}} \boldsymbol{I}_N$). Hence, algorithms with this combination will be composed below. Before that, the non-linear estimation and the respective unbiasing procedure are considered from an estimation-theoretic perspective.

4.4.2 Signal-Constrained Estimation

In this section, we recapitulate suitable bias compensation techniques for the scalar non-linear MMSE estimation[2] (2.19) similar to [Fis16, SF18, Spa19, FSG20, FS22], where the derivation is based on a geometric point of view as used in [Kay93, Fis02, For03]. Random variables allow a geometrical interpretation, since joint estimation can be interpreted as inner product of a vector space [Kay93, Sec. 12.4]. To understand this, recall that the observation model for this case is given by (cf. also (2.18))

$$\tilde{\mathsf{x}}_{\mathrm{s}} = \mathsf{x} + \mathsf{n}_{\mathrm{s}} \ , \quad \text{with} \quad \mathsf{x} \sim \mathsf{f}_{\mathsf{x}}(x) \ , \quad \mathsf{n}_{\mathrm{s}} \sim \mathcal{N}\Big(0, \, \sigma^2_{\mathsf{n}_{\mathrm{s}}}\Big) \ . \tag{4.63}$$

Note that the noise variance is denoted by $\sigma^2_{\mathsf{n}_{\mathrm{s}}}$ here, instead of $\tilde{\sigma}^2_{\mathrm{s}}$, as in (2.18). The denomination shall stress the connection to the observation noise n_{s}. For the algorithms, $\tilde{\sigma}^2_{\mathrm{s}}$ will be used again to point out the connection to $\tilde{x}_{\mathrm{s}}$.

In CS, the sparse prior $\mathbf{x}$ usually is assumed to have zero-mean, i.e., $m_{\mathsf{x}} = \mathrm{E}_{\mathsf{x}}\{\mathsf{x}\} = 0$. Furthermore, assuming that the noise / error n_{s} is *uncorrelated* to the signal, it is [Kay93]

$$\mathrm{E}_{\mathsf{x}\mathsf{n}_{\mathrm{s}}}\{\mathsf{x}\mathsf{n}_{\mathrm{s}}\} = \mathrm{E}_{\mathsf{x}}\{\mathsf{x}\}\mathrm{E}_{\mathsf{n}_{\mathrm{s}}}\{\mathsf{n}_{\mathrm{s}}\} = 0 \ , \tag{4.64}$$

where the first estimation is w.r.t. the joint pdf $\mathsf{f}_{\mathsf{x}\mathsf{n}_{\mathrm{s}}}(x, n_{\mathrm{s}})$. Interpreting the joint expectation $\mathrm{E}_{\mathsf{x}\mathsf{n}_{\mathrm{s}}}\{\mathsf{x}\mathsf{n}_{\mathrm{s}}\}$ as inner product, implies that the random variables x and n_{s} are *orthogonal*, when considered as vectors in a vector space, cf. [PP02, p. 211]; for a summary of the notion of vector spaces, the reader may be referred to Appendix A. The norm of the vector space is accordingly given by "joint" expectation of the random variable with itself, i.e., when considering a Euclidean vector space, the squared "length", i.e., the squared ℓ_2-norm of the vector representing observation noise n_{s} is given by

$$\mathrm{E}_{\mathsf{n}_{\mathrm{s}}}\Big\{\mathsf{n}^2_{\mathrm{s}}\Big\} = \sigma^2_{\mathsf{n}_{\mathrm{s}}} \ . \tag{4.65}$$

Respectively, the signal x has a squared length of $\sigma^2_{\mathsf{x}} = \mathrm{E}_{\mathsf{x}}\{\mathsf{x}^2\}$.

Let us get back to the MMSE criterion. There, the estimation error $\mathsf{e}_{\mathrm{s}} = \mathsf{x} - \mathsf{m}_{\mathrm{s}}$ between signal x and estimate m_{s} is considered. The criterion aims to minimize the squared version of this error given the observation $\tilde{x}_{\mathrm{s}}$, i.e.,

$$m_{\mathrm{s}} = \underset{m}{\operatorname{argmin}} \, \mathrm{E}_{\mathsf{x}}\Big\{(\mathsf{x} - m)^2 \mid \tilde{x}_{\mathrm{s}}\Big\} \ . \tag{4.66}$$

[2] For simplicity, the index j is omitted in this section.

The resulting conditional mean is justified by the fact that [Kay93, PP02]

$$\frac{\partial}{\partial m}\mathrm{E}_{\mathsf{x}}\Big\{(\mathsf{x}-m)^2 \mid \tilde{x}_{\mathrm{s}}\Big\} = -2(\mathrm{E}_{\mathsf{x}}\{\mathsf{x} \mid \tilde{x}_{\mathrm{s}}\} - m) \stackrel{!}{=} 0 \;, \tag{4.67}$$

requires $m = m_{\mathrm{s}} = \mathrm{E}_{\mathsf{x}}\{\mathsf{x} \mid \tilde{x}_{\mathrm{s}}\}$. The corresponding reliability of the estimate for the given observation is defined as [PP02, GWSV11]

$$\sigma_{\mathrm{s}}^2 = \mathrm{E}_{\mathsf{x}}\Big\{(\mathsf{x}-m_{\mathrm{s}})^2 \mid \tilde{x}_{\mathrm{s}}\Big\} = \mathrm{E}_{\mathsf{x}}\Big\{\mathsf{x}^2 \mid \tilde{x}_{\mathrm{s}}\Big\} - m_{\mathrm{s}}^2 \;, \tag{4.68}$$

cf. also (2.20). As both estimates are functions of $\tilde{x}_{\mathrm{s}}$, we may in the following use the notation $m_{\mathrm{s}}(\tilde{x}_{\mathrm{s}})$ and $\sigma_{\mathrm{s}}^2(\tilde{x}_{\mathrm{s}})$ to emphasize this dependency. The so-called (minimum) *mean-squared error* (MSE) [Kay93] of this strategy is obtained by integration over the statistic of the observation $\tilde{\mathsf{x}}_{\mathrm{s}}$, i.e., it defines the observation-independent reliability of the estimation, [Kay93, GWSV11]

$$\sigma^2 \stackrel{\text{def}}{=} \mathrm{E}_{\tilde{\mathsf{x}}_{\mathrm{s}}}\Big\{\sigma_{\mathrm{s}}^2(\tilde{\mathsf{x}}_{\mathrm{s}})\Big\} = \int \mathsf{f}_{\tilde{\mathsf{x}}_{\mathrm{s}}}(\tilde{x}_{\mathrm{s}})\sigma_{\mathrm{s}}^2(\tilde{x}_{\mathrm{s}})\,\mathrm{d}\tilde{x}_{\mathrm{s}} \;. \tag{4.69}$$

Calculating this integration within a recovery algorithm would lead to an unnecessary complexity overhead. Since the MSE is only a function of the variance $\sigma_{\mathsf{n}_{\mathrm{s}}}^2$ ($\tilde{\sigma}_{\mathrm{s}}^2$ within the algorithms), the computation can be done beforehand and stored in a look-up table (LUT).

The MSE σ^2 is depicted over the input $\sigma_{\mathsf{n}_{\mathrm{s}}}^2$ (a.k.a. $\tilde{\sigma}_{\mathrm{s}}^2$) in Fig. 4.5 for the distribution (2.4), which is called discrete ternary (DT) prior, for $N = 250$. Also, two bounds from [GWSV11] are given. The rougher one (dashed) states that the MSE σ^2 is always smaller than the input variance $\sigma_{\mathsf{n}_{\mathrm{s}}}^2$ and the variance of the prior σ_{x}^2. The second bound (dotted) is the MSE of a Gaussian random variable, which reads

$$\sigma^2 = (1/\sigma_{\mathsf{x}}^2 + 1/\sigma_{\mathsf{n}_{\mathrm{s}}}^2)^{-1} = \frac{\sigma_{\mathsf{x}}^2}{1+\sigma_{\mathsf{x}}^2/\sigma_{\mathsf{n}_{\mathrm{s}}}^2} = \frac{\sigma_{\mathsf{x}}^2\sigma_{\mathsf{n}_{\mathrm{s}}}^2}{\sigma_{\mathsf{n}_{\mathrm{s}}}^2+\sigma_{\mathsf{x}}^2} \;. \tag{4.70}$$

According to [GWSV11], the MSE of a Gaussian random variable is always larger (or equal) than the MSE of any other random variable with the same variance σ_{x}^2. Noteworthy, this feature is connected to the maximum-entropy property of Gaussian random variables. On the left side, the MSE of the DT prior is compared to the two bounds and the Bernoulli-Gaussian (BG) prior (2.5) with a sparsity of $s = 5$, i.e., $\sigma_{\mathsf{x}}^2 = s/N = 5/250 = 0.02$. One can see that the MSEs of both priors converge to the bound given by σ_{x}^2 for large values of the input variance $\sigma_{\mathsf{n}_{\mathrm{s}}}^2$. It is also visible that the DT prior has a tremendous decrease in the MSE for small input variances, when compared to the BG prior; only around $\sigma_{\mathsf{n}_{\mathrm{s}}}^2 = 10^{-1}$ the MSE of the DT prior is larger. The reason for the faster descent is the discrete nature of the prior, which simplifies the estimation in less noise. On the right side, the DT prior is compared for different sparsities that will be used in the simulations in Chapter 6. According to the bounds, the value of convergence towards large input variances changes, but the descents towards small input variances is similar. For more properties of the MSE, cf. [GSV05, GWSV11].

It can be shown that joint estimation of estimation error e_{s} and observation $\tilde{x}_{\mathrm{s}}$, in fact any function of $\tilde{x}_{\mathrm{s}}$, is zero, i.e., [Fis16, PP02]

$$\begin{aligned}\mathrm{E}_{\mathsf{e}_{\mathrm{s}}\tilde{\mathsf{x}}_{\mathrm{s}}}\{\mathsf{e}_{\mathrm{s}}f(\tilde{\mathsf{x}}_{\mathrm{s}})\} &= \mathrm{E}_{\mathsf{x}\tilde{\mathsf{x}}_{\mathrm{s}}}\{(\mathsf{x}-m_{\mathrm{s}}(\tilde{\mathsf{x}}_{\mathrm{s}}))f(\tilde{\mathsf{x}}_{\mathrm{s}})\} = \mathrm{E}_{\tilde{\mathsf{x}}_{\mathrm{s}}}\{f(\tilde{\mathsf{x}}_{\mathrm{s}})\mathrm{E}_{\mathsf{x}}\{\mathsf{x}-m_{\mathrm{s}}(\tilde{\mathsf{x}}_{\mathrm{s}}) \mid \tilde{\mathsf{x}}_{\mathrm{s}}\}\} \\ &= \mathrm{E}_{\tilde{\mathsf{x}}_{\mathrm{s}}}\{f(\tilde{\mathsf{x}}_{\mathrm{s}})(\mathrm{E}_{\mathsf{x}}\{\mathsf{x} \mid \tilde{\mathsf{x}}_{\mathrm{s}}\} - \mathrm{E}_{\mathsf{x}}\{m_{\mathrm{s}}(\tilde{\mathsf{x}}_{\mathrm{s}}) \mid \tilde{\mathsf{x}}_{\mathrm{s}}\})\} \\ &= \mathrm{E}_{\tilde{\mathsf{x}}_{\mathrm{s}}}\{f(\tilde{\mathsf{x}}_{\mathrm{s}})(m_{\mathrm{s}}(\tilde{\mathsf{x}}_{\mathrm{s}}) - m_{\mathrm{s}}(\tilde{\mathsf{x}}_{\mathrm{s}}))\} = 0 \;,\end{aligned} \tag{4.71}$$

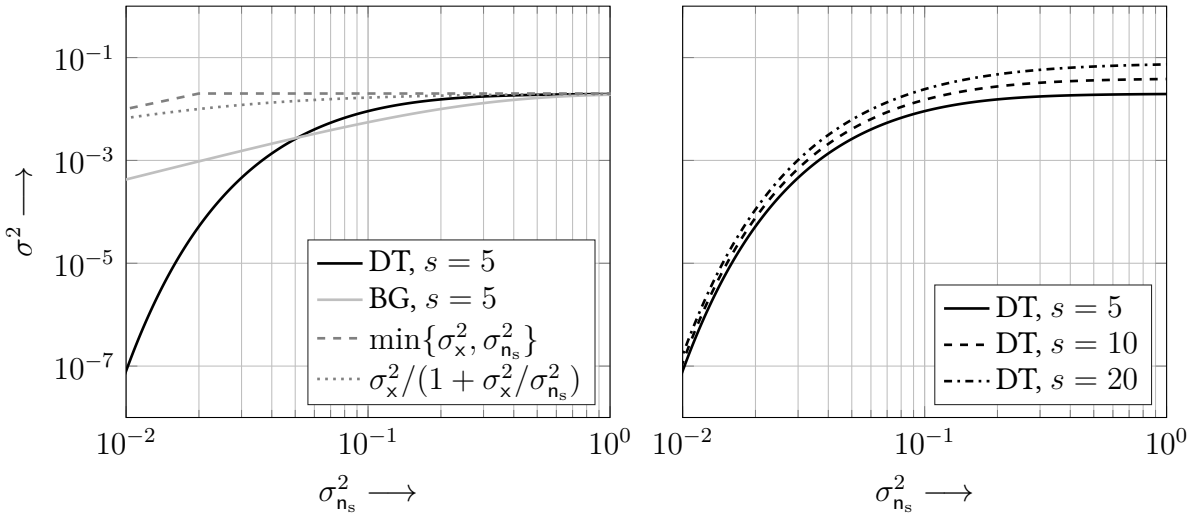


Figure 4.5: Comparison of the MSE for two different priors for $N = 250$. On the left, the priors are compared; on the right, the MSE of the DT prior (2.4) is compared for different sparsities. Note that $\sigma_{\mathsf{n_s}}^2$ is called $\tilde{\sigma}_{\mathrm{s}}^2$, when referring to the algorithmic procedures.

with $f(\tilde{x}_{\mathrm{s}})$ being an arbitrary function of $\tilde{x}_{\mathrm{s}}$. Since m_{s} is a function of $\tilde{x}_{\mathrm{s}}$, this means that the MMSE estimate is orthogonal to the resulting estimation error e_{s}. Naturally, also the observation $\tilde{\mathsf{x}}_{\mathrm{s}}$ when considered a random variable, is orthogonal to e_{s}.

Since there are right angles between different vectors, if the estimation is not linear, this requires a three-dimensional space to be displayed, because the estimate (considered as random variable) needs to lie on a Thales circle between x and $\sigma_{\mathsf{x}}^2/(\sigma_{\mathsf{x}}^2 + \sigma_{\mathsf{n_s}}^2)\tilde{\mathsf{x}}_{\mathrm{s}}$, which is the corresponding linear estimator and optimal if the signal x is Gaussian distributed, for a visualization see [For03]. The random variables are depicted as vectors in a vector space in Fig. 4.6, similarly to a figure in [FSG20]. The vector space is spanned by the direction of x, n_{s}, and a third axis, which we call v. The right angles are depicted as squares in the respective perspective. The vector for the estimate m_{s} is here constructed by the intersection of two tilted Thales circles, that span vector x and n_{s}, respectively, thereby ensuring the above derived right angles between the vectors. The displayed measures are squared distances, i.e., the variances of the corresponding random variables.

Since the estimation error e_{s} is not orthogonal to the signal x itself, i.e., a projection towards x does not yield zero, which means a part of the signal is accounted to the error. This may be interpreted as *bias*, for which two ways of compensation are introduced in the following sections.

Noteworthy, by use of total expectation, the expected value of the MMSE estimate is obtained as [Fis18]

$$\mathrm{E}_{\mathsf{m}_{\mathrm{s}}}\{\mathsf{m}_{\mathrm{s}}\} = \mathrm{E}_{\tilde{\mathsf{x}}_{\mathrm{s}}}\{\mathrm{E}_{\mathsf{x}}\{\mathsf{x} \mid \tilde{\mathsf{x}}_{\mathrm{s}}\}\} = \mathrm{E}_{\mathsf{x}}\{\mathsf{x}\} = 0 \,, \tag{4.72}$$

which means that on average an MMSE estimate is expectation-consistent and thereby unbiased, however, this notion of bias is not considered here.

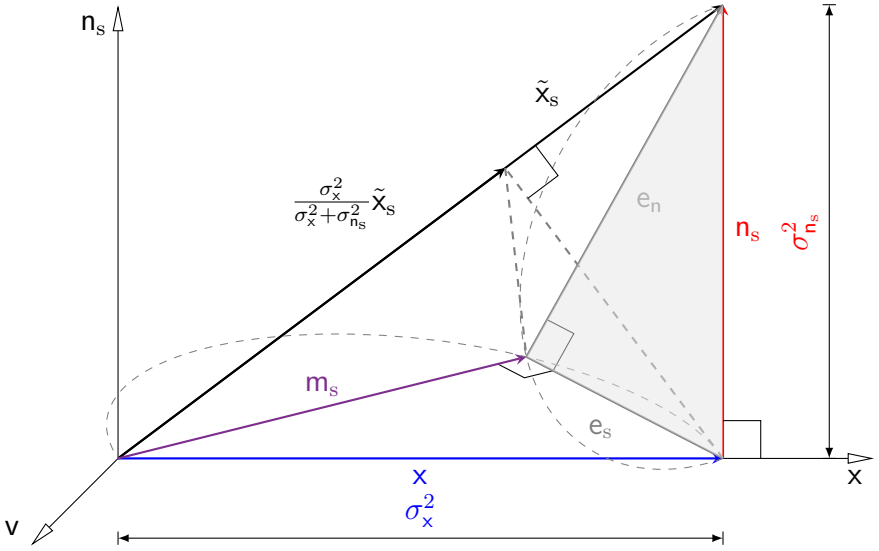


Figure 4.6: Visualization of the random variables involved in the estimation as vectors (cf. [Fis16]).

4.4.2.1 Individual Unbiasing

The precise handling of the bias compensation procedure is obtained by scaling realizations of the random variables, which are involved in the estimation, adequately. In this work, only what is called *noise-oriented* unbiasing in [FSG20] and [Spa19] is considered, since it has better properties for the recovery in CS, for more details, see [Spa19]. This means, that instead of scaling directly the estimate m_s, the error between observation $\tilde{x}_s$ and estimate m_s is regarded, which needs to be decomposed into a part of the actual observation noise n_s and an uncorrelated distortion d_n. For this, one needs to set up the model [Fis16, Ban13]

$$e_n = \tilde{x}_s - m_s = k_n n_s + d_n \ , \tag{4.73}$$

where k_n is the factor that is necessary to scale the observation noise such that it is uncorrelated to the distortion, i.e., $\mathrm{E}_{n_s d_n}\{n_s d_n\} = 0$. The error e_n is already depicted in Fig. 4.6 as vector. The scaling factor can be obtained by considering similar triangles in the shaded area of Fig. 4.6. The shaded triangle of Fig. 4.6 is drawn again in Fig. 4.7, such that the triangle lies in the drawing plane. The corner points of the triangle are the endpoints of x, m_s, and $\tilde{x}_s$, the edges are given by n_s, e_s, and e_n. The estimation error e_s has by definition a squared length equal to the MSE, i.e., $\mathrm{E}_{e_s}\{e_s^2\} = \sigma^2$. With the Pythagorean theorem, one can infer the respective squared length of e_n. The relations between sides of the two right triangles yields

$$\frac{\mathrm{E}_{n_s}\{(k_n n_s)^2\}}{\mathrm{E}_{e_n}\{e_n^2\}} = \frac{k_n^2 \sigma_{n_s}^2}{\sigma_{n_s}^2 - \sigma^2} \overset{!}{=} \frac{\mathrm{E}_{e_n}\{e_n^2\}}{\mathrm{E}_{n_s}\{n_s^2\}} = \frac{\sigma_{n_s}^2 - \sigma^2}{\sigma_{n_s}^2} \ , \tag{4.74}$$

and finally

$$k_n = \frac{\sigma_{n_s}^2 - \sigma^2}{\sigma_{n_s}^2} \ . \tag{4.75}$$

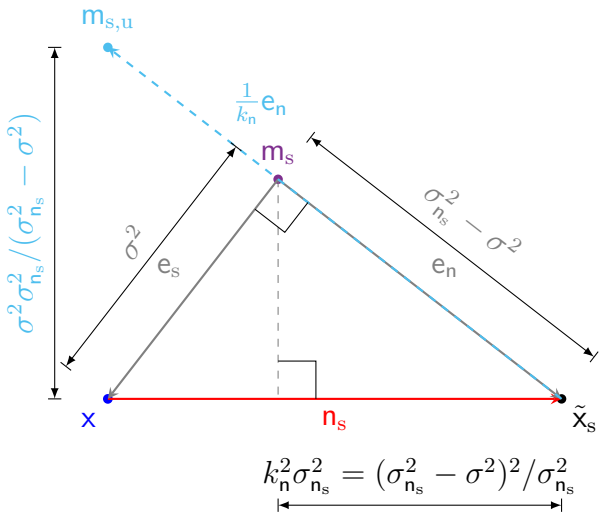


Figure 4.7: Visualization of the random variables involved in the noise-oriented bias compensation.

The scaling factor tells us, how much the observation noise n_s is scaled in order to obtain the difference e_n. Returning to the estimates, an unbiased version $m_{s,u}$ of estimate m_s is obtained by inversely applying the scaling to the difference between observation and estimate and subtracting it from the observation, i.e.,

$$m_{s,u} = \tilde{x}_s + \frac{1}{k_n}(\tilde{x}_s - m_s) = \tilde{x}_s + \frac{\sigma^2_{n_s}}{\sigma^2_{n_s} - \sigma^2}(m_s - \tilde{x}_s) \; . \tag{4.76}$$

The corresponding unbiased variance is obtained as

$$\sigma^2_{s,u} = \sigma^2_s + \left(\frac{\sigma^2}{\sigma^2 - \sigma^2_{n_s}}\right)^2 (m_s - \tilde{x}_s)^2 \; . \tag{4.77}$$

This is called the *individual* unbiasing case, as it is exactly tailored for one observation $\tilde{x}_s$ with respective noise variance $\sigma^2_{n_s}$. The case, where the reliability of an entire vector is specified in one variance is discussed in the next section.

4.4.2.2 Average Unbiasing

In the following, the connection between the previously explained *noise-oriented* unbiasing and the extrinsic calculations performed in VAMP is shown, which has similarly been stated in [SF18].

First, consider that since VAMP uses one variance $\sigma^2_{n_s}$ to adjust the non-linear estimators for all components of the signal vector $\boldsymbol{x}$, the corresponding MSE σ^2 is equal for all components because it is a function of the input variance only. Furthermore, for a high-dimensional signal, the empirical average over the conditional variances $\sigma^2_{s,j}$ suitably approximates the MSE, since the averaging may be interpreted as Monte-Carlo integration with many samples / observations $\tilde{x}_{s,j}$, i.e., it is

$$\sigma^2 = \lim_{N\to\infty} \frac{1}{N} \sum_{j=1}^{N} \sigma^2_{s,j}(\tilde{x}_{s,j}) \; . \tag{4.78}$$

With this in mind, (4.76) can be reformulated to obtain

$$\begin{aligned} m_{\mathrm{s,u}} &= \tilde{x}_{\mathrm{s}} + \frac{\sigma_{\mathsf{n}_{\mathrm{s}}}^2}{\sigma_{\mathsf{n}_{\mathrm{s}}}^2 - \sigma^2}(m_{\mathrm{s}} - \tilde{x}_{\mathrm{s}}) = \tilde{x}_{\mathrm{s}} + \frac{\sigma_{\mathsf{n}_{\mathrm{s}}}^2 \sigma^2}{\sigma_{\mathsf{n}_{\mathrm{s}}}^2 - \sigma^2}(m_{\mathrm{s}}/\sigma^2 - \tilde{x}_{\mathrm{s}}/\sigma^2) \\ &= \left(\frac{1}{\sigma^2} - \frac{1}{\sigma_{\mathsf{n}_{\mathrm{s}}}^2}\right)^{-1} \left(m_{\mathrm{s}}/\sigma^2 + \tilde{x}_{\mathrm{s}} \left(\frac{\sigma_{\mathsf{n}_{\mathrm{s}}}^2 - \sigma^2}{\sigma_{\mathsf{n}_{\mathrm{s}}}^2 \sigma^2} - \frac{1}{\sigma^2}\right)\right) \\ &= (1/\sigma^2 - 1/\sigma_{\mathsf{n}_{\mathrm{s}}}^2)^{-1}(m_{\mathrm{s}}/\sigma^2 - \tilde{x}_{\mathrm{s}}/\sigma_{\mathsf{n}_{\mathrm{s}}}^2) \,, \end{aligned} \tag{4.79}$$

which already reminds us of the update in VAMP without indices. Note that in VAMP $\sigma_{\mathsf{n}_{\mathrm{s}}}^2$ is denoted $\tilde{\sigma}_{\mathrm{s}}^2$ and as already discussed, σ_{s}^2 takes on the role of σ^2.

Last but not least, the unbiased MSE is stated, which is obtained as the observation-independent version of $\sigma_{\mathrm{s,u}}^2$, i.e., with (4.68), $\mathrm{E}_{\tilde{\mathsf{x}}_{\mathrm{s}}}\{\tilde{\mathsf{x}}_{\mathrm{s}}^2\} = \sigma_{\mathsf{x}}^2 + \sigma_{\mathsf{n}_{\mathrm{s}}}^2$,

$$\mathrm{E}_{\tilde{\mathsf{x}}_{\mathrm{s}}}\{m_{\mathrm{s}}(\tilde{\mathsf{x}}_{\mathrm{s}})\tilde{\mathsf{x}}_{\mathrm{s}}\} = \mathrm{E}_{\tilde{\mathsf{x}}_{\mathrm{s}}}\{(\mathsf{x} - \mathsf{e}_{\mathrm{s}})\tilde{\mathsf{x}}_{\mathrm{s}}\} = \mathrm{E}_{\tilde{\mathsf{x}}_{\mathrm{s}}}\{\mathsf{x}\tilde{\mathsf{x}}_{\mathrm{s}}\} = \mathrm{E}_{\tilde{\mathsf{x}}_{\mathrm{s}}}\{\mathsf{x}(\mathsf{x} + \mathsf{n}_{\mathrm{s}})\} = \sigma_{\mathsf{x}}^2 \,, \tag{4.80}$$

$$\mathrm{E}_{\tilde{\mathsf{x}}_{\mathrm{s}}}\left\{m_{\mathrm{s}}(\tilde{\mathsf{x}}_{\mathrm{s}})^2\right\} = \mathrm{E}_{\tilde{\mathsf{x}}_{\mathrm{s}}}\left\{\mathrm{E}_{\mathsf{x}}\left\{\mathsf{x}^2 \mid \tilde{\mathsf{x}}_{\mathrm{s}}\right\}\right\} - \mathrm{E}_{\tilde{\mathsf{x}}_{\mathrm{s}}}\left\{\sigma_{\mathrm{s}}^2(\tilde{\mathsf{x}}_{\mathrm{s}})\right\} = \mathrm{E}_{\mathsf{x}}\left\{\mathsf{x}^2\right\} - \sigma^2 = \sigma_{\mathsf{x}}^2 - \sigma^2 \,, \tag{4.81}$$

it is

$$\begin{aligned} \sigma_{\mathrm{u}}^2 &= \mathrm{E}_{\tilde{\mathsf{x}}_{\mathrm{s}}}\left\{\sigma_{\mathrm{s,u}}^2(\tilde{\mathsf{x}}_{\mathrm{s}})\right\} = \mathrm{E}_{\tilde{\mathsf{x}}_{\mathrm{s}}}\left\{\sigma_{\mathrm{s}}^2(\tilde{\mathsf{x}}_{\mathrm{s}})\right\} + \left(\frac{\sigma^2}{\sigma^2 - \sigma_{\mathsf{n}_{\mathrm{s}}}^2}\right)^2 \mathrm{E}_{\tilde{\mathsf{x}}_{\mathrm{s}}}\left\{(m_{\mathrm{s}}(\tilde{\mathsf{x}}_{\mathrm{s}}) - \tilde{\mathsf{x}}_{\mathrm{s}})^2\right\} \\ &= \sigma^2 + \left(\frac{\sigma^2}{\sigma^2 - \sigma_{\mathsf{n}_{\mathrm{s}}}^2}\right)^2 \left(\sigma_{\mathsf{x}}^2 - \sigma^2 - 2\sigma_{\mathsf{x}}^2 + \sigma_{\mathsf{x}}^2 + \sigma_{\mathsf{n}_{\mathrm{s}}}^2\right) = \sigma^2 + \frac{(\sigma^2)^2(\sigma_{\mathsf{n}_{\mathrm{s}}}^2 - \sigma^2)}{(\sigma^2 - \sigma_{\mathsf{n}_{\mathrm{s}}}^2)^2} \\ &= \frac{\sigma^2(\sigma_{\mathsf{n}_{\mathrm{s}}}^2 - \sigma^2) + (\sigma^2)^2}{\sigma_{\mathsf{n}_{\mathrm{s}}}^2 - \sigma^2} = \frac{\sigma^2 \sigma_{\mathsf{n}_{\mathrm{s}}}^2}{\sigma_{\mathsf{n}_{\mathrm{s}}}^2 - \sigma^2} = \left(\frac{1}{\sigma^2} - \frac{1}{\sigma_{\mathsf{n}_{\mathrm{s}}}^2}\right)^{-1} \,, \end{aligned} \tag{4.82}$$

which justifies the use of the respective update for the variance as well. This can also be derived graphically, the resulting squared length is shown in Fig. 4.7. Noteworthy, the averaging of variances performed in VAMP approximates MSEs[3], which are the parameters to use in bias compensation when deriving it from an estimation-theoretic perspective. Therefore, the procedure in VAMP is valid especially for high-dimensional signals $\boldsymbol{x}$.

All in all, this shows that the extrinsic computation derived for VAMP is also justified from an estimation-theoretic perspective.

4.5 Turbo-Type Algorithms—Estimation-Theoretic Adaption

The previous sections have opened different possibilities of treating the MMSE estimates to prepare them as input for the other estimator. This insight will be used in the following to compose algorithms by replacing the strategies at respective steps. At first, an algorithm with individual variances is considered.

[3] In case of linear estimation conditional variance and MSE are the same, so $\sigma_{\mathsf{n}_{\mathrm{s}}}^2$ a.k.a. $\tilde{\sigma}_{\mathrm{s}}^2$ is indeed an MSE.

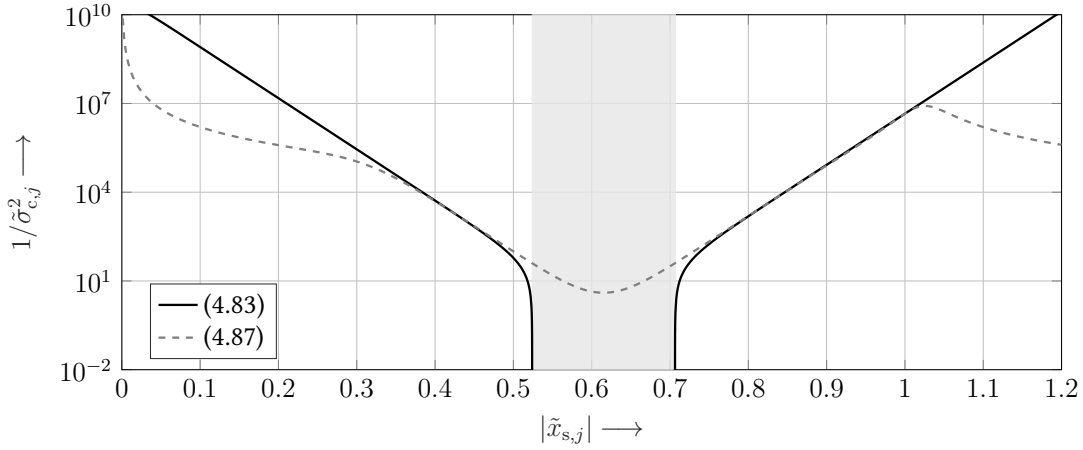


Figure 4.8: Inverse of the unbiased variance $\tilde{\sigma}^2_{\mathrm{c},j}$ for the two strategies: optimization-based approach (4.83) and estimation-theoretic bias compensation (4.87), for $N = 250$, $s = 5$, $10\log_{10}(1/\tilde{\sigma}^2_{\mathrm{s},j}) = 16\,\mathrm{dB}$.

4.5.1 Individual Variances and Estimation-Theoretic Bias Compensation

When comparing the individual unbiasing procedure in Sec. 4.4.2.1 with the extrinsic calculation in the algorithm of Sec. 4.3.2, one can see certain differences. For example, although the update for the mean estimates shares the same form, the initial algorithm does not make use of the MSE itself, but instead works with conditional variances. In [FSG20] this treatment of variances was shown to be disadvantageous as negative variances may result frequently in the algorithm, when computing

$$\frac{1}{\tilde{\sigma}^2_{\mathrm{c},j}} = \frac{1}{\sigma^2_{\mathrm{s},j}} - \frac{1}{\tilde{\sigma}^2_{\mathrm{s},j}}\,, \quad j \in \{1, \ldots, N\}\,. \tag{4.83}$$

Since negative variances have no useful meaning, this has to be prevented by, e.g., clipping the result to a positive interval [RSF19]. However, this affects the processing of the algorithm and ultimately deteriorates the performance.

A way to solve this problem is naturally given by the estimation-theoretic bias compensation view in Sec. 4.4.2.1. The corresponding algorithm is obtained by replacing the updates after the signal-constrained estimation with the derived Eqs. (4.76) and (4.77), which is stated below in Eqs. (4.86) and (4.87) in the notation of the algorithms again. The two computations of the unbiased variance (4.83) and (4.87) are compared in Fig. 4.8 for parameters that are used in the simulations in Chapter 6. The procedure with the subtraction leads to negative values in the gray area, while the estimation-theoretic unbiasing stays positive for all possible values of $\tilde{x}_{\mathrm{s},j}$ and $\tilde{\sigma}^2_{\mathrm{s},j}$.

The resulting algorithm was introduced in [FSG20] first, a similar version is called euIMS in [Spa19], here it is called *VAMPire*, for VAMP with individual reliabilities enhanced.

The main steps in the algorithm can be summarized as follows (in estimation-theoretic notation with mean and variances). Starting with a suitable initialization for the input variables of the channel-constrained estimation, e.g., $\tilde{\boldsymbol{x}}_{\mathrm{c}} = \boldsymbol{0}$ and $\tilde{\boldsymbol{\Phi}}_{\mathrm{c}} = s/N \cdot \boldsymbol{I}_N$, the unbiased

conditional mean and variance can directly be computed via

$$\tilde{\boldsymbol{x}}_{\mathrm{s}} = \tilde{\boldsymbol{x}}_{\mathrm{c}} + \boldsymbol{W}\tilde{\boldsymbol{\Phi}}_{\mathrm{c}}\boldsymbol{A}^{\top}(\boldsymbol{A}\tilde{\boldsymbol{\Phi}}_{\mathrm{c}}\boldsymbol{A}^{\top} + \sigma_{\mathsf{n}}^2\boldsymbol{I}_N)^{-1}(\boldsymbol{y} - \boldsymbol{A}\tilde{\boldsymbol{x}}_{\mathrm{c}}) \;, \tag{4.84}$$

$$\tilde{\sigma}_{\mathrm{s},j}^2 = \tilde{\sigma}_{\mathrm{c},j}^2([\boldsymbol{W}]_{jj} - 1) \;, \tag{4.85}$$

where the diagonal entries of the compensation matrix read $[\boldsymbol{W}]_{jj} = 1/[\tilde{\boldsymbol{\Phi}}_{\mathrm{c}}\boldsymbol{A}^{\top}(\boldsymbol{A}\tilde{\boldsymbol{\Phi}}_{\mathrm{c}}\boldsymbol{A}^{\top} + \sigma_{\mathsf{n}}^2\boldsymbol{I}_N)^{-1}\boldsymbol{A}]_{jj}$. The results are used to compute signal-constrained conditional mean $m_{\mathrm{s},j}$ from (2.19) and conditional variance $\sigma_{\mathrm{s},j}^2$ from (2.20) for $j \in \{1, \ldots, N\}$.

For the subsequent bias compensation, the procedure derived in Sec. 4.4.2.1 is used, i.e., one needs to compute

$$\tilde{x}_{\mathrm{c},j} = \tilde{x}_{\mathrm{s},j} + \frac{\tilde{\sigma}_{\mathrm{s},j}^2}{\tilde{\sigma}_{\mathrm{s},j}^2 - \sigma_j^2}(m_{\mathrm{s},j} - \tilde{x}_{\mathrm{s},j}) \;, \tag{4.86}$$

$$\tilde{\sigma}_{\mathrm{c},j}^2 = \sigma_{\mathrm{s},j}^2 + \left(\frac{\sigma_j^2}{\sigma_j^2 - \tilde{\sigma}_{\mathrm{s},j}^2}\right)^2 (m_{\mathrm{s},j} - \tilde{x}_{\mathrm{s},j})^2 \;. \tag{4.87}$$

The MSEs σ_j^2 are a direct function of $\tilde{\sigma}_{\mathrm{c},j}^2$ and can therefore be tabulated to reduce the complexity [FSG20]. In the next iteration, $\tilde{\boldsymbol{x}}_{\mathrm{s}} = [\tilde{x}_{\mathrm{s},1}, \ldots, \tilde{x}_{\mathrm{s},N}]^{\top}$ and $\tilde{\boldsymbol{\Phi}}_{\mathrm{c}} = \mathbf{diag}(\tilde{\sigma}_{\mathrm{c},j}^2)$ are used as input variables for the channel-constrained estimation and the computations are repeated until a suitable stopping criterion is met. The Algorithm is also stated in Appendix E.7.

It has to be noted that the computational complexity of this algorithm ranges in[4] $\mathcal{O}(M \cdot N^2)$, whereas VAMP is in $\mathcal{O}(MN)$, for the derivations see [Spa19]. The reason for the increased complexity of VAMPire is that in the channel-constrained estimation, a $M \times M$ matrix has to be inverted. For the VAMP case, the inverse can be simplified to the inversion of a diagonal matrix, i.e., scalar inversions are performed, by utilizing the singular value decomposition (SVD) of the sensing matrix [RSF19, Spa19]. Switching from scaled identity matrix $\tilde{\boldsymbol{\Phi}}_{\mathrm{c}} = \tilde{\sigma}_{\mathrm{c}}^2\boldsymbol{I}_N$ to a diagonal version $\tilde{\boldsymbol{\Phi}}_{\mathrm{c}} = \mathbf{diag}(\tilde{\sigma}_{\mathrm{c},j}^2)$, i.e., individual variances, the decomposition cannot yield a diagonal inverse, so the complexity reduction is not possible. This is of course a huge drawback and means that the gained insight into the individual reliabilities of the linear estimate is relative costly.

4.5.2 VAMP with Individually Unbiased LMMSE Estimate

It has been shown in Sec. 4.4.1.1 that the usual version of the VAMP algorithm inherently performs average bias compensation after the linear estimation. The individual bias compensation strategy has so far only been used, when individual variances were present. With individual variances in the linear estimator, this leads to a huge increase in complexity, since the computation of the inverse can not be facilitates by the SVD trick, as suggested in [RSF19]. However, there is no reason not to use individual unbiasing in a setting with average variances. As will be shown in numerical simulations in Chapter 6, individual unbiasing is particularly useful if the columns of the sensing matrix carry different energies, i.e., have different (squared) ℓ_2-norms. In the following, we therefore compose algorithms that combine the use of an average

[4] By $f(x) = \mathcal{O}(g(x))$ the Landau notation is denoted, which bounds the growth of $f(x)$ for $x \to \infty$ by the function $g(x)$ in the sense that $|f(x)| \leq \text{const.} \cdot g(x)$ for $x \geq x_0$ from some point $x_0 \in \mathbb{R}$ on [Lan09].

variance in the linear estimation with individual bias compensation after the linear estimation. The respective scheme has been published in [SF22].

The channel-constrained estimation is for this case given as

$$\tilde{\boldsymbol{x}}_{\mathrm{s}} = \tilde{\boldsymbol{x}}_{\mathrm{c}} + \boldsymbol{W}\boldsymbol{A}^{\top}(\boldsymbol{A}\boldsymbol{A}^{\top} + \frac{\sigma_{\mathsf{n}}^2}{\tilde{\sigma}_{\mathrm{c}}^2}\boldsymbol{I}_M)^{-1}(\boldsymbol{y} - \boldsymbol{A}\tilde{\boldsymbol{x}}_{\mathrm{c}}) \ , \tag{4.88}$$

where $[\boldsymbol{W}]_{jj} = 1/[\boldsymbol{A}^{\top}(\boldsymbol{A}\boldsymbol{A}^{\top} + \frac{\sigma_{\mathsf{n}}^2}{\tilde{\sigma}_{\mathrm{c}}^2}\boldsymbol{I}_M)^{-1}\boldsymbol{A}]_{jj}$. As reliability for the signal-constrained estimation in (2.19) and (2.20), one can either keep the individual unbiased variances

$$\tilde{\sigma}_{\mathrm{s},j}^2 = \sigma_{\mathsf{x}}^2\left([\boldsymbol{W}]_{jj} - 1\right) \ , \tag{4.89}$$

or average them to obtain

$$\tilde{\sigma}_{\mathrm{s}}^2 = \sigma_{\mathsf{x}}^2\left(\mathrm{M}_{\mathrm{A}}([\boldsymbol{W}]_{jj}) - 1\right) \ . \tag{4.90}$$

This leads to two different variants of the VAMP algorithm, which is denoted by *VAMPii* for individual unbiasing and individual (unbiased) variances and *VAMPia* for individual unbiasing and average (unbiased) variances.

For the unbiasing after the signal-constrained estimation, the procedure of average unbiasing is utilized, since the resulting input variance for the channel-constrained estimation needs to be an average one. Therefore, it is in both cases

$$\tilde{\boldsymbol{x}}_{\mathrm{c}} = \tilde{\sigma}_{\mathrm{c}}^2(\boldsymbol{m}_{\mathrm{s}}/\sigma_{\mathrm{s}}^2 - \tilde{\boldsymbol{x}}_{\mathrm{s}}/\tilde{\sigma}_{\mathrm{s}}^2) \ , \quad \tilde{\sigma}_{\mathrm{c}}^2 = (1/\sigma_{\mathrm{s}}^2 - 1/\tilde{\sigma}_{\mathrm{s}}^2)^{-1} \ , \tag{4.91}$$

where $\sigma_{\mathrm{s}}^2 = \mathrm{M}_{\mathrm{A}}(\sigma_{\mathrm{s},j}^2)$ has to be averaged for both algorithms and $\tilde{\sigma}_{\mathrm{s}}^2 = \mathrm{M}_{\mathrm{A}}(\tilde{\sigma}_{\mathrm{s},j}^2)$ additionally for VAMPii.

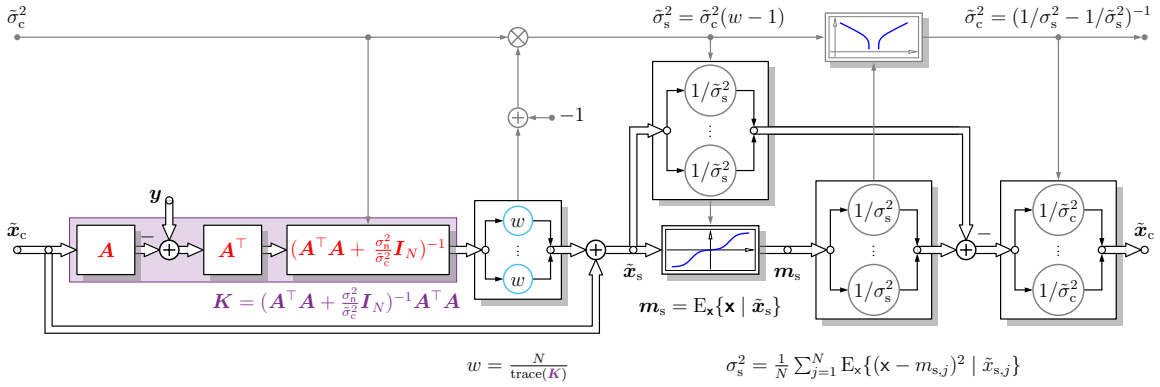
The complete algorithms are stated in Appendix E.8.

On the next double page, the Turbo algorithms are summarized in the form of block diagrams.

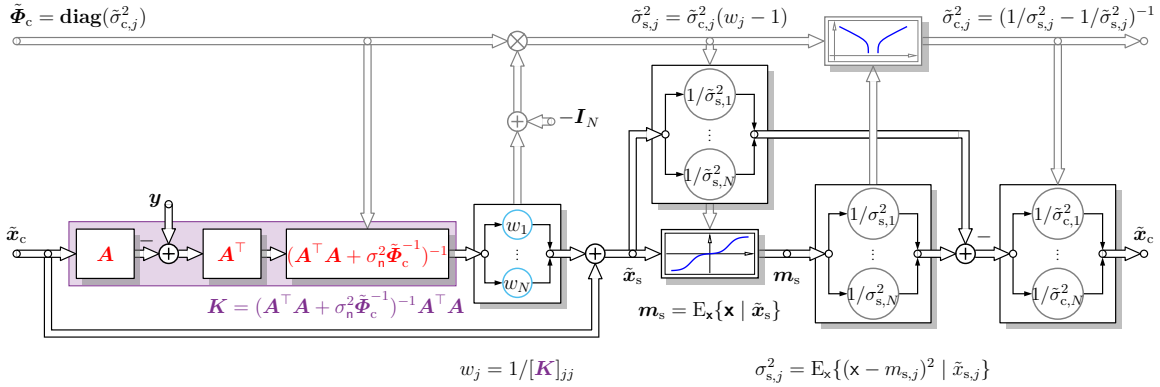
4.6 Block Diagrams of the Turbo Algorithms

In the following, the Turbo-type algorithms are summarized in the form of block diagrams. Here, the block for the signal-constrained estimation accepts the mean $\tilde{\boldsymbol{x}}_{\mathrm{s}}$ and corresponding variance(s) as input parameters and returns the conditional mean $\boldsymbol{m}_{\mathrm{s}}$. The depiction shows exemplary the signal-constrained estimation for the DT prior over input values $\tilde{x}_{\mathrm{s},j}$ for a given variance.

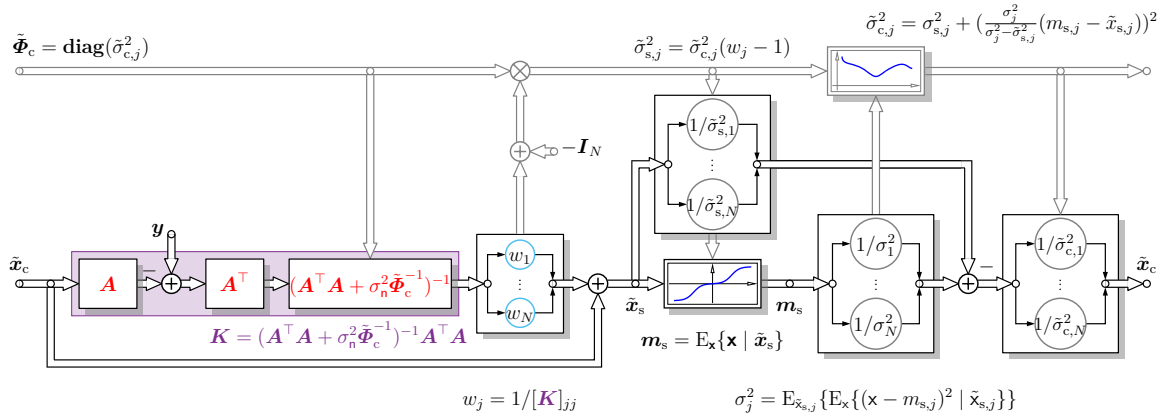
4.6.1 VAMP



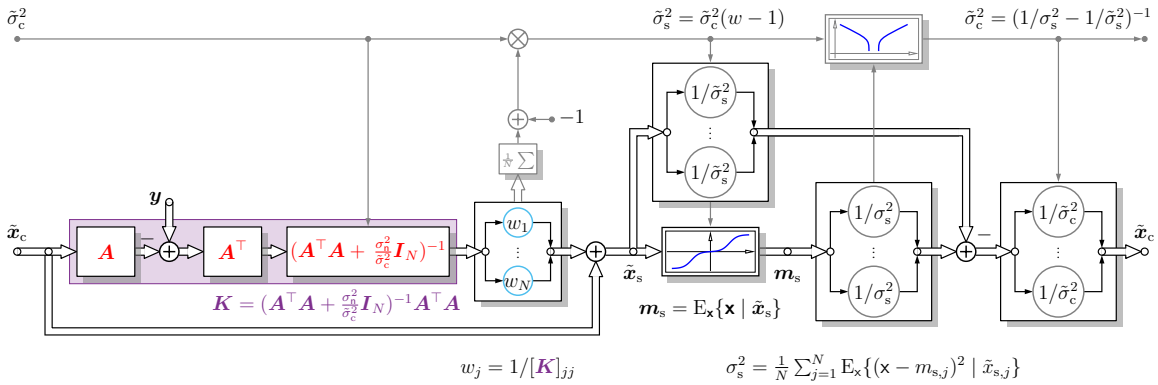
4.6.2 VAMPind



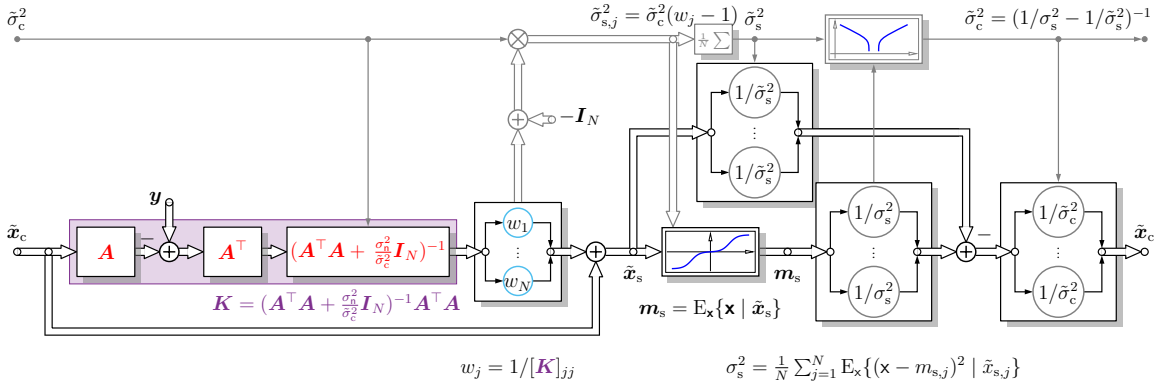
4.6.3 VAMPire



4.6.4 VAMPia



4.6.5 VAMPii



5. Sequential Inference

In the last two chapters, we have seen the two extreme cases of working with the factor graph for CS. Chapter 3 factorized the conditional pdf (2.7) as much as possible to yield the most simplest estimators, which led to a relative high complexity because of the huge number ($\mathcal{O}(M \cdot N)$ due to the number of connecting edges) of messages that needed to be broadcasted.

In Chapter 4, the opposite approach was considered by factorizing to a minimum number of components, i.e., separating only what really needed to be apart. The result were two major components, between which the recovery algorithms iterated to achieve a suitable solution.

The chapter at hand, describes cases that lie in between these two extreme approaches. The idea is to factorize the posterior $\mathsf{f}_{\mathsf{x}|\mathsf{y}}(\boldsymbol{x})$ more than in the Turbo case (Chapter 4) but not as much as in the MP case (Chapter 3). In contrast to the Turbo procedure, this leads to the possibility of (different) schedules, as in the MP case, but without the corresponding high computational complexity in terms of messages to be propagated. The processing will result in a *sequential* version of the (parallel) Turbo case.

Subsequently two different approaches will be considered that follow the remaining factorization possibilities of Fig. 2.3, starting with separated variables nodes, but combined channel factors, i.e., Fig. 2.3(c) and finishing off with the final part Fig. 2.3(b), which depicts separated channel factor nodes and a combined variable node.

5.1 Sequential Processing of Variables

When considering the processing in the Turbo case, one may realize that the computation of the signal-constraint estimates (2.19) and (2.20), although being processed in parallel, not necessarily need to be computed together, since the variables are considered i.i.d., cf. (2.18). A natural idea for sequentializing the Turbo process is, therefore, to consider only one component x_j of the signal vector in the signal-constrained estimation to find the impact of the single variable on the channel-constrained estimation. In the next step, another component is considered, eventually leading to a sequential processing of the variables $x_1, \ldots, x_N$.

Sequential processing of variables can for the CS literature be found, e.g., in [MKTZ15],

where the AMP algorithm is sequentialized. In contrast, in this chapter sequential versions of VAMPind, i.e., VAMP with individual variances, and VAMPire are considered. Other examples for sequential (variable) processing in CS are [SN08], where the focus lies on the design of the sensing matrix $\boldsymbol{A}$—in contrast, we consider $\boldsymbol{A}$ fixed and focus on the algorithmic part— and [BMPP20], which considers a Bernoulli-Gaussian prior whose sparsity parameter is learned throughout the algorithm. A similar algorithm to the one in [BMPP20] is also described in [OW05, Appendix D] for a general inference problem with binary prior. The respective algorithm for CS with arbitrary prior is explained in Sec. 5.1.4 below. The algorithm was improved in [SF21] by an estimation-theoretic consideration of the bias compensation; here, this version of the algorithm is discussed in Sec. 5.1.5.

Noteworthy, the mentioned algorithms in [OW05, BMPP20, SN08] are all based on the *expectation propagation* (EP) [Min01] framework, which considers (in its initial version) sequential updates in a general inference problem but is often interpreted in a parallel schedule, last but not least also for VAMP [RSF19].

Recent work on sequential processing is also given in [ÇLO22], which analyzes a sequential version of VAMP with random schedule for a Gaussian model with latent parameter.

Another study on sequential CS, which is further away than the previously mentioned references, is [LCJ15], where dynamically changing signals with spatial and temporal correlation are processed sequentially in a wireless sensor network (WSN).

The derivation given here, follows the optimization-based approach that was also considered in the previous chapters.

5.1.1 Structure of the Factor Graph

The idea for the sequential processing of the variables $x_1, \ldots, x_N$ that is pursued in this part of the chapter is motivated by the fact that the components of the signal vector $\mathbf{x}$ are i.i.d.. Consequently, the signal factors $\mathsf{f}_{\mathsf{x}}(x_j)$ are considered individually in the factorization, which yields individual processing of the signal-constrained estimates. The channel-constrained part, on the other hand, is kept together, thereby enabling the possibility to propagate effects of a single signal-constrained estimation to the other signal components in one step. The factorization that is considered here is therefore given by

$$\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x}) = \text{const.} \cdot \mathsf{f}_{\mathbf{y}|\mathbf{x}}(\boldsymbol{x}) \cdot \prod_{j=1}^{N} \mathsf{f}_{\mathsf{x}}(x_j) \; . \tag{5.1}$$

The corresponding factor graph is the one shown in Fig. 2.3(c), which is plotted in Fig. 5.1 for convenience again.

5.1.1.1 Factorization of the Substitute Distribution

Following the factorization in (5.1), $\mathsf{q}_{\mathrm{c}}(\boldsymbol{x})$ approximates $\mathsf{f}_{\mathbf{y}|\mathbf{x}}(\boldsymbol{x})$ as in (4.4) and $\mathsf{q}_{\mathrm{s},j}(x_j)$ approximates $\mathsf{f}_{\mathsf{x}}(x_j)$ ($j \in \{1, \ldots, N\}$) as in (3.3), where it is assumed that the deviation in the

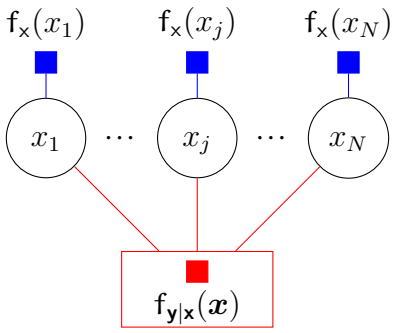


Figure 5.1: Factor graph for sequential processing of variable nodes.

approximation for the channel-constrained factor splits according to

$$c(\boldsymbol{x}) = \prod_{j=1}^{N} c_j(x_j) \; . \tag{5.2}$$

The individual marginals $\mathsf{q}_{\mathrm{v},j}(x_j)$ are then given by

$$\mathsf{q}_{\mathrm{v},j}(x_j) = \mathrm{const.} \cdot s_j(x_j) \cdot c_j(x_j) \; . \tag{5.3}$$

The individual representation is used (instead of an entire representative $\mathsf{q}_{\mathrm{v}}(\boldsymbol{x})$) because the variable nodes are considered separately in the factor graph. The resulting factorization of the substitute distribution reads

$$\mathsf{q}(\boldsymbol{x}) = \frac{\mathsf{q}_{\mathrm{c}}(\boldsymbol{x}) \cdot \prod_{j=1}^{N} \mathsf{q}_{\mathrm{s},j}(x_j)}{\prod_{j=1}^{N} \mathsf{q}_{\mathrm{v},j}(x_j)} \; . \tag{5.4}$$

5.1.1.2 The Moment-Matching Constraint

The consistency constraints, which need to be defined per edge in the factor graph, are in the given case set up for all variables $x_1, \ldots, x_N$. Since each variable node is connected to a signal factor and the channel factor, two constraints need to be established per variable node. As before, exponential families are used, especially in the form of Gaussian distributions, because eventually the (first-order) moment of the MMSE estimate (2.9) is desired. However, since the signal factors are considered individually per variable, it is necessary to specify the reliability of the estimation individually. Therefore, the sufficient statistics of the exponential family is determined by

$$\boldsymbol{g}(\boldsymbol{x}) = [\boldsymbol{g}_{\boldsymbol{\lambda}}(\boldsymbol{x})^{\top}, \boldsymbol{g}_{\boldsymbol{\Lambda}}(\boldsymbol{x})^{\top}]^{\top} \tag{5.5}$$

with

$$\boldsymbol{g}_{\boldsymbol{\lambda}}(\boldsymbol{x}) = [x_1, \ldots, x_N]^{\top} \; , \quad \boldsymbol{g}_{\boldsymbol{\Lambda}}(\boldsymbol{x}) = -\frac{1}{2}[x_1^2, \ldots, x_N^2]^{\top} \; , \tag{5.6}$$

cf. Ex. D.2 in Appendix D. For convenience, the notation $\boldsymbol{g}(x_j) = [g_{\lambda_j}(x_j),\, g_{\Lambda_j}(x_j)]^\top = [x_j,\, -x_j^2/2]^\top$ may be used. With these specifications, the constraints on the moments read

$$\mathrm{E}_{\mathsf{x}_j \sim \mathsf{q}_{\mathrm{v},j}}\{\boldsymbol{g}(\mathsf{x}_j)\} \overset{!}{=} \mathrm{E}_{\mathsf{x}_j \sim \mathsf{q}_{\mathrm{s},j}}\{\boldsymbol{g}(\mathsf{x}_j)\}\,, \qquad j \in \{1,\, \ldots,\, N\}\,, \tag{5.7}$$

$$\mathrm{E}_{\mathsf{x}_j \sim \mathsf{q}_{\mathrm{v},j}}\{\boldsymbol{g}(\mathsf{x}_j)\} \overset{!}{=} \mathrm{E}_{\mathsf{x}_j \sim \mathsf{q}_{\mathrm{c}}}\{\boldsymbol{g}(\mathsf{x}_j)\}\,, \qquad j \in \{1,\, \ldots,\, N\}\,. \tag{5.8}$$

5.1.2 Constrained Optimization

The optimization-based approach that is used to derive the parameter connections for the algorithm takes on the form of a constrained optimization problem, where the task of minimizing the Kullback–Leibler divergence is combined with the moment constraints from above and normalization constraints for the representatives $\mathsf{q}_{\mathrm{c}}(\boldsymbol{x})$, $\mathsf{q}_{\mathrm{s},j}(x_j)$, and $\mathsf{q}_{\mathrm{v},j}(x_j)$ ($j \in \{1,\, \ldots,\, N\}$). The entire problem reads

$$\begin{aligned}
&\text{minimize} && \mathrm{D}_{\mathrm{KL}}\Big(\mathsf{q}(\boldsymbol{x}) \,||\, \mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})\Big) \\
&\text{subject to} && \mathrm{E}_{\mathsf{x}_j \sim \mathsf{q}_{\mathrm{v},j}}\{\boldsymbol{g}(\mathsf{x}_j)\} - \mathrm{E}_{\mathsf{x}_j \sim \mathsf{q}_{\mathrm{c}}}\{\boldsymbol{g}(\mathsf{x}_j)\} = \mathbf{0}\,, \qquad j \in \{1,\, \ldots,\, N\}\,, \\
&&& \mathrm{E}_{\mathsf{x}_j \sim \mathsf{q}_{\mathrm{v},j}}\{\boldsymbol{g}(\mathsf{x}_j)\} - \mathrm{E}_{\mathsf{x}_j \sim \mathsf{q}_{\mathrm{s},j}}\{\boldsymbol{g}(\mathsf{x}_j)\} = \mathbf{0}\,, \qquad j \in \{1,\, \ldots,\, N\}\,, \\
&&& \int \mathsf{q}_{\mathrm{v},j}(x_j)\,\mathrm{d}x_j = 1\,, \quad \int \mathsf{q}_{\mathrm{s},j}(x_j)\,\mathrm{d}x_j = 1\,, \quad j \in \{1,\, \ldots,\, N\}\,, \\
&&& \int \mathsf{q}_{\mathrm{c}}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = 1\,.
\end{aligned} \tag{5.9}$$

5.1.2.1 Cost Function for Compressed Sensing

For the sake of completeness, the form of the Kullback–Leibler divergence for the given case is stated, which is somewhat a mixture of the cases in the chapters before

$$\begin{aligned}
\mathrm{D}_{\mathrm{KL}}\Big(\mathsf{q}(\boldsymbol{x}) \,||\, \mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})\Big) = {} & \sum_{j=1}^{N} \int \mathsf{q}_{\mathrm{s},j}(x_j) \log \mathsf{q}_{\mathrm{s},j}(x_j)\,\mathrm{d}x_j + \int \mathsf{q}_{\mathrm{c}}(\boldsymbol{x}) \log \mathsf{q}_{\mathrm{c}}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \\
& - \sum_{j=1}^{N} \int \mathsf{q}_{\mathrm{s},j}(x_j) \log \mathsf{f}_{\mathsf{x}}(x_j)\,\mathrm{d}x_j - \int \mathsf{q}_{\mathrm{c}}(\boldsymbol{x}) \log \mathsf{f}_{\mathbf{y}|\mathbf{x}}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \\
& - \sum_{j=1}^{N} \int \mathsf{q}_{\mathrm{v},j}(x_j) \log \mathsf{q}_{\mathrm{v},j}(x_j)\,\mathrm{d}x_j\,.
\end{aligned} \tag{5.10}$$

5.1.2.2 Stationary Points of the Lagrangian

To see the parameter connections that are necessary for the recovery algorithm, a Lagrangian is set up and the stationary points are derived, because the stationary points indicate the form of the solution to the problem. For the Lagrangian, multipliers $\boldsymbol{\nu}_{\mathrm{c},j}$ and $\boldsymbol{\nu}_{\mathrm{s},j}$ ($j \in \{1,\, \ldots,\, N\}$) are defined for the moment-matching constraint, as well as ν_{c}, ν_j, and $\nu_{\mathrm{s},j}$ ($j \in \{1,\, \ldots,\, N\}$)

for the normalization constraint. The resulting Lagrangian reads

$$\begin{aligned}\mathcal{L}(\mathsf{q}(\boldsymbol{x})) = \mathrm{D}_{\mathrm{KL}}\Big(\mathsf{q}(\boldsymbol{x}) \mid\mid \mathsf{f}_{\mathsf{x}|\mathsf{y}}(\boldsymbol{x})\Big) &+ \sum_{j=1}^{N} \boldsymbol{\nu}_{\mathrm{c},j}^{\top} \Big(\mathrm{E}_{\mathsf{x}_j \sim \mathsf{q}_{\mathrm{v},j}}\{\boldsymbol{g}(\mathsf{x}_j)\} - \mathrm{E}_{\mathsf{x}_j \sim \mathsf{q}_{\mathrm{c}}}\{\boldsymbol{g}(\mathsf{x}_j)\}\Big) \\ &+ \sum_{j=1}^{N} \boldsymbol{\nu}_{\mathrm{s},j}^{\top} \Big(\mathrm{E}_{\mathsf{x}_j \sim \mathsf{q}_{\mathrm{v},j}}\{\boldsymbol{g}(\mathsf{x}_j)\} - \mathrm{E}_{\mathsf{x}_j \sim \mathsf{q}_{\mathrm{s},j}}\{\boldsymbol{g}(\mathsf{x}_j)\}\Big) \\ &+ \sum_{j=1}^{N} \Big(\nu_j \big(\int \mathsf{q}_{\mathrm{v},j}(x_j)\,\mathrm{d}x_j - 1\big) + \nu_{\mathrm{s},j}\big(\int \mathsf{q}_{\mathrm{s},j}(x_j)\,\mathrm{d}x_j - 1\big)\Big) \\ &+ \nu_{\mathrm{c}}\big(\int \mathsf{q}_{\mathrm{c}}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} - 1\big)\ . \end{aligned} \tag{5.11}$$

Similar to the previous chapters, the stationary points are obtained by differentiating w.r.t. the representatives $\mathsf{q}_{\mathrm{c}}(\boldsymbol{x})$, $\mathsf{q}_{\mathrm{v},j}(x_j)$, and $\mathsf{q}_{\mathrm{s},j}(x_j)$ ($j \in \{1, \ldots, N\}$), respectively, setting the results to zero and solve for the respective representative, i.e.,

$$\frac{\partial}{\partial \mathsf{q}_{\mathrm{v},j}} \mathcal{L}(\mathsf{q}(\boldsymbol{x})) = -\log \mathsf{q}_{\mathrm{v},j}(x_j) + (\boldsymbol{\nu}_{\mathrm{c},j} + \boldsymbol{\nu}_{\mathrm{s},j})^{\top} \boldsymbol{g}(\mathsf{x}_j) + \mathrm{const.} \stackrel{!}{=} 0\ , \tag{5.12}$$

$$\frac{\partial}{\partial \mathsf{q}_{\mathrm{s},j}} \mathcal{L}(\mathsf{q}(\boldsymbol{x})) = \log \mathsf{q}_{\mathrm{s},j}(x_j) - \log \mathsf{f}_{\mathsf{x}}(x_j) - \boldsymbol{\nu}_{\mathrm{s},j}^{\top} \boldsymbol{g}(\mathsf{x}_j) + \mathrm{const.} \stackrel{!}{=} 0\ , \tag{5.13}$$

$$\frac{\partial}{\partial \mathsf{q}_{\mathrm{c}}} \mathcal{L}(\mathsf{q}(\boldsymbol{x})) = \log \mathsf{q}_{\mathrm{c}}(\boldsymbol{x}) - \log \mathsf{f}_{\mathsf{y}|\mathsf{x}}(\boldsymbol{x}) - \sum_{j=1}^{N} \boldsymbol{\nu}_{\mathrm{c},j}^{\top} \boldsymbol{g}(\mathsf{x}_j) + \mathrm{const.} \stackrel{!}{=} 0\ . \tag{5.14}$$

Finally, one obtains

$$\mathsf{q}_{\mathrm{v},j}(x_j) = \frac{1}{Z_{\mathrm{v},j}} \exp\Big((\boldsymbol{\nu}_{\mathrm{s},j} + \boldsymbol{\nu}_{\mathrm{c},j})^{\top} \boldsymbol{g}(x_j)\Big)\ , \tag{5.15}$$

$$\mathsf{q}_{\mathrm{c}}(\boldsymbol{x}) = \frac{1}{Z_{\mathrm{c}}} \mathsf{f}_{\mathsf{y}|\mathsf{x}}(\boldsymbol{x}) \exp\left(\sum_{j=1}^{N} \boldsymbol{\nu}_{\mathrm{c},j}^{\top} \boldsymbol{g}(x_j)\right)\ , \tag{5.16}$$

$$\mathsf{q}_{\mathrm{s},j}(x_j) = \frac{1}{Z_{\mathrm{s},j}} \mathsf{f}_{\mathsf{x}}(x_j) \exp\Big(\boldsymbol{\nu}_{\mathrm{s},j}^{\top} \boldsymbol{g}(x_j)\Big)\ , \tag{5.17}$$

with normalizing partition functions $Z_{\mathrm{v},j}$, Z_{c}, and $Z_{\mathrm{s},j}$.

The basic result of the derivations is that again the connection of the natural parameters in the variable node is given by

$$\boldsymbol{\theta}_j \stackrel{\mathrm{def}}{=} \boldsymbol{\nu}_{\mathrm{c},j} + \boldsymbol{\nu}_{\mathrm{s},j}\ , \tag{5.18}$$

which means that in the algorithm, *post-processing* is done as extrinsic calculation, i.e., the inputs to the estimations are obtained by subtractions of previously computed estimates. The procedure will be discussed in detail below.

5.1.2.3 Connection to Expectation Propagation

The basic idea behind expectation propagation (EP) [Min01] is to refine an approximation $\mathsf{q}(\boldsymbol{x})$ by replacing a representing factor of the approximation by the corresponding exact term

and projecting the refinement back into the space of distributions that is considered. This is iteratively conducted for all the parts / factors, that are available in the structure of the considered problem.

In terms of exponential families, it is directly visible how this connects to the derivation here. Starting with a representative $\mathsf{q}(\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{\theta})} \exp\Big(\boldsymbol{\theta}^\top \boldsymbol{g}(\boldsymbol{x})\Big)$, which encodes the structure of the problem by a respective summation of its natural parameters, e.g., $\boldsymbol{\theta}_j = \boldsymbol{\nu}_{\mathrm{c},j} + \boldsymbol{\nu}_{\mathrm{s},j}$ in the case above (then the entire natural parameter might be $\boldsymbol{\theta} = [\boldsymbol{\theta}_1^\top, \ldots, \boldsymbol{\theta}_N^\top]^\top$). The representing factor is removed by subtracting its corresponding natural parameter from the natural parameter representing the entire distribution. The resulting parameter is called the *cavity parameter*, because it excludes the influence of the respective factor. Inserting the exact factor—in the given case any of $\mathsf{f}_\mathsf{x}(x_j)$ ($j \in \{1, \ldots, N\}$)—yields something of the form of the stationary points $\mathsf{q}_{\mathrm{s},j}(x_j) = \frac{1}{Z_{\mathrm{s},j}} \mathsf{f}_\mathsf{x}(x_j) \exp\Big(\boldsymbol{\nu}_{\mathrm{s},j}^\top \boldsymbol{g}(x_j)\Big)$ of the Lagrangian, i.e., the cavity parameter is here $\boldsymbol{\nu}_{\mathrm{s},j} = \boldsymbol{\theta}_j - \boldsymbol{\nu}_{\mathrm{c},j}$. The projection is then performed as minimization of the Kullback–Leibler divergence such that the result is a member of an exponential family, i.e., of the form, e.g., [Min05]

$$\mathsf{q}_{\mathrm{v},j}(x_j) = \min_{\mathsf{q}(x_j)} \mathrm{D}_{\mathrm{KL}}(\mathsf{q}_{\mathrm{s},j}(x_j) \mid\mid \mathsf{q}(x_j)) \; . \tag{5.19}$$

As indicated, the result of the projection is of the form of $\mathsf{q}_{\mathrm{v},j}(x_j)$ from (5.15). The restriction to members of an exponential family causes a matching of moments, as shown in the insert below. Noteworthy, the minimization of the Kullback–Leibler divergence considered here has the position of substitute distribution and given distribution swapped compared to the instances used so far and it requires arguments of a different form.

The matching of moments can be interpreted as in Chapter 3 and be realized by computing moments and the resulting natural parameters of a corresponding member of an exponential family, which yields an updated $\boldsymbol{\theta}_j$. The remaining task is to update the so-called *site parameters*, i.e., the complementary part of the cavity parameters, here by $\boldsymbol{\nu}_{\mathrm{c},j} = \boldsymbol{\theta}_j - \boldsymbol{\nu}_{\mathrm{s},j}$.

Another interesting aspect is that the computation of extrinsic is directly realized in this procedure and can intuitively be explained by the fact that one has to make room in the approximation (by subtracting the corresponding natural parameter representation) for the exact part to be inserted.

Insert

We show that, when restricting the result of the minimization (5.19) to an exponential family, a matching of moments is obtained. Assume that $\mathsf{q}(x_j) = \frac{1}{Z(\boldsymbol{\theta})} \exp\Big(\boldsymbol{\theta}^\top \boldsymbol{g}(x_j)\Big)$. Under this assumption, the Kullback–Leibler divergence can be written as

$$\mathrm{D}_{\mathrm{KL}}(\mathsf{q}_{\mathrm{s},j}(x_j) \mid\mid \mathsf{q}(x_j)) = \int \mathsf{q}_{\mathrm{s},j}(x_j) \log \mathsf{q}_{\mathrm{s},j}(x_j) \, \mathrm{d}x_j - \boldsymbol{\theta}^\top \int \mathsf{q}_{\mathrm{s},j}(x_j) \boldsymbol{g}(x_j) \, \mathrm{d}x_j + \log Z(\boldsymbol{\theta}) \; . \tag{5.20}$$

Then differentiation of $\mathrm{D}_{\mathrm{KL}}(\mathsf{q}_{\mathrm{s},j}(x_j) || \mathsf{q}(x_j))$ w.r.t. $\boldsymbol{\theta}$ yields

$$\begin{aligned} \frac{\partial}{\partial \boldsymbol{\theta}} \mathrm{D}_{\mathrm{KL}}(\mathsf{q}_{\mathrm{s},j}(x_j) \mid\mid \mathsf{q}(x_j)) &= \frac{\partial}{\partial \boldsymbol{\theta}} \Big(\log Z(\boldsymbol{\theta}) - \boldsymbol{\theta}^\top \mathrm{E}_{\mathsf{x}_j \sim \mathsf{q}_{\mathrm{s},j}} \{\boldsymbol{g}(\mathsf{x}_j)\} \Big) \\ &= \mathrm{E}_{\mathsf{x}_j \sim \mathsf{q}} \{\boldsymbol{g}(\mathsf{x}_j)\} - \mathrm{E}_{\mathsf{x}_j \sim \mathsf{q}_{\mathrm{s},j}} \{\boldsymbol{g}(\mathsf{x}_j)\} \; , \end{aligned} \tag{5.21}$$

where (D.18) was used for the differentiation of $\log Z(\boldsymbol{\theta})$. Subsequently setting this to zero, yields the matching of moments $\mathrm{E}_{\mathsf{x}_j\sim\mathsf{q}}\{\boldsymbol{g}(\mathsf{x}_j)\} = \mathrm{E}_{\mathsf{x}_j\sim\mathsf{q}_{\mathrm{s},j}}\{\boldsymbol{g}(\mathsf{x}_j)\}$, which means that any extremum must fulfill this relation.

End Insert

5.1.3 Expectations and Rank-One Update

The approach considered here has been motivated by the idea of refining the LMMSE (channel-constrained) estimate sequentially based on the impact of single estimations on the signal-constrained side of the problem. For the signal-constrained estimations, the individual calculations are needed for which it is known from Chapter 3 that with $\boldsymbol{\nu}_{\mathrm{s},j} = [\tilde{x}_{\mathrm{s},j}/\tilde{\sigma}^2_{\mathrm{s},j},\, 1/\tilde{\sigma}^2_{\mathrm{s},j}]^\top$, it is

$$\mathrm{E}_{\mathsf{x}_j\sim\mathsf{q}_{\mathrm{s},j}}\Big\{g_{\lambda_j}(\mathsf{x}_j)\Big\} = \mathrm{E}_{\mathsf{x}_j}\{\mathsf{x}_j \mid \tilde{x}_{\mathrm{s},j}\} = m_{\mathrm{s},j}\ , \tag{5.22}$$

$$\mathrm{E}_{\mathsf{x}_j\sim\mathsf{q}_{\mathrm{s},j}}\Big\{g_{\Lambda_j}(\mathsf{x}_j)\Big\} = -\frac{1}{2}\mathrm{E}_{\mathsf{x}_j}\Big\{\mathsf{x}_j^2 \mid \tilde{x}_{\mathrm{s},j}\Big\} = -\frac{1}{2}\left(\sigma^2_{\mathrm{s},j} + m^2_{\mathrm{s},j}\right)\ . \tag{5.23}$$

On the side of the channel-constrained estimation it is useful to stack the input parameters such that

$$\sum_{j=1}^{N} \boldsymbol{\nu}_{\mathrm{c},j}^\top \boldsymbol{g}(x_j) = \boldsymbol{\nu}_{\mathrm{c}}^\top \boldsymbol{g}(\boldsymbol{x})\ . \tag{5.24}$$

With the identification

$$\boldsymbol{\nu}_{\mathrm{c}} = [m_{\mathsf{x},1}/\sigma^2_{\mathsf{x},1},\, \ldots,\, m_{\mathsf{x},N}/\sigma^2_{\mathsf{x},N},\, 1/\sigma^2_{\mathsf{x},1},\, \ldots,\, 1/\sigma^2_{\mathsf{x},N}]^\top\ , \tag{5.25}$$

it can be referred to Chapter 4, where for individual variances it has been found that

$$\mathrm{E}_{\mathbf{x}\sim\mathsf{q}_{\mathrm{c}}}\{\boldsymbol{g}_{\boldsymbol{\lambda}}(\mathbf{x})\} = \boldsymbol{m}_{\mathrm{c}}\ , \tag{5.26}$$

respectively,

$$\mathrm{E}_{\mathbf{x}\sim\mathsf{q}_{\mathrm{c}}}\{\boldsymbol{g}_{\boldsymbol{\Lambda}}(\mathbf{x})\} = -\frac{1}{2}\mathbf{diag}\Big(\boldsymbol{\Phi}_{\mathrm{c}} + \boldsymbol{m}_{\mathrm{c}}\boldsymbol{m}_{\mathrm{c}}^\top\Big)\ . \tag{5.27}$$

For the moment-matching, the respective jth components are required, i.e.,

$$\mathrm{E}_{\mathsf{x}_j\sim\mathsf{q}_{\mathrm{c}}}\Big\{g_{\lambda_j}(\mathsf{x}_j)\Big\} = m_{\mathrm{c},j}\ , \tag{5.28}$$

and

$$\mathrm{E}_{\mathsf{x}_j\sim\mathsf{q}_{\mathrm{c}}}\Big\{g_{\Lambda_j}(\mathsf{x}_j)\Big\} = -\frac{1}{2}(\sigma^2_{\mathrm{c},j} + m^2_{\mathrm{c},j})\ . \tag{5.29}$$

In the following, the effect of the changes in the channel-constrained estimates induced by a single signal-constrained estimation at position j is considered. To that end, assume that

$\boldsymbol{\nu}_\mathrm{c}$ and the corresponding estimates $\boldsymbol{m}_\mathrm{c}$ and $\boldsymbol{\Phi}_\mathrm{c}$ are fixed. In order to simplify the notation, the following definitions are set up

$$\boldsymbol{\lambda}_\mathrm{c} \stackrel{\text{def}}{=} [\lambda_{\mathrm{c},1},\, \ldots,\, \lambda_{\mathrm{c},N}]^\top = [m_{\mathsf{x},1}/\sigma^2_{\mathsf{x},1},\, \ldots,\, m_{\mathsf{x},N}/\sigma^2_{\mathsf{x},N}]^\top \,, \tag{5.30}$$

$$\boldsymbol{\Lambda}_\mathrm{c} \stackrel{\text{def}}{=} [\Lambda_{\mathrm{c},1},\, \ldots,\, \Lambda_{\mathrm{c},N}]^\top = [\quad 1/\sigma^2_{\mathsf{x},1},\, \ldots,\quad 1/\sigma^2_{\mathsf{x},N}]^\top \,, \tag{5.31}$$

then the combined input parameter reads $\boldsymbol{\nu}_\mathrm{c} = [\boldsymbol{\lambda}_\mathrm{c}^\top,\, \boldsymbol{\Lambda}_\mathrm{c}^\top]^\top$. With these input parameters, the channel-constrained estimation can be written as follows by reformulating (2.10) and making the identifications $\boldsymbol{\Phi}_\mathsf{x}^{-1} = \mathbf{diag}(\Lambda_{\mathrm{c},j})$ and $\boldsymbol{\lambda}_\mathrm{c} = \boldsymbol{\Phi}_\mathsf{x}^{-1}\boldsymbol{m}_\mathsf{x} = \mathbf{diag}(\Lambda_{\mathrm{c},j})\boldsymbol{m}_\mathsf{x}$

$$\begin{aligned}
\boldsymbol{\Phi}_\mathrm{c} &= \sigma^2_\mathsf{n}(\boldsymbol{A}^\top\boldsymbol{A} + \sigma^2_\mathsf{n}\mathbf{diag}(\Lambda_{\mathrm{c},j}))^{-1} \\
&= (\frac{1}{\sigma^2_\mathsf{n}}\boldsymbol{A}^\top\boldsymbol{A} + \mathbf{diag}(\Lambda_{\mathrm{c},j}))^{-1} \,, \qquad (5.32)\\
\boldsymbol{m}_\mathrm{c} &= \boldsymbol{m}_\mathsf{x} + (\boldsymbol{A}^\top\boldsymbol{A} + \sigma^2_\mathsf{n}\mathbf{diag}(\Lambda_{\mathrm{c},j}))^{-1}\boldsymbol{A}^\top(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{m}_\mathsf{x}) \\
&= (\boldsymbol{A}^\top\boldsymbol{A} + \sigma^2_\mathsf{n}\mathbf{diag}(\Lambda_{\mathrm{c},j}))^{-1}((\boldsymbol{A}^\top\boldsymbol{A} + \sigma^2_\mathsf{n}\mathbf{diag}(\Lambda_{\mathrm{c},j}))\boldsymbol{m}_\mathsf{x} + \boldsymbol{A}^\top\boldsymbol{y} - \boldsymbol{A}^\top\boldsymbol{A}\boldsymbol{m}_\mathsf{x}) \\
&= (\boldsymbol{A}^\top\boldsymbol{A} + \sigma^2_\mathsf{n}\mathbf{diag}(\Lambda_{\mathrm{c},j}))^{-1}(\boldsymbol{A}^\top\boldsymbol{y} + \sigma^2_\mathsf{n}\mathbf{diag}(\Lambda_{\mathrm{c},j})\boldsymbol{m}_\mathsf{x}) \\
&= (\frac{1}{\sigma^2_\mathsf{n}}\boldsymbol{A}^\top\boldsymbol{A} + \mathbf{diag}(\Lambda_{\mathrm{c},j}))^{-1}(\frac{1}{\sigma^2_\mathsf{n}}\boldsymbol{A}^\top\boldsymbol{y} + \boldsymbol{\lambda}_\mathrm{c}) \\
&= \boldsymbol{\Phi}_\mathrm{c}(\frac{1}{\sigma^2_\mathsf{n}}\boldsymbol{A}^\top\boldsymbol{y} + \boldsymbol{\lambda}_\mathrm{c}) \,. \qquad (5.33)
\end{aligned}$$

The change in position j affects only the input parameter $\boldsymbol{\nu}_{\mathrm{c},j} = [\lambda_{\mathrm{c},j},\, \Lambda_{\mathrm{c},j}]^\top$. The changed (new) parameters at position j are denoted by $\lambda^\circ_{\mathrm{c},j}$ and $\Lambda^\circ_{\mathrm{c},j}$, respectively. With jth unit vector $\boldsymbol{e}_j$, which contains a 1 at the jth position and zeros elsewhere, and $\Delta\lambda_{\mathrm{c},j} = \lambda^\circ_{\mathrm{c},j} - \lambda_{\mathrm{c},j}$, as well as $\Delta\Lambda_{\mathrm{c},j} = \Lambda^\circ_{\mathrm{c},j} - \Lambda_{\mathrm{c},j}$ one can write the resulting input parameter vectors as (cf. [OW05])

$$\boldsymbol{\lambda}^\circ_\mathrm{c} = [\lambda_{\mathrm{c},1},\, \ldots,\, \lambda_{\mathrm{c},j-1},\, \lambda^\circ_{\mathrm{c},j},\, \lambda_{\mathrm{c},j+1},\, \ldots,\, \lambda_{\mathrm{c},N}]^\top = \boldsymbol{\lambda}_\mathrm{c} + \Delta\lambda_{\mathrm{c},j}\boldsymbol{e}_j \,, \tag{5.34}$$

$$\boldsymbol{\Lambda}^\circ_\mathrm{c} = [\Lambda_{\mathrm{c},1},\, \ldots,\, \Lambda_{\mathrm{c},j-1},\, \Lambda^\circ_{\mathrm{c},j},\, \Lambda_{\mathrm{c},j+1},\, \ldots,\, \Lambda_{\mathrm{c},N}]^\top = \boldsymbol{\Lambda}_\mathrm{c} + \Delta\Lambda_{\mathrm{c},j}\boldsymbol{e}_j \,. \tag{5.35}$$

The last line can be written in matrix form as

$$\mathbf{diag}(\boldsymbol{\Lambda}^\circ_\mathrm{c}) = \mathbf{diag}(\boldsymbol{\Lambda}_\mathrm{c}) + \Delta\Lambda_{\mathrm{c},j}\boldsymbol{e}_j\boldsymbol{e}_j^\top \,. \tag{5.36}$$

The additive form of the change in the vector helps us to formulate the change in the resulting channel-constrained as follows. Using the Sherman-Morrison-Woodbury identity [GV96], the new covariance matrix $\boldsymbol{\Phi}^\circ_\mathrm{c}$ can be written as a rank-one update of the previous one $\boldsymbol{\Phi}_\mathrm{c}$ via

$$\begin{aligned}
\boldsymbol{\Phi}^\circ_\mathrm{c} &= \left(\frac{1}{\sigma^2_\mathsf{n}}\boldsymbol{A}^\top\boldsymbol{A} + \mathbf{diag}(\boldsymbol{\Lambda}^\circ_\mathrm{c})\right)^{-1} = \left(\boldsymbol{\Phi}_\mathrm{c}^{-1} + \Delta\Lambda_{\mathrm{c},j}\boldsymbol{e}_j\boldsymbol{e}_j^\top\right)^{-1} \\
&= \boldsymbol{\Phi}_\mathrm{c} - \frac{\Delta\Lambda_{\mathrm{c},j}}{1 + \Delta\Lambda_{\mathrm{c},j}\boldsymbol{e}_j^\top\boldsymbol{\Phi}_\mathrm{c}\boldsymbol{e}_j}\boldsymbol{\Phi}_\mathrm{c}\boldsymbol{e}_j\boldsymbol{e}_j^\top\boldsymbol{\Phi}_\mathrm{c} \,. \qquad (5.37)
\end{aligned}$$

Since the unit vector picks the jth column, i.e., it is $\boldsymbol{\Phi}_\mathrm{c}\boldsymbol{e}_j = \boldsymbol{\phi}_{\mathrm{c},j}$, respectively $\boldsymbol{e}_j^\top\boldsymbol{\Phi}_\mathrm{c} = \boldsymbol{\phi}_{\mathrm{c},j}^\top$ for $\boldsymbol{\Phi}_\mathrm{c} = [\boldsymbol{\phi}_{\mathrm{c},1},\, \ldots,\, \boldsymbol{\phi}_{\mathrm{c},N}]^\top$ and $\boldsymbol{e}_j^\top\boldsymbol{\Phi}_\mathrm{c}\boldsymbol{e}_j = \sigma^2_{\mathrm{c},j}$, this can be simplified to

$$\boldsymbol{\Phi}^\circ_\mathrm{c} = \boldsymbol{\Phi}_\mathrm{c} - \frac{\Delta\Lambda_{\mathrm{c},j}}{1 + \Delta\Lambda_{\mathrm{c},j}\sigma^2_{\mathrm{c},j}}\boldsymbol{\phi}_{\mathrm{c},j}\boldsymbol{\phi}_{\mathrm{c},j}^\top \,. \tag{5.38}$$

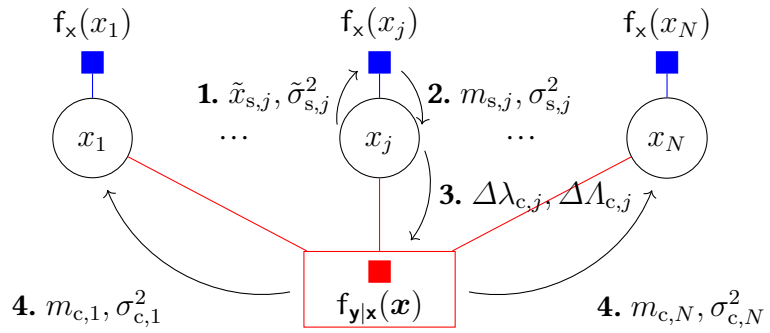


Figure 5.2: Procedure for sequential processing of variable nodes [SF21].

The corresponding new conditional mean $\boldsymbol{m}_{\mathrm{c}}^{\circ}$ can be calculated from the "old" one $\boldsymbol{m}_{\mathrm{c}}$ by inserting the result into (5.33) and using (5.34)

$$\begin{aligned}
\boldsymbol{m}_{\mathrm{c}}^{\circ} &= \boldsymbol{\Phi}_{\mathrm{c}}^{\circ}\left(\frac{1}{\sigma_{\mathsf{n}}^2}\boldsymbol{A}^{\top}\boldsymbol{y} + \boldsymbol{\lambda}_{\mathrm{c}}^{\circ}\right) \\
&= \left(\boldsymbol{\Phi}_{\mathrm{c}} - \frac{\Delta\Lambda_{\mathrm{c},j}}{1+\Delta\Lambda_{\mathrm{c},j}\sigma_{\mathrm{c},j}^2}\boldsymbol{\phi}_{\mathrm{c},j}\boldsymbol{\phi}_{\mathrm{c},j}^{\top}\right)\left(\frac{1}{\sigma_{\mathsf{n}}^2}\boldsymbol{A}^{\top}\boldsymbol{y} + \boldsymbol{\lambda}_{\mathrm{c}} + \Delta\lambda_{\mathrm{c},j}\boldsymbol{e}_j\right) \\
&= \boldsymbol{\Phi}_{\mathrm{c}}\left(\frac{1}{\sigma_{\mathsf{n}}^2}\boldsymbol{A}^{\top}\boldsymbol{y} + \boldsymbol{\lambda}_{\mathrm{c}}\right) + \boldsymbol{\Phi}_{\mathrm{c}}\Delta\lambda_{\mathrm{c},j}\boldsymbol{e}_j - \frac{\Delta\Lambda_{\mathrm{c},j}}{1+\Delta\Lambda_{\mathrm{c},j}\sigma_{\mathrm{c},j}^2}\boldsymbol{\phi}_{\mathrm{c},j}\boldsymbol{\phi}_{\mathrm{c},j}^{\top}\left(\frac{1}{\sigma_{\mathsf{n}}^2}\boldsymbol{A}^{\top}\boldsymbol{y} + \boldsymbol{\lambda}_{\mathrm{c}}\right) \\
&\quad - \frac{\Delta\Lambda_{\mathrm{c},j}\Delta\lambda_{\mathrm{c},j}}{1+\Delta\Lambda_{\mathrm{c},j}\sigma_{\mathrm{c},j}^2}\boldsymbol{\phi}_{\mathrm{c},j}\boldsymbol{\phi}_{\mathrm{c},j}^{\top}\boldsymbol{e}_j \,.
\end{aligned} \tag{5.39}$$

Noticing that $m_{\mathrm{c},j} = \boldsymbol{\phi}_{\mathrm{c},j}^{\top}\left(\frac{1}{\sigma_{\mathsf{n}}^2}\boldsymbol{A}^{\top}\boldsymbol{y} + \boldsymbol{\lambda}_{\mathrm{c}}\right)$ and $\boldsymbol{\phi}_{\mathrm{c},j}\boldsymbol{e}_j = \sigma_{\mathrm{c},j}^2$, this may be simplified to

$$\begin{aligned}
\boldsymbol{m}_{\mathrm{c}}^{\circ} &= \boldsymbol{m}_{\mathrm{c}} + \boldsymbol{\phi}_{\mathrm{c},j}\Delta\lambda_{\mathrm{c},j}\left(1 - \frac{\Delta\Lambda_{\mathrm{c},j}\sigma_{\mathrm{c},j}^2}{1+\Delta\Lambda_{\mathrm{c},j}\sigma_{\mathrm{c},j}^2}\right) - \frac{\Delta\Lambda_{\mathrm{c},j}m_{\mathrm{c},j}}{1+\Delta\Lambda_{\mathrm{c},j}\sigma_{\mathrm{c},j}^2}\boldsymbol{\phi}_{\mathrm{c},j} \\
&= \boldsymbol{m}_{\mathrm{c}} + \frac{\Delta\lambda_{\mathrm{c},j} - \Delta\Lambda_{\mathrm{c},j}m_{\mathrm{c},j}}{1+\Delta\Lambda_{\mathrm{c},j}\sigma_{\mathrm{c},j}^2}\boldsymbol{\phi}_{\mathrm{c},j} \,.
\end{aligned} \tag{5.40}$$

This way, a change induced by signal-constrained estimation of one variable, can simply be propagated to the other variables. The procedure is depicted in Fig. 5.2; starting with a signal-constrained estimation, the change in the extrinsic parameters is fed into the channel-constrained estimation and via the rank-one update, the impact on other variables can be computed. Then, another variable can be chosen.

The procedure of the resulting algorithm is summarized in the section below.

5.1.4 Sequential Version of VAMPind

The basis for the algorithm is the linear estimate $\boldsymbol{m}_{\mathrm{c}}$, which is sequentially refined by single signal-constrained estimations. Therefore, the algorithm is initialized by computing the LMMSE estimate (5.33) and (5.32) once with suitable values for $\boldsymbol{\lambda}_{\mathrm{c}}$ and $\boldsymbol{\Lambda}_{\mathrm{c}}$.

For the refinement, a component x_j of the signal vector $\boldsymbol{x}$ is chosen. As it is known from (5.18), the extrinsic is calculated via a subtraction. Hence, the input to the signal-constrained

estimator is computed via

$$\tilde{x}_{\mathrm{s},j}/\tilde{\sigma}^2_{\mathrm{s},j} = m_{\mathrm{c},j}/\sigma^2_{\mathrm{c},j} - \lambda_{\mathrm{c},j} \;, \qquad 1/\tilde{\sigma}^2_{\mathrm{s},j} = 1/\sigma^2_{\mathrm{c},j} - \Lambda_{\mathrm{c},j} \;, \tag{5.41}$$

so that the signal-constrained estimate can be computed as usual by

$$m_{\mathrm{s},j} = \mathrm{E}_{\mathsf{x}_j}\left\{\mathsf{x}_j \mid \tilde{x}_{\mathrm{s},j},\, \tilde{\sigma}^2_{\mathrm{s},j}\right\} , \tag{5.42}$$

$$\sigma^2_{\mathrm{s},j} = \mathrm{E}_{\mathsf{x}_j}\left\{(\mathsf{x}_j - m_{\mathrm{s},j})^2 \mid \tilde{x}_{\mathrm{s},j},\, \tilde{\sigma}^2_{\mathrm{s},j}\right\} . \tag{5.43}$$

The connection (5.18) is used afterwards for the extrinsic calculation again, so that

$$\lambda^{\circ}_{\mathrm{c},j} = m_{\mathrm{s},j}/\sigma^2_{\mathrm{s},j} - \tilde{x}_{\mathrm{s},j}/\tilde{\sigma}^2_{\mathrm{s},j} \;, \quad \Lambda^{\circ}_{\mathrm{c},j} = 1/\sigma^2_{\mathrm{s},j} - 1/\tilde{\sigma}^2_{\mathrm{s},j} \;, \tag{5.44}$$

The additive parameters $\Delta\lambda_{\mathrm{c},j}$ and $\Delta\Lambda_{\mathrm{c},j}$ in (5.34) and (5.35), respectively, thus, read

$$\begin{aligned}
\Delta\lambda_{\mathrm{c},j} &= \lambda^{\circ}_{\mathrm{c},j} - \lambda_{\mathrm{c},j} = m_{\mathrm{s},j}/\sigma^2_{\mathrm{s},j} - \tilde{x}_{\mathrm{s},j}/\tilde{\sigma}^2_{\mathrm{s},j} - \lambda_{\mathrm{c},j} \\
&= m_{\mathrm{s},j}/\sigma^2_{\mathrm{s},j} - (m_{\mathrm{c},j}/\sigma^2_{\mathrm{c},j} - \lambda_{\mathrm{c},j}) - \lambda_{\mathrm{c},j} = m_{\mathrm{s},j}/\sigma^2_{\mathrm{s},j} - m_{\mathrm{c},j}/\sigma^2_{\mathrm{c},j} \;, \qquad (5.45) \\
\Delta\Lambda_{\mathrm{c},j} &= \Lambda^{\circ}_{\mathrm{c},j} - \Lambda_{\mathrm{c},j} \\
&= 1/\sigma^2_{\mathrm{s},j} - 1/\tilde{\sigma}^2_{\mathrm{s},j} - \lambda_{\mathrm{c},j} \\
&= 1/\sigma^2_{\mathrm{s},j} - (1/\sigma^2_{\mathrm{c},j} - \Lambda_{\mathrm{c},j}) - \Lambda_{\mathrm{c},j} \\
&= 1/\sigma^2_{\mathrm{s},j} - 1/\sigma^2_{\mathrm{c},j} \;, \qquad (5.46)
\end{aligned}$$

With this, the rank-one update (5.38) simplifies to

$$\begin{aligned}
\boldsymbol{\Phi}^{\circ}_{\mathrm{c}} &= \boldsymbol{\Phi}_{\mathrm{c}} - \frac{1/\sigma^2_{\mathrm{s},j} - 1/\sigma^2_{\mathrm{c},j}}{1 + (1/\sigma^2_{\mathrm{s},j} - 1/\sigma^2_{\mathrm{c},j})\sigma^2_{\mathrm{c},j}} \boldsymbol{\phi}_{\mathrm{c},j}\boldsymbol{\phi}^{\mathsf{T}}_{\mathrm{c},j} \\
&= \boldsymbol{\Phi}_{\mathrm{c}} - \frac{\sigma^2_{\mathrm{s},j}}{\sigma^2_{\mathrm{c},j}} \left(\frac{1}{\sigma^2_{\mathrm{s},j}} - \frac{1}{\sigma^2_{\mathrm{c},j}}\right) \boldsymbol{\phi}_{\mathrm{c},j}\boldsymbol{\phi}^{\mathsf{T}}_{\mathrm{c},j} \\
&= \boldsymbol{\Phi}_{\mathrm{c}} + \frac{1}{(\sigma^2_{\mathrm{c},j})^2}(\sigma^2_{\mathrm{s},j} - \sigma^2_{\mathrm{c},j})\boldsymbol{\phi}_{\mathrm{c},j}\boldsymbol{\phi}^{\mathsf{T}}_{\mathrm{c},j} \;, \qquad (5.47)
\end{aligned}$$

which yields the interesting interpretation that the covariance matrix is adjusted by the difference of the variances of the two estimators. Basically that means, at position j it is

$$[\boldsymbol{\Phi}^{\circ}_{\mathrm{c}}]_{jj} = \sigma^2_{\mathrm{s},j} \;, \tag{5.48}$$

i.e., the previous variance $\sigma^2_{\mathrm{c},j}$ is replaced by the variance $\sigma^2_{\mathrm{s},j}$ obtained in the signal-constrained estimation and at any other position $j' \neq j$

$$[\boldsymbol{\Phi}^{\circ}_{\mathrm{c}}]_{j'j'} = \sigma^2_{\mathrm{c},j'} + \frac{(\sigma^2_{\mathrm{c},j'})^2}{(\sigma^2_{\mathrm{c},j})^2}(\sigma^2_{\mathrm{s},j} - \sigma^2_{\mathrm{c},j}) \;. \tag{5.49}$$

For the update of the conditional mean, one obtains ($\boldsymbol{m}^{\circ}_{\mathrm{c}} = [m^{\circ}_{\mathrm{c},1}, \ldots, m^{\circ}_{\mathrm{c},N}]^{\mathsf{T}}$)

$$\begin{aligned}
\boldsymbol{m}^{\circ}_{\mathrm{c}} &= \boldsymbol{m}_{\mathrm{c}} + \frac{\sigma^2_{\mathrm{s},j}}{\sigma^2_{\mathrm{c},j}} \left(\Delta\lambda_{\mathrm{c},j} - \Delta\Lambda_{\mathrm{c},j} m_{\mathrm{c},j}\right) \boldsymbol{\phi}_{\mathrm{c},j} \\
&= \boldsymbol{m}_{\mathrm{c}} + \frac{\sigma^2_{\mathrm{s},j}}{\sigma^2_{\mathrm{c},j}} \left(\frac{m_{\mathrm{s},j}}{\sigma^2_{\mathrm{s},j}} - \frac{m_{\mathrm{c},j}}{\sigma^2_{\mathrm{c},j}} - \left(\frac{1}{\sigma^2_{\mathrm{s},j}} - \frac{1}{\sigma^2_{\mathrm{c},j}}\right) m_{\mathrm{c},j}\right) \boldsymbol{\phi}_{\mathrm{c},j} \\
&= \boldsymbol{m}_{\mathrm{c}} + \frac{1}{\sigma^2_{\mathrm{c},j}}(m_{\mathrm{s},j} - m_{\mathrm{c},j})\boldsymbol{\phi}_{\mathrm{c},j} \;, \qquad (5.50)
\end{aligned}$$

i.e., the update is based on the difference between the conditional mean estimates. Here, again $m_{\mathrm{c},j}$ is replaced by $m_{\mathrm{s},j}$ and at other positions $j' \neq j$

$$m_{\mathrm{c},j'}^{\circ} = m_{\mathrm{c},j'} + \frac{\sigma_{\mathrm{c},j'}^2}{\sigma_{\mathrm{c},j}^2}(m_{\mathrm{s},j} - m_{\mathrm{c},j}) \ , \tag{5.51}$$

which means the estimate is updated by the scaled difference between the conditional means of the two opposing estimators, where the scaling is defined by the relation between the variance at the considered position j' and the variance of the position that has been updated.

At this point, the next position needs to be chosen to calculate another signal-constrained estimate and further refine the channel-constrained estimate. The complete algorithm is stated in Appendix E.9 as seqVAMPind. The denomination is chosen because it is the sequential version of VAMPind. The algorithm can be found in [OW05, Appendix D] for a binary prior and in [BMPP20] for a Bernoulli-Gaussian prior.

5.1.5 Sequential Version with Estimation-Theoretic Bias Compensation

In Chapter 4 we have seen that an estimation-theoretic view on the extrinsic calculation can lead to different procedures than derived from the optimization-based view. These updates have been used to compose the VAMPire algorithm, which has been proposed in [FSG20]. Similarly, the respective updates will be used here to obtain a variant of the previously explained algorithm seqVAMPind. The resulting algorithm is called seqVAMPire, as it is a sequential version of the VAMPire algorithm. It was first published in [SF21].

The start of the algorithm is identically to before, i.e., the channel-constrained estimates (5.33) and (5.32) are computed with suitably chosen input parameters $\boldsymbol{\lambda}_{\mathrm{c}}$ and $\boldsymbol{\Lambda}_{\mathrm{c}}$. The ensuing extrinsic computation is equal for estimation-theoretic and optimization-based view, i.e., position j is chosen and (5.41), as well as the signal-constrained estimates as in (5.42) and (5.43) are computed. The estimation-theoretic bias compensation after the signal-constrained estimation differs from the optimization-based extrinsic, since the estimator is non-linear. Instead of the subtraction of natural parameters, the updates (4.86) and (4.86) are used, which are stated here for convenience again

$$\tilde{x}_{\mathrm{c},j} = \tilde{x}_{\mathrm{s},j} + \frac{\tilde{\sigma}_{\mathrm{s},j}^2}{\tilde{\sigma}_{\mathrm{s},j}^2 - \sigma_j^2}(m_{\mathrm{s},j} - \tilde{x}_{\mathrm{s},j}) \ , \tag{5.52}$$

$$\tilde{\sigma}_{\mathrm{c},j}^2 = \sigma_{\mathrm{s},j}^2 + \left(\frac{\sigma_j^2}{\sigma_j^2 - \tilde{\sigma}_{\mathrm{s},j}^2}\right)^2 (m_{\mathrm{s},j} - \tilde{x}_{\mathrm{s},j})^2 \ , \tag{5.53}$$

where $\sigma_j^2 = \mathrm{E}_{\tilde{\mathrm{x}}_{\mathrm{s},j}}\{\sigma_{\mathrm{s},j}^2(\tilde{\mathrm{x}}_{\mathrm{s},j})\}$ is the MSE of the respective prior. Noteworthy, with

$$1/\tilde{\sigma}_j^2 = 1/\sigma_j^2 - 1/\tilde{\sigma}_{\mathrm{s},j}^2 \tag{5.54}$$

the first line can be rewritten to

$$\tilde{x}_{\mathrm{c},j}/\tilde{\sigma}_j^2 = m_{\mathrm{s},j}/\sigma_j^2 - \tilde{x}_{\mathrm{s},j}/\tilde{\sigma}_{\mathrm{s},j}^2 \ . \tag{5.55}$$

One has to be aware, that this differs from the update in (5.44) since the MSE σ_j^2 and its unbiased version $\tilde{\sigma}_j^2$ are used instead of $\sigma_{\mathrm{s},j}^2$ and $\tilde{\sigma}_{\mathrm{c},j}^2$, respectively. Also, $\tilde{\sigma}_{\mathrm{c},j}^2$ is computed differently here. All in all, the updated parameters are obtained as

$$\lambda_{\mathrm{c},j}^{\circ} = \tilde{x}_{\mathrm{c},j}/\tilde{\sigma}_{\mathrm{c},j}^2 = \frac{\tilde{\sigma}_j^2}{\tilde{\sigma}_{\mathrm{c},j}^2}(m_{\mathrm{s},j}/\sigma_j^2 - \tilde{x}_{\mathrm{s},j}/\tilde{\sigma}_{\mathrm{s},j}^2) \ , \tag{5.56}$$

$$\Lambda_{\mathrm{c},j}^{\circ} = 1/\tilde{\sigma}_{\mathrm{s},j}^2 \ , \tag{5.57}$$

and the additive components, thus, as

$$\begin{aligned} \Delta\lambda_{\mathrm{c},j} &= \frac{\tilde{\sigma}_j^2}{\tilde{\sigma}_{\mathrm{c},j}^2}(m_{\mathrm{s},j}/\sigma_j^2 - \tilde{x}_{\mathrm{s},j}/\tilde{\sigma}_{\mathrm{s},j}^2) - \lambda_{\mathrm{c},j} \\ &= \frac{\tilde{\sigma}_j^2}{\tilde{\sigma}_{\mathrm{c},j}^2}(m_{\mathrm{s},j}/\sigma_j^2 - m_{\mathrm{c},j}/\sigma_{\mathrm{c},j}^2) + \lambda_{\mathrm{c},j}\left(\frac{\tilde{\sigma}_j^2}{\tilde{\sigma}_{\mathrm{c},j}^2} - 1\right) \ , \end{aligned} \tag{5.58}$$

$$\Delta\Lambda_{\mathrm{c},j} = 1/\tilde{\sigma}_{\mathrm{c},j}^2 - \Lambda_{\mathrm{c},j} \ . \tag{5.59}$$

These equations have the effect that the rank-one updates (5.38) and (5.40) cannot be simplified to easier scaling operations as in the previous section. Nevertheless, the changes induced by the single signal-constrained estimation can be computed and another position for refinement may be chosen to proceed with the recovery.

The entire procedure is stated in Appendix E.10 as seqVAMPire(sequential version of VAMPire).

So far, we have focused on the steps of updating the changes induced by a single signal-constrained estimation. For the overall procedure of the algorithm, one has to think of the order of processing the variables one after the other, as the sequential approach opens the possibilities for various schedules. The discussion of these opportunities is postponed to Chapter 6.

5.2 Sequential Processing of Observations

In this section, a CS model is considered that slightly differs from the one that is introduced in Chapter 2. So far, it has been assumed that the noise vector $\mathbf{n}$ is i.i.d., which means each element of the vector is distributed in the same way, and especially the noise variance σ_{n}^2 is the same for all elements. In the following, the case of noise with individual variances is considered, i.e., $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{diag}(\sigma_{\mathsf{n},i}^2))$, respectively $\mathsf{n}_i \sim \mathcal{N}(0, \sigma_{\mathsf{n},i}^2)$ for $i \in \{1, \ldots, M\}$, where it is assumed that the individual variances are known. The entries of the noise vector are thereby still independent, but not identically distributed anymore. We call this case the non-i.i.d. noise case. Such a model is justified, if some form of channel equalization, which colors the noise, is performed on the measurements before reconstructing the signal, which happens, e.g., in several PAR reduction schemes for OFDM [KPN+14, LHD20], or when sparse noise is assumed [SSF13].

Since the observations y_i carry different reliabilities in this model, one may benefit from a sequential processing of the observations. This will be examined in the following based on the optimization-based approach that was considered in the previous chapters as well.

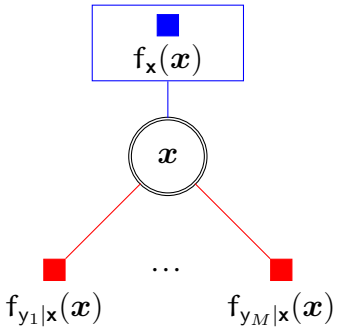


Figure 5.3: Factor graph for sequential processing of the observations.

Sequential processing of observations has been considered in [HTP12, MSW10], where one observation is processed after the other and the reconstruction is stopped if enough observations have been processed. In contrast, here, a particular observation vector $\boldsymbol{y}$ is measured once and the algorithm works on the individual observations of the vector, so that the particular schedule is in the focus. Other work that considered variable noise variances in the CS scenario is given in [BG18], which introduces an algorithm based on AMP. Also the GAMP algorithm [Ran11], which is also stated in Appendix E can cope with the variable noise, if the noise variance σ_{n}^2 is replaced by the respective individual noise variances. As before, the work at hand is based on VAMP instead of AMP.

5.2.1 Structure of the Factor Graph

For the considerations that are pursued here, the channel factors are considered $\mathsf{f}_{\mathsf{y}_i|\mathsf{x}}(\boldsymbol{x})$ independently, since each one should get its own efficacy. In order not to complicate the problem too much—separating the signal components as well, would lead to the MP case of Chapter 3 anyway—the signal-constrained part is kept together. Hence, the factorization of the posterior reads

$$\mathsf{f}_{\mathsf{x}|\mathsf{y}}(\boldsymbol{x}) = \text{const.} \cdot \mathsf{f}_{\mathsf{x}}(\boldsymbol{x}) \cdot \prod_{i=1}^{M} \mathsf{f}_{\mathsf{y}_i|\mathsf{x}}(\boldsymbol{x}) \ , \tag{5.60}$$

and the resulting factor graph is the remaining one in Fig. 2.3, i.e., Fig. 2.3(b), which is plotted in Fig. 5.3 for convenience again.

5.2.1.1 Factorization of the Substitute Distribution

The factorization above tells us that $\mathsf{q}_{\mathsf{s}}(\boldsymbol{x})$ is needed as approximation for $\mathsf{f}_{\mathsf{x}}(\boldsymbol{x})$, as well as the individual $\mathsf{q}_{\mathsf{c},i}(\boldsymbol{x})$ for approximating $\mathsf{f}_{\mathsf{y}_i|\mathsf{x}}(\boldsymbol{x})$ ($i \in \{1, \ldots, M\}$), i.e.,

$$\mathsf{q}_{\mathsf{s}}(\boldsymbol{x}) = \mathsf{f}_{\mathsf{x}}(\boldsymbol{x}) s(\boldsymbol{x}) \ , \tag{5.61}$$

$$\mathsf{q}_{\mathsf{c},i}(\boldsymbol{x}) = \mathsf{f}_{\mathsf{y}_i|\mathsf{x}}(\boldsymbol{x}) c_i(\boldsymbol{x}) \ . \tag{5.62}$$

The (combined) variable node is therefore constituted by $M+1$ instances, which leads to an M-fold integration as in Chapter 3, i.e., the internal behavior of the variable node is given by

$$\frac{\text{const.}}{s(\boldsymbol{x})\prod_{i=1}^{M} c_i(\boldsymbol{x})} = \text{const.} \cdot \int \cdots \int \left(s(\boldsymbol{x}) \prod_{i=1}^{M} c_i(\boldsymbol{x}_i) \right)^{-1} \prod_{i=1}^{M} \delta(\boldsymbol{x} - \boldsymbol{x}_i) \, \mathrm{d}\boldsymbol{x}_1 \ldots \boldsymbol{x}_M \tag{5.63}$$

with slack variables $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_M$ (the argument of $s(\boldsymbol{x})$ is chosen to be the original argument for convenience). So, the connection between variable node and deviations is obtained as

$$\mathsf{q}_\mathrm{v}(\boldsymbol{x}) = \text{const.} \cdot \sqrt[M]{s(\boldsymbol{x}) \prod_{i=1}^{M} c_i(\boldsymbol{x})} \, . \tag{5.64}$$

The resulting factorization reads

$$\mathsf{q}(\boldsymbol{x}) = \frac{\mathsf{q}_\mathrm{s}(\boldsymbol{x}) \cdot \prod_{i=1}^{M} \mathsf{q}_{\mathrm{c},i}(\boldsymbol{x})}{(\mathsf{q}_\mathrm{v}(\boldsymbol{x}))^M} \, . \tag{5.65}$$

5.2.1.2 The Moment-Matching Constraint

As can be seen in the factorization (5.60), respectively the factor graph in Fig. 5.3 above, all factors are connected to the (combined) variable node, which represents the entire vector $\boldsymbol{x}$, in the center. The matching of moments for the sufficient statistics $\boldsymbol{g}(\boldsymbol{x})$ is therefore set up between the representative of the variable node $\mathsf{q}_\mathrm{v}(\boldsymbol{x})$ and the respective representatives of the factors, yielding

$$\mathrm{E}_{\mathsf{x}\sim\mathsf{q}_\mathrm{v}}\{\boldsymbol{g}(\mathsf{x})\} \stackrel{!}{=} \mathrm{E}_{\mathsf{x}\sim\mathsf{q}_\mathrm{s}}\{\boldsymbol{g}(\mathsf{x})\} \, , \tag{5.66}$$

$$\mathrm{E}_{\mathsf{x}\sim\mathsf{q}_\mathrm{v}}\{\boldsymbol{g}(\mathsf{x})\} \stackrel{!}{=} \mathrm{E}_{\mathsf{x}\sim\mathsf{q}_{\mathrm{c},i}}\{\boldsymbol{g}(\mathsf{x})\} \, , \qquad i \in \{1, \ldots, M\} \, . \tag{5.67}$$

Here, the expectation constraint is vector-wise, i.e., the focus is not on individual signal components x_j. Therefore it is possible to restrict to the average variance case, i.e., $\boldsymbol{g}(\boldsymbol{x}) = [\boldsymbol{x}^\top, -\frac{1}{2}\boldsymbol{x}^\top\boldsymbol{x}]^\top$ from Ex. D.1 and (D.12), which means the reliability of the entire vector is represented with a single variance.

5.2.2 Constrained Optimization

The constrained optimization problem that is obtained, when combining minimization of the Kullback–Leibler divergence with the moment-matching constraints above (and additional normalization constraints for valid pdfs) reads

$$\begin{aligned}
\text{minimize} \quad & \mathrm{D}_\mathrm{KL}\Big(\mathsf{q}(\boldsymbol{x}) \,||\, \mathsf{f}_{\mathsf{x}|\mathsf{y}}(\boldsymbol{x})\Big) \\
\text{subject to} \quad & \mathrm{E}_{\mathsf{x}\sim\mathsf{q}_\mathrm{v}}\{\boldsymbol{g}(\mathsf{x})\} - \mathrm{E}_{\mathsf{x}\sim\mathsf{q}_{\mathrm{c},i}}\{\boldsymbol{g}(\mathsf{x})\} = \boldsymbol{0} \, , \qquad i \in \{1, \ldots, M\} \, , \\
& \mathrm{E}_{\mathsf{x}\sim\mathsf{q}_\mathrm{v}}\{\boldsymbol{g}(\mathsf{x})\} - \mathrm{E}_{\mathsf{x}\sim\mathsf{q}_\mathrm{s}}\{\boldsymbol{g}(\mathsf{x})\} = \boldsymbol{0} \, , \\
& \int \mathsf{q}_\mathrm{v}(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} = 1 \, , \quad \int \mathsf{q}_\mathrm{s}(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} = 1 \, , \\
& \int \mathsf{q}_{\mathrm{c},i}(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} = 1 \, , \qquad i \in \{1, \ldots, M\} \, .
\end{aligned} \tag{5.68}$$

5.2.2.1 Cost Function for Compressed Sensing

The corresponding Kullback–Leibler divergence between substitute distribution $\mathsf{q}(\boldsymbol{x})$ and posterior $\mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})$, which needs to be minimized, is here given by

$$\begin{aligned}\mathrm{D}_{\mathrm{KL}}\Big(\mathsf{q}(\boldsymbol{x}) \,||\, \mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})\Big) = &\int \mathsf{q}_{\mathrm{s}}(\boldsymbol{x}) \log \mathsf{q}_{\mathrm{s}}(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} + \sum_{i=1}^{M} \int \mathsf{q}_{\mathrm{c},i}(\boldsymbol{x}) \log \mathsf{q}_{\mathrm{c},i}(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} \\ &- \int \mathsf{q}_{\mathrm{s}}(\boldsymbol{x}) \log \mathsf{f}_{\mathbf{x}}(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} - \sum_{i=1}^{M} \int \mathsf{q}_{\mathrm{c},i}(\boldsymbol{x}) \log \mathsf{f}_{\mathrm{y}_i|\mathbf{x}}(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} \\ &- M \int \mathsf{q}_{\mathrm{v}}(\boldsymbol{x}) \log \mathsf{q}_{\mathrm{v}}(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} \,. \end{aligned} \tag{5.69}$$

5.2.2.2 Stationary Points of the Lagrangian

In order to find the basic relations that a solution to the CS problem with the given structural conditions need to fulfill, the stationary points of the respective Lagrangian are computed. To that end, the Lagrangian multipliers $\boldsymbol{\nu}_{\mathrm{c},i}$ and $\boldsymbol{\nu}_{\mathrm{s}}$ are defined for the moment-matching constraints, as well as ν_{v}, ν_{s}, and $\nu_{\mathrm{c},i}$ for the normalization constraints, respectively. Then, the Lagrangian reads

$$\begin{aligned}\mathcal{L}(\mathsf{q}(\boldsymbol{x})) = &\,\mathrm{D}_{\mathrm{KL}}\Big(\mathsf{q}(\boldsymbol{x}) \,||\, \mathsf{f}_{\mathbf{x}|\mathbf{y}}(\boldsymbol{x})\Big) + \sum_{i=1}^{M} \boldsymbol{\nu}_{\mathrm{c},i}^{\top} \Big(\mathrm{E}_{\mathbf{x}\sim\mathsf{q}_{\mathrm{v}}}\{\boldsymbol{g}(\mathbf{x})\} - \mathrm{E}_{\mathbf{x}\sim\mathsf{q}_{\mathrm{c},i}}\{\boldsymbol{g}(\mathbf{x})\}\Big) \\ &+ \boldsymbol{\nu}_{\mathrm{s}}^{\top} \left(\mathrm{E}_{\mathbf{x}\sim\mathsf{q}_{\mathrm{v}}}\{\boldsymbol{g}(\mathbf{x})\} - \mathrm{E}_{\mathbf{x}\sim\mathsf{q}_{\mathrm{s}}}\{\boldsymbol{g}(\mathbf{x})\}\right) \\ &+ \sum_{j=1}^{N} \left(\nu_{\mathrm{v}}\Big(\int \mathsf{q}_{\mathrm{v}}(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} - 1\Big) + \nu_{\mathrm{s}}\Big(\int \mathsf{q}_{\mathrm{s}}(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} - 1\Big)\right) \\ &+ \sum_{i=1}^{M} \nu_{\mathrm{c},i}\Big(\int \mathsf{q}_{\mathrm{c},i}(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} - 1\Big) \,. \end{aligned} \tag{5.70}$$

As usual, the stationary points are obtained by differentiating w.r.t. the representatives $\mathsf{q}_{\mathrm{v}}(\boldsymbol{x})$, $\mathsf{q}_{\mathrm{c},i}(\boldsymbol{x})$, and $\mathsf{q}_{\mathrm{s}}(\boldsymbol{x})$, respectively, and setting the result to zero, which yields

$$\frac{\partial}{\partial \mathsf{q}_{\mathrm{v}}} \mathcal{L}(\mathsf{q}(\boldsymbol{x})) = -M \log \mathsf{q}_{\mathrm{v}}(\boldsymbol{x}) + \left(\boldsymbol{\nu}_{\mathrm{s}} + \sum_{i=1}^{M} \boldsymbol{\nu}_{\mathrm{c},i}\right)^{\top} \boldsymbol{g}(\boldsymbol{x}) + \mathrm{const.} \stackrel{!}{=} 0 \,, \tag{5.71}$$

$$\frac{\partial}{\partial \mathsf{q}_{\mathrm{c},i}} \mathcal{L}(\mathsf{q}(\boldsymbol{x})) = \log \mathsf{q}_{\mathrm{c},i}(\boldsymbol{x}) - \log \mathsf{f}_{\mathrm{y}_i|\mathbf{x}}(\boldsymbol{x}) - \boldsymbol{\nu}_{\mathrm{c},i}^{\top} \boldsymbol{g}(\boldsymbol{x}) + \mathrm{const.} \stackrel{!}{=} 0 \,, \tag{5.72}$$

$$\frac{\partial}{\partial \mathsf{q}_{\mathrm{s}}} \mathcal{L}(\mathsf{q}(\boldsymbol{x})) = \log \mathsf{q}_{\mathrm{s}}(\boldsymbol{x}) - \log \mathsf{f}_{\mathbf{x}}(\boldsymbol{x}) - \boldsymbol{\nu}_{\mathrm{s}}^{\top} \boldsymbol{g}(\boldsymbol{x}) + \mathrm{const.} \stackrel{!}{=} 0 \,. \tag{5.73}$$

With Z_{v}, $Z_{\mathrm{c},i}$, and Z_{s} functioning as normalizing factors, the stationary-point forms for the

representatives read

$$\mathsf{q}_{\mathrm{v}}(\boldsymbol{x}) = \frac{1}{Z_{\mathrm{v}}} \exp\left(\frac{1}{M} (\boldsymbol{\nu}_{\mathrm{s}} + \sum_{i=1}^{M} \boldsymbol{\nu}_{\mathrm{c},i})^{\top} \boldsymbol{g}(\boldsymbol{x}) \right) \ , \tag{5.74}$$

$$\mathsf{q}_{\mathrm{c},i}(\boldsymbol{x}) = \frac{1}{Z_{\mathrm{c},i}} \mathsf{f}_{\mathsf{y}_i|\mathsf{x}}(\boldsymbol{x}) \exp\left(\boldsymbol{\nu}_{\mathrm{c},i}^{\top} \boldsymbol{g}(\boldsymbol{x}) \right) \ , \tag{5.75}$$

$$\mathsf{q}_{\mathrm{s}}(\boldsymbol{x}) = \frac{1}{Z_{\mathrm{s}}} \mathsf{f}_{\mathsf{x}}(\boldsymbol{x}) \exp\left(\boldsymbol{\nu}_{\mathrm{s}}^{\top} \boldsymbol{g}(\boldsymbol{x}) \right) \ . \tag{5.76}$$

Again, as expected, exponential families are involved.

The respective natural parameter of the (combined) variable node representative $\mathsf{q}_{\mathrm{v}}(\boldsymbol{x})$ is given by

$$\boldsymbol{\theta} \overset{\text{def}}{=} \frac{1}{M} \left(\boldsymbol{\nu}_{\mathrm{s}} + \sum_{i=1}^{M} \boldsymbol{\nu}_{\mathrm{c},i} \right) \ . \tag{5.77}$$

This reminds us of the connection (3.25) of Chapter 3 and will be used in the following to derive the parameter update for the recovery algorithm tailored to the structure given by the factor graph considered here. The correspondences between edges and the respective edge-dependent natural parameters are depicted in Fig. 5.4(a).

5.2.3 Algorithm

Based on the stationary points of the given structure, an algorithm for the recovery of signal $\boldsymbol{x}$ is derived in the following.

5.2.3.1 Expectations

First, the computations at the factor nodes are regarded, which are expectations with input parameters $\boldsymbol{\nu}_{\mathrm{c},i}$ ($i \in \{1, \ldots, M\}$) and $\boldsymbol{\nu}_{\mathrm{s}}$, respectively. Since only the average variance case is considered and the estimates of all signal vector elements are computed at once, one can (as in Sec. 4.3.1) connect the signal-constrained estimation at factor $\mathsf{f}_{\mathsf{x}}(\boldsymbol{x})$ to (2.19) and (2.20) by making the identification $\boldsymbol{\nu}_{\mathrm{s}} = [\tilde{\boldsymbol{x}}_{\mathrm{s}}^{\top}/\tilde{\sigma}_{\mathrm{s}}^2, \ 1/\tilde{\sigma}_{\mathrm{s}}^2]^{\top}$, i.e.,

$$\mathrm{E}_{\mathsf{x} \sim \mathsf{q}_{\mathrm{s}}} \{ \boldsymbol{g}_{\boldsymbol{\lambda}}(\mathsf{x}) \} = \boldsymbol{m}_{\mathrm{s}} \ , \tag{5.78}$$

$$\mathrm{E}_{\mathsf{x} \sim \mathsf{q}_{\mathrm{s}}} \{ g_{\Lambda}(\mathsf{x}) \} = -\frac{1}{2} \sum_{j=1}^{N} (\sigma_{\mathrm{s},j}^2 + m_{\mathrm{s},j}^2) \ . \tag{5.79}$$

The corresponding average variance σ_{s}^2 is obtained as in (4.35), i.e.,

$$\sigma_{\mathrm{s}}^2 = \frac{1}{N} \sum_{j=1}^{N} \sigma_{\mathrm{s},j}^2 = -\frac{2}{N} \mathrm{E}_{\mathsf{x} \sim \mathsf{q}_{\mathrm{s}}} \{ g_{\Lambda}(\mathsf{x}) \} - \boldsymbol{m}_{\mathrm{s}}^{\top} \boldsymbol{m}_{\mathrm{s}} \ . \tag{5.80}$$

The expectations at channel factors $\mathsf{f}_{\mathsf{y}_i|\mathsf{x}}(\boldsymbol{x})$ can be related to the estimates in Sec. 2.2.1.2 by considering the average variance case, i.e., $\boldsymbol{\Phi}_{\mathsf{x}} = \sigma_{\mathsf{x}}^2 \boldsymbol{I}_N$, and making the identification

$\boldsymbol{\nu}_{\mathrm{c},i} = [\boldsymbol{m}_{\mathsf{x}}^\top/\sigma_{\mathsf{x}}^2,\, 1/\sigma_{\mathsf{x}}^2]^\top$. The estimation reads then

$$\begin{aligned}\mathrm{E}_{\mathbf{x}\sim\mathrm{q}_{\mathrm{c},i}}\{\boldsymbol{g}_{\boldsymbol{\lambda}}(\mathbf{x})\} &= \boldsymbol{m}_{\mathrm{c},i} = \boldsymbol{m}_{\mathsf{x}} + \sigma_{\mathsf{x}}^2\boldsymbol{a}_i(\sigma_{\mathsf{x}}^2\boldsymbol{a}_i^\top\boldsymbol{a}_i + \sigma_{\mathsf{n},i}^2)^{-1}(y_i - \boldsymbol{a}_i^\top\boldsymbol{m}_{\mathsf{x}}) \\ &= \boldsymbol{m}_{\mathsf{x}} + \frac{y_i - \sum_{j=1}^N a_{ij}m_{\mathsf{x},j}}{\sigma_{\mathsf{n},i}^2/\sigma_{\mathsf{x}}^2 + \sum_{j=1}^N a_{ij}^2}\boldsymbol{a}_i\ , \qquad (5.81)\\ \mathrm{E}_{\mathbf{x}\sim\mathrm{q}_{\mathrm{c},i}}\{g_A(\mathbf{x})\} &= -\frac{1}{2}\mathrm{trace}\Big(\boldsymbol{\Phi}_{\mathrm{c},i} + \boldsymbol{m}_{\mathrm{c},i}\boldsymbol{m}_{\mathrm{c},i}^\top\Big) \\ &= -\frac{1}{2}\mathrm{trace}\left(\sigma_{\mathsf{n},i}^2(\boldsymbol{a}_i\boldsymbol{a}_i^\top + \frac{\sigma_{\mathsf{n},i}^2}{\sigma_{\mathsf{x}}^2}\boldsymbol{I}_N)^{-1} + \boldsymbol{m}_{\mathrm{c},i}\boldsymbol{m}_{\mathrm{c},i}^\top\right)\ . \qquad (5.82)\end{aligned}$$

Furthermore, the resulting average variance of the estimate $\boldsymbol{m}_{\mathrm{c},i}$ reads

$$\sigma_{\mathrm{c},i}^2 = \frac{1}{N}\mathrm{trace}(\boldsymbol{\Phi}_{\mathrm{c},i}) = \frac{\sigma_{\mathsf{n},i}^2}{N}\mathrm{trace}\left((\boldsymbol{a}_i\boldsymbol{a}_i^\top + \frac{\sigma_{\mathsf{n},i}^2}{\sigma_{\mathsf{x}}^2}\boldsymbol{I}_N)^{-1}\right)\ . \qquad (5.83)$$

5.2.3.2 Projection to Exponential Family

The moment-matching between the variable node and the respective factor node, which shall ensure consistency, means that the computed moments of the section above have to represent the variable node distribution. Therefore, the natural parameter of $\mathrm{q}_{\mathrm{v}}(\boldsymbol{x})$ can directly be obtained from these moments via

$$\boldsymbol{\theta} = [\boldsymbol{m}_{\mathrm{s}}^\top/\sigma_{\mathrm{s}}^2,\, 1/\sigma_{\mathrm{s}}^2]^\top\ , \qquad (5.84)$$

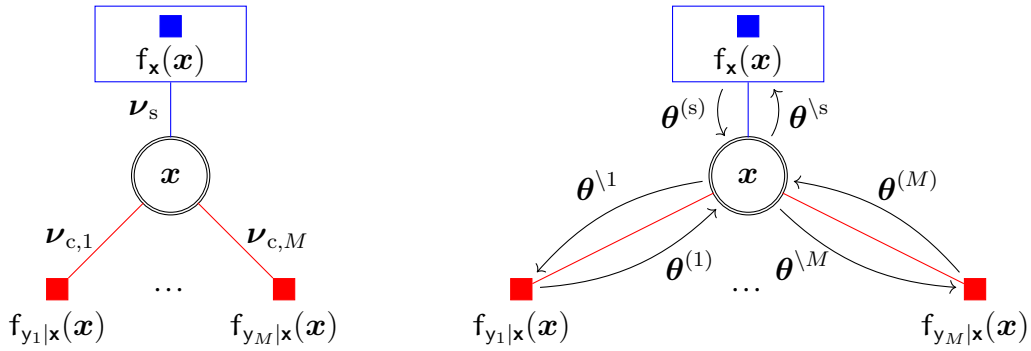
respectively,

$$\boldsymbol{\theta} = [\boldsymbol{m}_{\mathrm{c},i}^\top/\sigma_{\mathrm{c},i}^2,\, 1/\sigma_{\mathrm{c},i}^2]^\top\ , \quad i \in \{1,\,\ldots,\,M\}\ . \qquad (5.85)$$

For the further processing, one needs to consider the restrictions on the natural parameters induced by the structure of the problem, which finds its expression in the natural parameter (5.77) of the representative $\mathrm{q}_{\mathrm{v}}(\boldsymbol{x})$ of the variable node.

5.2.3.3 Extrinsic

In order to make use of the connection (5.77) for the processing of (natural) parameters in the algorithm, recall that the basic idea for the iterative schemes aiming towards stationary points is the notion of extrinsic, i.e., only knowledge obtained via the processing is submitted for further processing (in order to prevent feedback loops). Compared to in Chapter 3, the derivation is reversed, i.e., it starts with the processing that sums up all incoming edges except the one that the message is going to be sent on, to see that in the end, the same connections hold. To apply the notion here, the Lagrangian multipliers are interpreted as messages on the corresponding edges and a distinction between the messages in different directions is made. As the multipliers itself specify the representative of the variable node $\mathrm{q}_{\mathrm{v}}(\boldsymbol{x})$ entirely, they are interpreted as *outgoing* messages of the variable node and denoted by $\boldsymbol{\theta}^{\backslash\mathrm{s}} = \boldsymbol{\nu}_{\mathrm{s}}$ and $\boldsymbol{\theta}^{\backslash i} = \boldsymbol{\nu}_{\mathrm{c},i}$ ($i \in \{1,\,\ldots,\,M\}$), respectively. The messages on the backward direction, from the



(a) Depiction of edge-dependent multipliers.

(b) Depiction of edge- and direction-dependent messages.

Figure 5.4: Factor graph for the sequential processing of observations and edge-dependent parameters.

factor nodes towards the variable node will be denoted by $\boldsymbol{\theta}^{(\mathrm{s})}$ and $\boldsymbol{\theta}^{(i)}$ ($i \in \{1, \ldots, M\}$), respectively. The direction-dependent messages are shown on the respective edges in the factor graph in Fig. 5.4(b). The notation is chosen in order to show the connection to the expectation propagation (EP) [Min01] framework, where the idea is to replace parts of the approximations by the respective exact factor, in order to refine the estimation by the knowledge obtained by the factor. In the EP literature, the variables $\boldsymbol{\theta}^{\backslash \bullet}$ are called the *cavity* parameters and $\boldsymbol{\theta}^{(\bullet)}$ are the so-called *site* parameters.

The notion of extrinsic can now be established by ensuring that an outgoing message from variable node to factor node is only specified by incoming messages from factor nodes other than the one to send to, i.e.,

$$\boldsymbol{\theta}^{\backslash \mathrm{s}} = \sum_{i=1}^{M} \boldsymbol{\theta}^{(i)} \,, \tag{5.86}$$

$$\boldsymbol{\theta}^{\backslash i} = \boldsymbol{\theta}^{(\mathrm{s})} + \sum_{\substack{i'=1 \\ i' \neq i}}^{M} \boldsymbol{\theta}^{(i')} \,, \quad i \in \{1, \ldots, M\} \,. \tag{5.87}$$

By inserting this into the connection (5.77), one finds

$$\begin{aligned} \boldsymbol{\theta} &= \frac{1}{M} \left(\sum_{i=1}^{M} \boldsymbol{\theta}^{(i)} + \sum_{i=1}^{M} \left(\boldsymbol{\theta}^{(\mathrm{s})} + \sum_{\substack{i'=1 \\ i' \neq i}}^{M} \boldsymbol{\theta}^{(i')} \right) \right) \\ &= \frac{1}{M} \sum_{i=1}^{M} \boldsymbol{\theta}^{(i)} + \boldsymbol{\theta}^{(\mathrm{s})} + \frac{1}{M}(M-1) \sum_{i=1}^{M} \boldsymbol{\theta}^{(i)} \\ &= \boldsymbol{\theta}^{(\mathrm{s})} + \sum_{i=1}^{M} \boldsymbol{\theta}^{(i)} \,. \end{aligned} \tag{5.88}$$

A comparison to the definitions for the outgoing messages of the variable node, which are also the incoming messages for the factor nodes, shows that the inputs for the estimations at the

factors can be obtained from the natural parameter $\boldsymbol{\theta}$ of the variable node by

$$\boldsymbol{\theta}^{\backslash \mathrm{s}} = \boldsymbol{\theta} - \boldsymbol{\theta}^{(\mathrm{s})} \,, \tag{5.89}$$

$$\boldsymbol{\theta}^{\backslash i} = \boldsymbol{\theta} - \boldsymbol{\theta}^{(i)} \,, \quad i \in \{1, \ldots, M\} \,, \tag{5.90}$$

respectively, i.e., *post-processed* extrinsic calculations are at hand and the sum of messages in different directions yields, as in Chapter 3, the knowledge about the signal. This means that in order to be able to get an input to the expectations, it is necessary to store and keep track of the site parameters $\boldsymbol{\theta}^{(\bullet)}$, which can be updated after the estimation via $\boldsymbol{\theta}^{(\bullet)} = \boldsymbol{\theta} - \boldsymbol{\theta}^{\backslash \bullet}$ ($\bullet \in \{\mathrm{s}, 1, \ldots, M\}$). With this knowledge, it is possible to state the entire procedure.

5.2.3.4 Procedure

When starting the algorithm with expectations, first suitable input parameters $\boldsymbol{\theta}^{\backslash \bullet}$ are required for all factors. For the simplification of notation, we introduce $\boldsymbol{\theta}^{\backslash \bullet} = [(\tilde{\boldsymbol{x}}^{\backslash \bullet})^\top / \sigma^2_{\backslash \bullet}, \, 1/\sigma^2_{\backslash \bullet}]^\top$, and $\boldsymbol{\theta}^{(\bullet)} = [(\tilde{\boldsymbol{x}}^{(\bullet)})^\top / \sigma^2_{(\bullet)}, \, 1/\sigma^2_{(\bullet)}]^\top$ ($\bullet \in \{\mathrm{s}, 1, \ldots, M\}$) and use $\boldsymbol{\theta} = [\boldsymbol{m}^\top/\sigma^2, \, 1/\sigma^2]^\top$ to represent the moments that are mapped to the natural parameters of the variable node. The computation of the signal-constrained estimation is performed as usual,

$$\boldsymbol{m} = \mathrm{E}_{\mathsf{x}}\Big\{ \mathsf{x} \mid \tilde{\boldsymbol{x}}^{\backslash \mathrm{s}}, \, \sigma^2_{\backslash \mathrm{s}} \Big\} \,, \tag{5.91}$$

$$\sigma^2 = \frac{1}{N} \sum_{j=1}^{N} \mathrm{E}_{\mathsf{x}}\Big\{ (\mathsf{x} - m_j)^2 \mid \tilde{x}_j^{\backslash \mathrm{s}}, \, \sigma^2_{\backslash \mathrm{s}} \Big\} \,, \tag{5.92}$$

and the site parameters are obtained by

$$\tilde{\boldsymbol{x}}^{(\mathrm{s})}/\sigma^2_{(\mathrm{s})} = \boldsymbol{m}/\sigma^2 - \tilde{\boldsymbol{x}}^{\backslash \mathrm{s}}/\sigma^2_{\backslash \mathrm{s}} \,, \qquad 1/\sigma^2_{(\mathrm{s})} = 1/\sigma^2 - 1/\sigma^2_{\backslash \mathrm{s}} \,. \tag{5.93}$$

For the channel-constrained estimations, one can spare the computation of the $N \times N$ inverse for the covariance matrix by using the knowledge about bias compensation from Sec. 4.4.1. Since average unbiasing is performed here, one can compute the unbiased estimate directly by computing the scaling factor based on the end-to-end cascade between $\boldsymbol{x}$ and $\boldsymbol{m}_{\mathrm{c},i}$ via

$$w = \frac{1}{N} \mathrm{trace}\Big(\boldsymbol{a}_i (\boldsymbol{a}_i^\top \boldsymbol{a}_i + \sigma^2_{\mathsf{n},i}/\sigma^2_{\backslash i})^{-1} \boldsymbol{a}_i^\top \Big) = \frac{1}{N} \frac{\sum_{j=1}^{N} a_{ij}^2}{\sigma^2_{\mathsf{n},i}/\sigma^2_{\backslash i} + \sum_{j=1}^{N} a_{ij}^2} \,, \tag{5.94}$$

which involves as visible only a scalar inverse. The unbiased a.k.a. site estimates then read

$$\tilde{\boldsymbol{x}}^{(i)} = \tilde{\boldsymbol{x}}^{\backslash i} + \frac{\sigma^2_{\backslash i}}{w} \boldsymbol{a}_i (\sigma^2_{\backslash i} \boldsymbol{a}_i^\top \boldsymbol{a}_i + \sigma^2_{\mathsf{n},i})^{-1} (y_i - \boldsymbol{a}_i^\top \tilde{\boldsymbol{x}}^{\backslash i}) \,, \tag{5.95}$$

$$\sigma^2_{(i)} = \sigma^2_{\backslash i} (1/w - 1) \,, \tag{5.96}$$

and the biased estimates can be retrieved by

$$\sigma^2 = (1/\sigma^2_{(i)} + 1/\sigma^2_{\backslash i})^{-1} \,, \tag{5.97}$$

$$\boldsymbol{m} = \sigma^2 (\tilde{\boldsymbol{x}}^{(i)}/\sigma^2_{(i)} + \tilde{\boldsymbol{x}}^{\backslash i}/\sigma^2_{\backslash i}) \,. \tag{5.98}$$

Noteworthy, the estimates $\boldsymbol{m}$ and σ^2 need to proceed with the algorithm at the next factor and the site parameters $\tilde{\boldsymbol{x}}^{(i)}$ and $\sigma^2_{(i)}$ must be stored for when the factor $\mathsf{f}_{\mathsf{y}_i|\mathsf{x}}(\boldsymbol{x})$ is addressed the next time. Whenever a factor is accessed again, one can simply compute $\boldsymbol{\theta}^{\backslash\bullet} = \boldsymbol{\theta} - \boldsymbol{\theta}^{(\bullet)}$ ($\bullet \in \{\mathrm{s}, 1, \ldots, M\}$) to obtain the corresponding input parameter. The entire algorithm is called seqVAMPobsand stated in Appendix E.11.

6. Numerical Results

This chapter comprises the results of numerical simulations that compare the algorithms presented in this thesis in different scenarios. Before the simulation results are shown, an overview over the algorithms that were presented and whose results will be shown in the following is given.

6.1 Overview over Recovery Algorithms

The algorithms that are presented in this thesis are summarized and categorized according to their derivations in Fig. 6.1. The first two columns show the different approaches in terms of factorization of the factor graph, which leads to a separation into message-passing-type, Turbo-type, and sequential algorithms. The last two columns specify, whether the algorithm is directly obtained from the processing in the factor graph, or if it has been adjusted by an estimation-theoretic bias compensation strategy. The arrows indicate which algorithms are related to each other. The background color highlights algorithms utilizing individual variances.

The message-passing-type algorithms that are considered here are Gaussian message passing (GMP), GMPpostExt, which uses post-processed extrinsic, instead of pre-processed extrinsic, and GMPpostExtLin, which is a mixture of both with post-processed extrinsic only at the linear (channel-constrained) estimation.

For the Turbo algorithms, there are two starting points: VAMP, which utilizes average variances as reliabilities and VAMPind with individual variances. The algorithms VAMPia, and VAMPii are obtained from VAMP by employing individual bias compensation after the channel-constrained estimation (instead of average bias compensation). The difference between VAMPia and VAMPii is that VAMPia uses an average variance in the signal-constrained estimation, whereas VAMPii keeps the individual ones.

The algorithm VAMPire results from VAMPind by inserting the estimation-theoretic bias compensation after the signal-constrained estimation.

seqVAMPind and seqVAMPire can be seen as sequential versions (sequentially process-

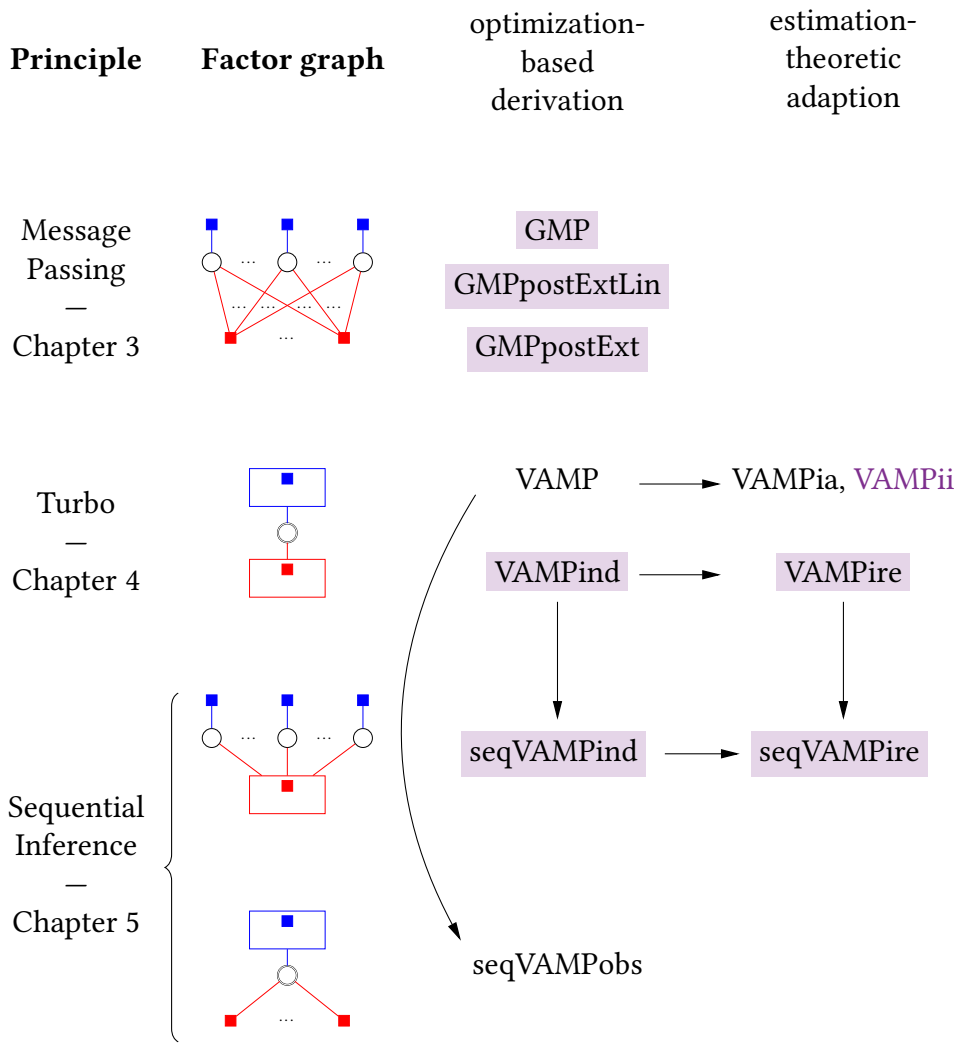


Figure 6.1: Categorization of the algorithms presented in this thesis. The arrows indicate which algorithms are related to each other; the background color highlights algorithms utilizing individual variances. VAMPii uses individual variances only in the signal-constrained estimation and is therefore highlighted differently.

Table 6.1: References for the algorithms used in the simulations.

Algorithm	Main Part	Appendix	no.	Reference(s)
AMP	Sec. 3.3.5	Sec. E.3	Alg. E.3	[DMM09, Mal11]
GAMP	Sec. 3.3.5	Sec. E.4	Alg. E.4	[Ran11]
GMP	Sec. 3.3.4	Sec. E.1	Alg. E.1	[KMS+12]
GMPpostExt	Sec. 3.3.5	Sec. E.2	Alg. E.2	
GMPpostExtLin	Sec. 3.3.5			
seqVAMPind	Sec. 5.1.4	Sec. E.9	Alg. E.10	[OW05, BMPP20]
seqVAMPire	Sec. 5.1.5	Sec. E.10	Alg. E.11	[SF21]
seqVAMPobs	Sec. 5.2	Sec. E.11	Alg. E.12	
VAMP	Sec. 4.3.1	Sec. E.5	Alg. E.5	[RSF19]
VAMPia	Sec. 4.5.2	Sec. E.8	Alg. E.8	[SF22]
VAMPii	Sec. 4.5.2	Sec. E.8	Alg. E.9	[SF22]
VAMPind	Sec. 4.3.2	Sec. E.6	Alg. E.6	[FSRS16]
VAMPire	Sec. 4.5.1	Sec. E.7	Alg. E.7	[FSG20]

ing variables) of VAMPind, respectively VAMPire, because this sequential processing requires individual variances (so it is not a direct descendant from VAMP).

However, the algorithm that processes observations sequentially (seqVAMPobs) utilizes average variances and can therefore be interpreted as a sequential version (in terms of processing observations) of VAMP.

Additionally, all the algorithms used in the simulations are listed in Table 6.1 with references to the respective sections in the main part of the thesis, as well as section and algorithm number in the appendix. If the algorithms were published previous to this thesis, the reference is also given in the last column.

6.2 Simulation Setup

For the simulations, 10^5 realizations of sensing matrices $\boldsymbol{A}$, as well as signal vectors $\boldsymbol{x}$ obeying the pdf $\mathsf{f}_{\mathsf{x}}(\boldsymbol{x})$, are drawn. If not stated otherwise, the elements of the sensing matrices are drawn i.i.d. Gaussian and the columns are normalized to unit ℓ_2-norm, i.e., $\|\tilde{\boldsymbol{a}}_j\|_2^2 = 1$ for $j \in \{1, \ldots, N\}$. As signal model, the discrete ternary (DT) prior from (2.4) with non-zero signal alphabet $\mathcal{X} \in \{-1, +1\}$ is used. The Bernoulli-Gaussian prior is not considered, because in engineering applications often a specific (discrete) signal model is given [Pro00, SF14].

The computations for the signal-constrained estimation in case of the DT prior are given as follows, cf. also [SF16, Spa19]. Defining the input variables $\lambda = \tilde{x}_{\mathrm{s},j}/\tilde{\sigma}_{\mathrm{s},j}^2$ and $\Lambda = 1/\tilde{\sigma}_{\mathrm{s},j}^2$, the estimates read

$$m_{\mathrm{s},j} = \frac{\exp(\lambda - \Lambda/2) - \exp(-\lambda - \Lambda/2)}{\exp(\lambda - \Lambda/2) + \exp(-\lambda - \Lambda/2) + \frac{2(N-s)}{s}} \,, \tag{6.1}$$

$$\mathrm{E}_{\mathsf{x}_j}\left\{\mathsf{x}_j^2 \mid \tilde{x}_{\mathrm{s},j}\right\} = \frac{\exp(\lambda - \Lambda/2) + \exp(-\lambda - \Lambda/2)}{\exp(\lambda - \Lambda/2) + \exp(-\lambda - \Lambda/2) + \frac{2(N-s)}{s}} \,, \tag{6.2}$$

$$\sigma_{\mathrm{s},j}^2 = \mathrm{E}_{\mathsf{x}_j}\left\{\mathsf{x}_j^2 \mid \tilde{x}_{\mathrm{s},j}\right\} - m_{\mathrm{s},j}^2 \,. \tag{6.3}$$

Note that this implementation is numerically more stable than the one used in [SF16], because the assembly of the two parameters λ and Λ in the exponential functions have a neutralizing effect, which makes the terms being computable for a wide range of values. If numerical instabilities occur, this makes itself noticeable already at the computation of $m_{\mathrm{s},j}$. Using the implementation above this only occurs, for very small variances $\tilde{\sigma}^2_{\mathrm{s},j}$ when $m_{\mathrm{s},j} \in \{+1, -1\}$ should actually be the result. In this case, the result can be obtained by setting $m_{\mathrm{s},j} = \mathrm{sgn}(\lambda)$, where $\mathrm{sgn}(\cdot)$ is the signum-function, i.e.,

$$m_{\mathrm{s},j} = \mathrm{sgn}(\lambda) \stackrel{\text{def}}{=} \begin{cases} +1\,, & \text{if } \lambda > 0\,, \\ 0\,, & \text{if } \lambda = 0\,, \\ -1\,, & \text{if } \lambda < 0\,, \end{cases}\,, \tag{6.4}$$

and the variance $\sigma^2_{\mathrm{s},j}$ to an arbitrary small positive value.

For a detailed explanation of the construction of the estimators for other priors, the reader may be referred to [Bir19].

Since the signal amplitude of the DT prior only attains discrete values, the estimates resulting from the algorithms are quantized before evaluation. The model (2.4) assumes to know the sparsity s of the signal, indeed the signal vectors are drawn such that they entail sparsity s. Therefore, this knowledge is applied by setting the smallest (in magnitude) $N - s$ values in the final estimate $\boldsymbol{m}_{\mathrm{s}}$ to zero. Other methods are discussed in [Spa19]. Subsequently, the remaining s entries of $\boldsymbol{m}_{\mathrm{s}}$ are quantized to $\mathcal{X}$. Here, this can easily be done by utilizing the signum-function, i.e., $\hat{m}_{\mathrm{s},j} = \mathrm{sgn}(m_{\mathrm{s},j})$ for $j \in \{1, \ldots, N\}$.

The noise vector is drawn i.i.d. zero-mean Gaussian with variance σ^2_{n}, which results in a signal-to-noise ratio (SNR) defined as $10 \log_{10}(1/\sigma^2_{\mathsf{n}})$ in decibel (dB).

The performance is evaluated after a fixed number of iterations. As performance measure the symbol error rate (SER) is averaged over the realizations. The SER per realization is measured as

$$\mathrm{SER}(\hat{\boldsymbol{m}}_{\mathrm{s}}, \boldsymbol{x}) = \frac{1}{N} |\{\hat{m}_{\mathrm{s},j} \neq x_j \mid j \in \{1, \ldots, N\}\}|\,, \tag{6.5}$$

where $|\cdot|$ denotes the cardinality of a set. If not stated otherwise, the performance is evaluated after 20 iterations. As pointed out in [RSF19] and [FSG20], the occurrence of negative variances in any of the recovery algorithm causes a drastic deterioration in performance, since a negative variance does not have a suitable meaning. Hence, this has to be mitigated. As suggested in [RSF19], the precisions, a.k.a. inverse variances are clipped to a suitable interval. The result of the signal-constrained estimation is clipped stricter than other variances, i.e., $1/\sigma^2_{\mathrm{s},j} \in [10^{-8}, 10^{8}]$ and $1/\sigma^2_{\mathrm{c},j}, 1/\tilde{\sigma}^2_{\mathrm{c},j}, 1/\tilde{\sigma}^2_{\mathrm{s},j} \in [10^{-12}, 10^{12}]$, since this resulted in stable simulations for a wide range of parameters. Note that clipping the precisions (instead of the variances) results in negative variances being transferred to the upper end of the interval, which is more useful since an estimation result corresponding to a negative variance (even if only produced by numerical inaccuracy) is usually not reliable. If one uses an implementation with precision ($\Lambda = 1/\sigma^2$) and scaled mean ($\lambda = m/\sigma^2$) and the precision is clipped, one has to make sure that the parameter λ is scaled respectively, in order not to change the underlying mean value m.

6.3 I.i.d. Gaussian Sensing Matrix

The standard case of sensing matrices for recovery in CS in the literature [Don06, EK12] is to draw the elements of the sensing matrices i.i.d. from a zero-mean Gaussian distribution with variance $1/M$, i.e., $a_{ij} \sim \mathcal{N}(0,\, 1/M)$. The choice of the variance causes the ℓ_2-norm of the columns to be roughly one, i.e., $\|\tilde{\boldsymbol{a}}_j\|_2^2 \approx 1$ for $j \in \{1,\, \ldots,\, N\}$. For the first scenario, such Gaussian matrices are considered, but with a scaling factor that ensures unit column norms, i.e., $\|\tilde{\boldsymbol{a}}_j\|_2^2 = 1$, cf. [Spa19].

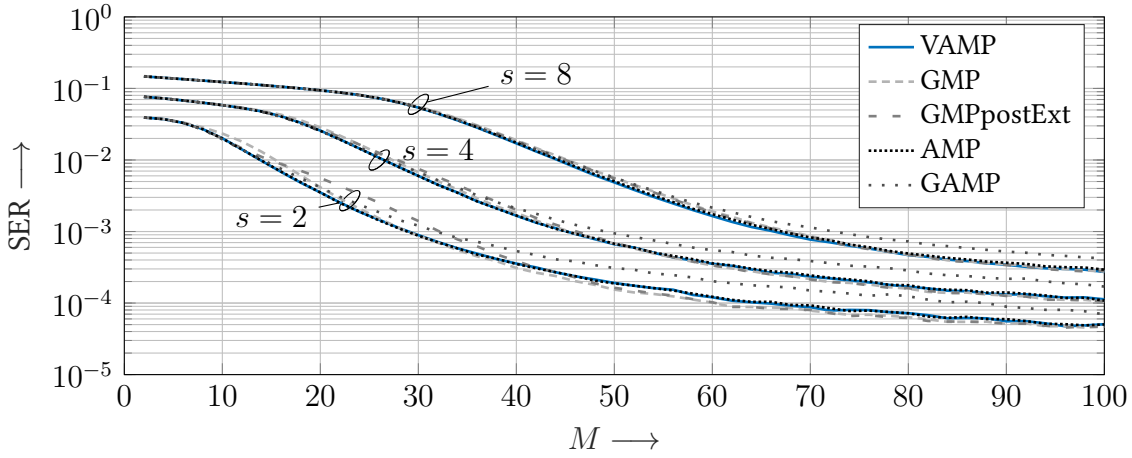
In this scenario, the ability of the different algorithms to compress the signal $\boldsymbol{x}$ is investigated by displaying steady-state performance, which denotes the performance after convergence, over the observation dimension M.

To that end, all the algorithms listed above are simulated, except for VAMPia, VAMPii, and seqVAMPobs, because separate sections (Sec. 6.4 and Sec. 6.6) are spent for the discussion of their properties and they perform exactly as VAMP in the given scenario, which will be shown in the respective sections.

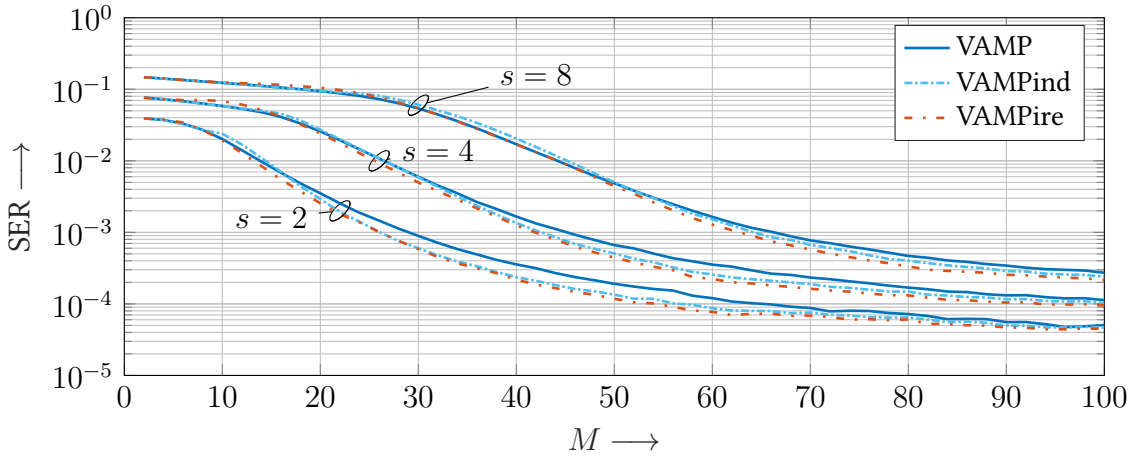
The first results in Fig. 6.2 show the performance over the observation dimension M for a signal dimension $N = 100$ and different values of sparsity s. The upper plot (Fig. 6.2(a)) compares the Turbo algorithm VAMP, with the message-passing-type algorithms GMP and GMPpostExt, as well as the approximate versions AMP [Mal11] and our implementation (Algorithm E.4) of GAMP [Ran11]. Throughout this chapter, the message-passing-type algorithms utilize schedules that iterate between the upward and downward messages by computing all messages of one type, e.g., downward, sequentially and then switching to the other type. In this scenario, AMP and VAMP show a very similar performance in terms of steady-state error, which is not surprising. Our implementation of the GAMP algorithm shows worse performance, especially for larger observation dimensions M, because the neglection of the off-diagonal elements in the LMMSE inverse (for details see Appendix E.4) gets harder to compensate for larger M and GAMP cannot benefit from the averaging effects of usual AMP. The message-passing-type algorithms perform similar to or only slightly worse than VAMP, but GMPpostExt shows some deterioration for small to mid-level values of M. The fact that the performance of GMPpostExt is generally very close to the one of GMP shows that the "post-processing" of the extrinsics after the signal-constrained estimation does not differ much from the "pre-processing" case; at least for larger sparsities s.

In Fig. 6.2(b) the focus lies on the comparison of the Turbo algorithms with VAMP on the one hand, which uses average variances, and on the other hand VAMPind and VAMPire using individual variances. It is visible that the algorithms with individual variances can improve over VAMP in terms of compressibility and general steady-state performance, except for very small M. Moreover, VAMPire usually is slightly better than VAMPind, because of the estimation-theoretic bias compensation after the signal-constrained estimation that omits negative variances, thereby guaranteeing sensible variance values and improving performance compared to the treatment in VAMPind.

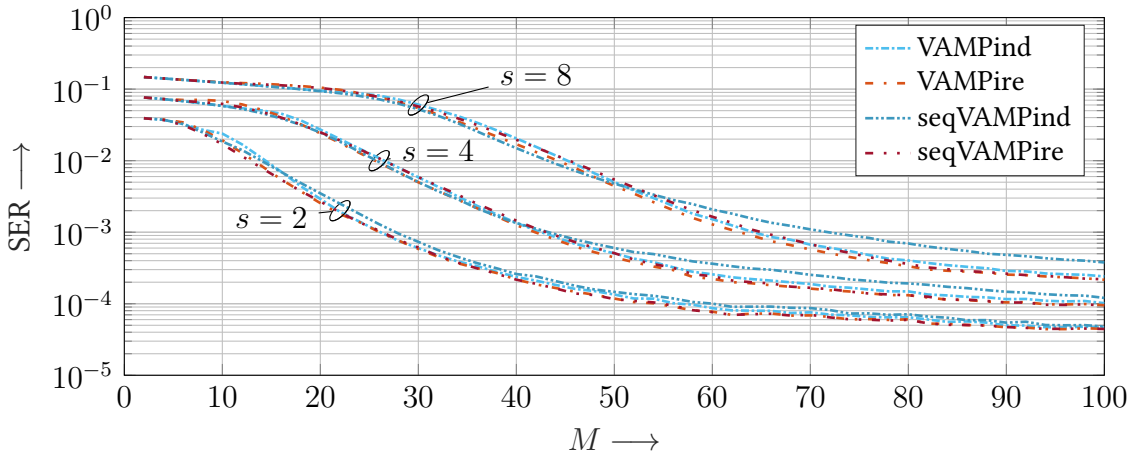
This difference is even more visible in Fig. 6.2(c), where the Turbo algorithms with individual variances are compared to its versions that process the signal components sequentially



(a) Comparison of the message-passing-type algorithms (GMP and GMPpostExt), VAMP as representative of a Turbo-type algorithm, and the AMP-type algorithms.



(b) Comparison of Turbo algorithms with average (VAMP) and individual variances (VAMPind, VAMPire).



(c) Comparison between parallel and sequential algorithms (processing of variable nodes, individual variances).

Figure 6.2: Performance over the observation dimension M. $N = 100$, i.i.d. Gaussian sensing matrices with columns normalized to unit ℓ_2-norm, DT prior, $10\log_{10}(1/\sigma_{\mathsf{n}}^2) = 16\,\mathrm{dB}$, evaluation after 20 iterations.

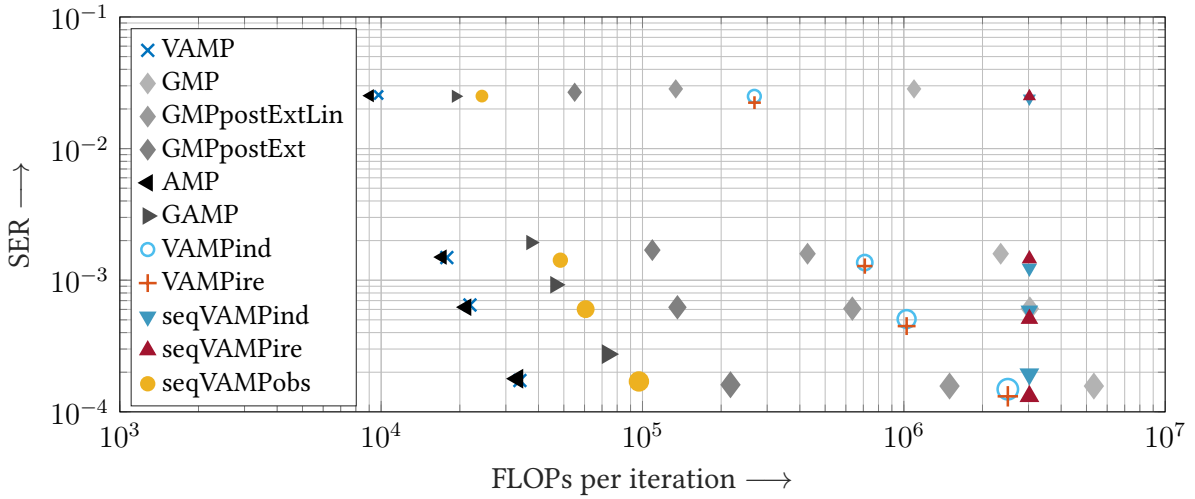


Figure 6.3: Performance over FLOPs per iteration. $N = 100$, i.i.d.Gaussian sensing matrices with columns normalized to unit ℓ_2-norm, DT prior with sparsity $s = 4$, $M \in \{20, 40, 50, 80\}$ with increasing mark size, $10 \log_{10}(1/\sigma_{\mathsf{n}}^2) = 16\,\mathrm{dB}$, evaluation after 50 iterations.

(seqVAMPind and seqVAMPire). Here, the performance of the purely optimization-based sequential approach seqVAMPind performs visibly worse than the other algorithms. The reason therefor is that a negative variance affects the processing even more, since the result is directly broadcasted to all other variables, without a compensating effect from other variables. The sequential version of the estimation-theoretically improved algorithm (seqVAMPire) performs very close to the Turbo-type version VAMPire.

Before turning to more results that investigate the compressibility, let us have a quick look at the complexity of the algorithms. The computational complexity of the algorithms is compared by counting the floating-point operations (FLOPs) per iteration. The term FLOPs shall here comprise additions, subtractions, multiplications, and divisions. The counting is done as explained in [Spa19, Appendix E]. Note that for the signal-constrained estimation the FLOP count in [Spa19, Appendix E] involves the cardinality of the alphabet of non-zero entries, which is for the given prior (2.4) $|\mathcal{X}| = 2$.

The counting for the GMPpostExt algorithm is based on the notation that can be found in Appendix E.2. For GMP a version that utilizes scaled means and inverse variances to save the inversions in Lines 7 and 8 is used as basis for the complexity calculations. Of course, all algorithms could further be optimized, both in terms of storage and computational complexity, but since the requirements depend a lot on the architecture on which the algorithm runs, only a rough overview over the range that the algorithms span in terms of complexity is given.

The results are shown in Fig. 6.3 for the scenario that was investigated before with sparsity $s = 4$ and observation dimension $M \in \{20, 40, 50, 80\}$. On the horizontal axis, the counted FLOPs per iteration are shown; the vertical axis depicts, as usual, the SER. For increasing M the performance gets, as expected, better but in most algorithms also the FLOP count increases.

The AMP and the VAMP algorithm have roughly the same complexity, because the SVD implementation is used for the inverse in the LMMSE, as suggested in [RSF19, Algorithm 2]. The effort of computing the SVD is not added, because it can be computed once before the

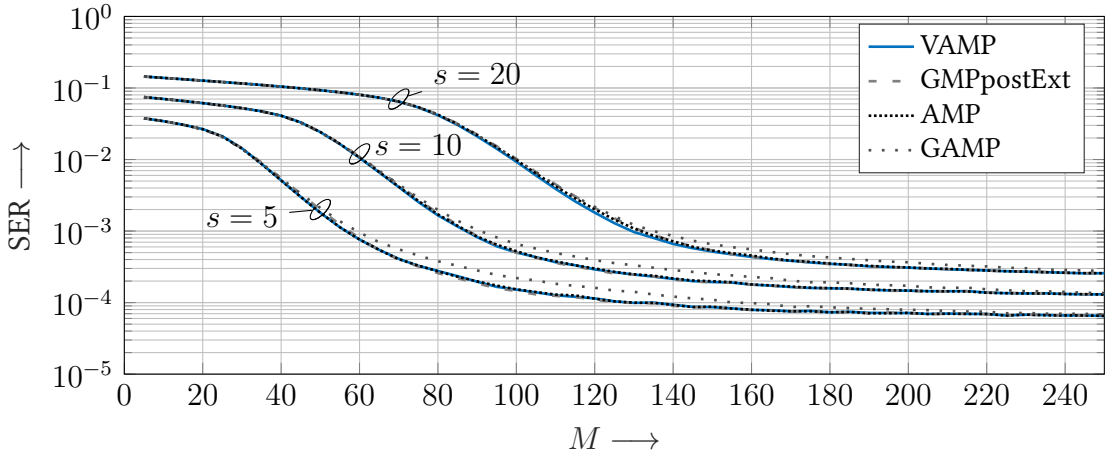
execution of the algorithm. So, especially if a sensing matrix is used several times, the expense can be neglected. For the algorithms with individual variances (VAMPind and VAMPire), this simplification is not possible, because the decomposition does not lead to a diagonal to be inverted, as for the case with an average variance [Spa19, Appendix C]; this leads to a drastic increase in complexity compared to AMP and VAMP. The FLOP count for GAMP is slightly higher than the one of VAMP and AMP, because it involves individual variances, but not as high as VAMPind and VAMPire, because it does not compute the entire LMMSE inverse, but instead only the diagonal elements.

The computational complexity of the GMP-based algorithms (diamonds) spreads over a wide range, i.e., the FLOP counts depends very much on the implementation. The version with pre-processed extrinsic (GMP) has the highest FLOP count, because the expectations have to be computed per edge ($M \cdot N$). Post-processing the extrinsic after the channel-constrained estimation (GMPpostExtLin) already saves so much that the algorithm costs less than the Turbo-type algorithms with individual variances. Note that this implementation performs exactly as GMP, because the updates are mathematically the same. The second simplification (GMPpostExt), which uses post-processed extrinsic also at the signal-constrained estimation, may lead to a slight loss in performance, but can further decrease the computational complexity. Hence, it makes sense to implement in message-passing-type algorithms for CS at least the channel-constrained estimation with post-processed extrinsic.

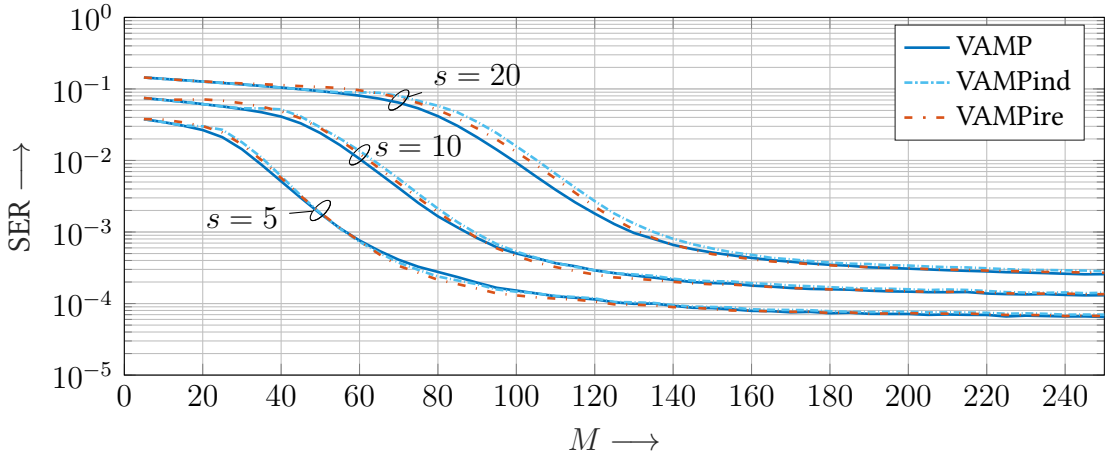
The computational complexity of the sequential approaches, seqVAMPind and seqVAMPire, does not scale well. Since the rank-one update operates on an $N \times N$ matrix, one such update is in the order of $\mathcal{O}(N^2)$ and as will be visible in simulations later on, at least N inner iterations are required so that the overall complexity is of order $\mathcal{O}(N^3)$. Nevertheless, the number of FLOPs does not depend on M so that these algorithms can compete with its Turbo-type versions for values of M close to N. A slight reduction in complexity may be possible by utilizing a Cholesky factorization as suggested in [SN08]; however the factorization dictates the succession of processing the variables, so the flexibility of most of the schedules to be proposed in Sec. 6.5 is lost. Recalling that this kind of processing is related to the (sequential) EP algorithm [Min01, SN11], cf. also Sec. 5.1.2.3, one may find other approaches for simplification in the respective literature. The idea of average EP (AEP) [DB18] and stochastic EP (SEP) [LHLT15] is for example to reduce the parameter space and thereby the storage complexity by in case of AEP averaging the natural parameters suitably such that only the averaged parameter has to be kept. This also simplifies the analysis of the procedure [DB18]. In the CS case this seems to remove to much of the necessary information, so that these techniques were not considered in this thesis. Consequently, these sequential algorithms are only suitable for small systems.

The version of VAMP that processes the observations sequentially (seqVAMPobs) is shown here for completeness; its FLOP count is in the order of GAMP, because the sequentialization increases the computational effort w.r.t. VAMP.

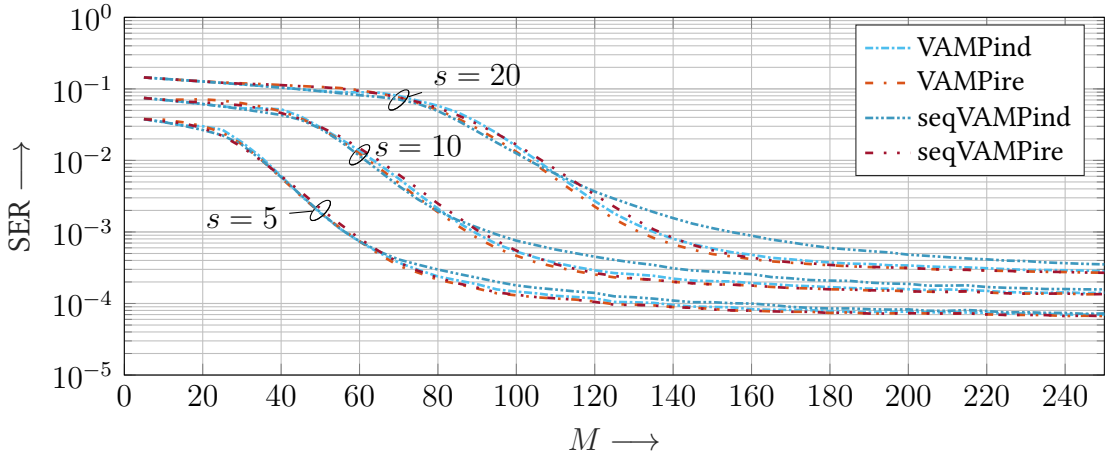
The results in Fig. 6.4 show the scenario with a signal dimension $N = 250$, i.e., the dimensionality of $\boldsymbol{x}$ is increased. The results are plotted for the same relative sparsities s/N as in the figures for $N = 100$. For this choice of signal dimension, the computation of the



(a) Comparison of VAMP, to (approximate) message-passing-type algorithms.



(b) Comparison of Turbo algorithms with average and individual variances.



(c) Comparison between parallel and sequential algorithms (processing of variable nodes, individual variances).

Figure 6.4: Performance over the observation dimension M. $N = 250$, i.i.d. Gaussian sensing matrices with columns normalized to unit ℓ_2-norm, DT prior, $10\log_{10}(1/\sigma_{\mathsf{n}}^2) = 16\,\mathrm{dB}$, evaluation after 20 iterations.

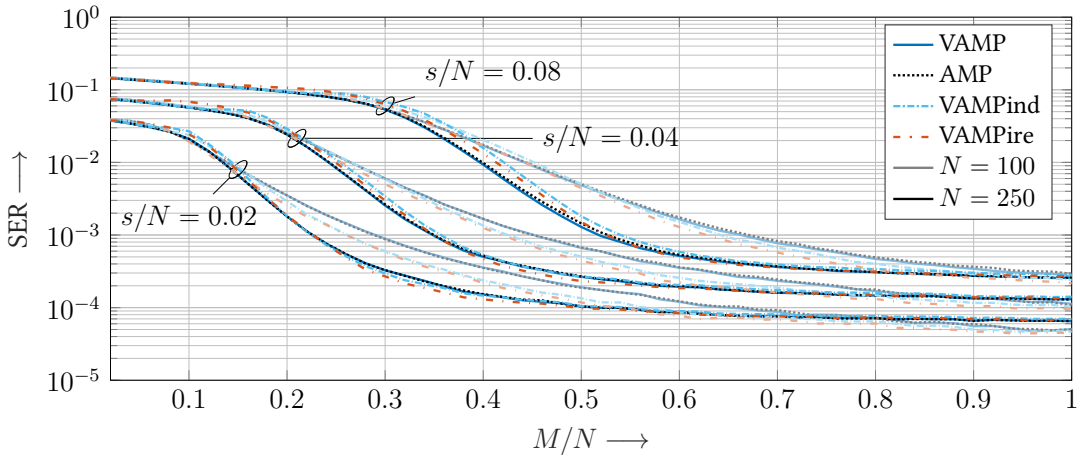


Figure 6.5: Performance over M/N for the two considered values of N and $s/N \in \{0.02, 0.04, 0.08\}$). I.i.d. Gaussian sensing matrices with columns normalized to unit ℓ_2-norm, DT prior, $10\log_{10}(1/\sigma_{\mathsf{n}}^2) = 16\,\mathrm{dB}$, evaluation after 20 iterations.

GMP algorithm is already very tedious, especially for large M. Therefore, the results for the GMP algorithm are not shown. The performances of AMP and VAMP are now even closer, the degradation in GAMP stays, cf. Fig. 6.4(a).

In the middle (Fig. 6.4(b)) the comparison of the behavior of the Turbo-type algorithms with average and individual variances is shown. Here, the performances are also closer than before, i.e., the performance of VAMP improves compared to the algorithms with individual variances (VAMPind and VAMPire). The improvement of AMP and VAMP can be explained by the fact that for larger signal dimensions the performance of recovery algorithms in CS (with average variance) generally improves for growing N because the averaging pulls the (conditional) variances towards the MSE, which is the actual desired measure. For higher sparsities the algorithms with individual variances tend to have a worse compression capability than VAMP, but the steady-state error for large values of M is the same. This shows that the algorithms with individual variances are most suited for systems of small dimensions and very sparse vectors.

The comparison of parallel and sequential algorithms in Fig. 6.4(c) shows similar features as for $N = 100$. The performance of seqVAMPind is worst; the other algorithms differ only slightly in compressibility and steady-state error.

In Fig. 6.5 the effect of an increased signal dimension on the recovery performance for AMP, VAMP, VAMPind, and VAMPire is assessed. The main difference is that the relative compressibility gets better, i.e., the so-called *phase-transition region* [WV12] gets steeper, which is a well known fact in the literature [DMM09]. Indeed, the algorithms with average variances, AMP and VAMP, play to their strengths the larger the system size. Therefore, the algorithms with individual variances can improve mostly for smaller system sizes.

In Fig. 6.6 the convergence of the Turbo algorithms over the iterations is depicted; on the left for $N = 100$, $M = 50$ and on the right for $N = 250$, $M = 125$ with the respective sparsity values used above. On the right side, AMP and VAMP are compared to the algorithms with individual variances, VAMPind and VAMPire, for $N = 250$. AMP shows the slowest

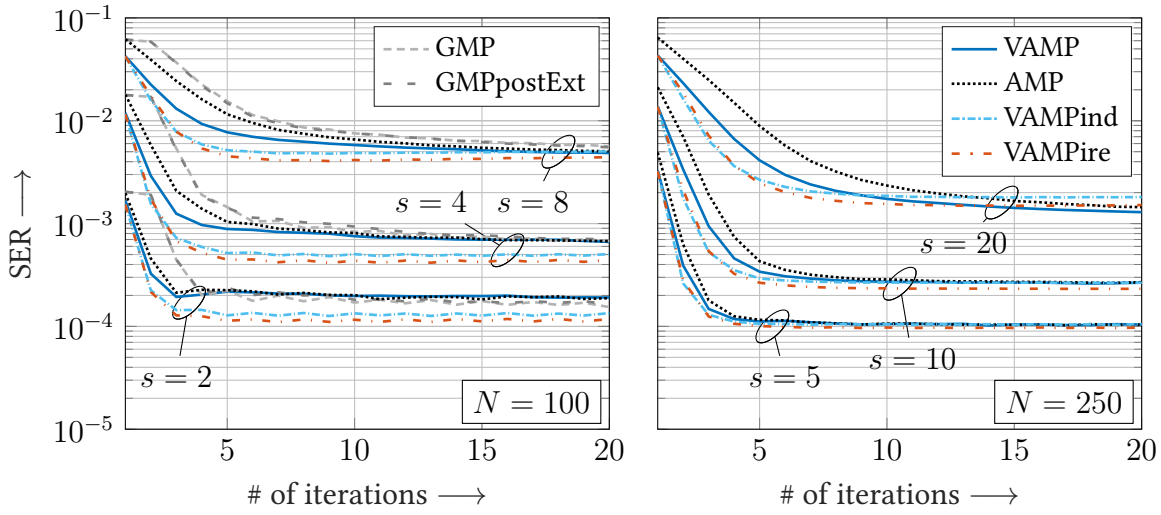


Figure 6.6: Performance over iterations for the two considered values of N and several sparsities (the relative sparsity is the same, i.e., $s/N \in \{0.02,\, 0.04,\, 0.08\}$). $M = N/2$, i.i.d. Gaussian sensing matrices with columns normalized to unit ℓ_2-norm, DT prior, $10\log_{10}(1/\sigma_{\mathsf{n}}^2) = 16\,\mathrm{dB}$.

convergence but reaches (as VAMP) better steady-state performance than the algorithms with individual variances for $s = 20$. For smaller sparsities (in the examples $s \in \{5,\, 10\}$), VAMPire supersedes the other algorithms in terms of steady-state performance, whereas VAMPind only achieves the performance of AMP and VAMP, despite using the more complex computation with individual variances. Generally, the algorithms with individual variances show the fastest convergence. This shows that the algorithms benefit from utilizing individual variances only if the treatment is performed correctly, as derived in the estimation-theoretic approach.

On the left, additionally to the algorithms on the right, also GMP and GMPpostExt are displayed in the scenario with $N = 100$. Both message-passing-type algorithms show slower convergence than AMP, because the fine-granular processing requires more steps to settle. The steady-state error is close to the one of AMP and VAMP, which is in general higher than on the right, which is due to less compressibility for smaller signal dimensions, as discussed beforehand. Again, one can see that the benefit of individual variances has most effect for the smaller system size of $N = 100$.

6.4 Model for Wireless Sensor Networks

Wireless sensor networks (WSNs) [YMG08] are an interesting application for CS [CSD+17, LCJ15, SD11, XDS13]. For several applications, it is necessary to gather information from a certain area. If the respective sensors are spread within this area, the direct access of the sensors might not be given or not desired. For this use, wireless sensor networks are deployed, in which the information of the sensors is collected at a so-called *fusion center*.

In order to reduce the power consumption at the sensors they only send if it is necessary, i.e., their activity may be sporadic [SD11]. Considering a large amount of sensors of whom only few are active in one time step, results in a sparse vector of high dimensionality. This brings

compressed sensing into play, which can simplify the hardware at the fusion center, e.g., by measuring with fewer antennas. Subsequently, the sparse vector needs to be recovered from the measurements. Therefore, the suitability of the discussed algorithms in such a scenario is investigated in the following.

The main effect that has to be addressed in this scenario is the fact that, since the sensors are distributed arbitrarily around the fusion center, the distances to the fusion center vary and together with it the receive power with which the signals of the sensors reach the fusion center. The assumption of unit-ℓ_2-norm columns as in the previous section would require power control of the senders, i.e., the sensors need to send with a signal strength such that the receive power at the fusion center is equal for all sensors. In order to control this, additional communication is required, which is usually not desired since it reduces the battery lifetime of the sensors.

Alternatively, the effect may be interpreted as a non-uniform power distribution over the signal components $x_1, \ldots, x_N$ that is modeled into the sensing matrix $\boldsymbol{A}$ by scaling the columns accordingly, which means that the power distribution is known to the receiver. The power distribution, given by p_j ($j \in \{1, \ldots, N\}$) is applied as follows

$$\boldsymbol{A} = c_{\boldsymbol{A}} \cdot \tilde{\boldsymbol{A}} \cdot \mathbf{diag}\Big(\sqrt{p_j}\Big) = [\tilde{\boldsymbol{a}}_1, \ldots, \tilde{\boldsymbol{a}}_N] \,, \tag{6.6}$$

where $\tilde{\boldsymbol{A}}$ consists of i.i.d. Gaussian entries and the columns are normalized to unit ℓ_2-norm. For comparability, indeed to ensure that each sensing matrix carries the same energy, the factor $c_{\boldsymbol{A}}$ is adjusted such that $\|\boldsymbol{A}\|_{\mathrm{F}}^2 = N$ by

$$c_{\boldsymbol{A}} = \frac{\sqrt{N}}{\left\|\tilde{\boldsymbol{A}} \cdot \mathbf{diag}(\sqrt{p_j})\right\|_{\mathrm{F}}} \,, \tag{6.7}$$

i.e., the Frobenius norm (for the definition see Appendix A) is equal to the unit-ℓ_2-norm case. In the simulations, the power distribution is specified by parameter $v \in (0,\, 1]$ in the form of

$$p_\ell = v^{(\ell-1)/(N-1)} \,, \qquad \ell \in \{1, \ldots, N\} \,, \tag{6.8}$$

and the scaling factor is assigned to the jth column by a random permutation $j = \pi(\ell)$. For $v = 1$, the unit-ℓ_2-norm case is recovered; with decreasing v the variations in the scalings vary more and more. Furthermore, also the condition number (cf. Appendix A) of the sensing matrix increases. Noteworthy, the largest column power (before Frobenius normalization) is 1, the smallest is v.

An example for such a power distribution is plotted for $v = 0.5$ and $N = 250$ in Fig. 6.7. It shows also the effect of the Frobenius normalization for a sensing matrix of size 125×250.

As pointed out in Sec. 4.4, VAMP is challenged by this scenario because the effects of the sensing matrix are treated on average, not individually. As alternative, the algorithms that utilize individual unbiasing instead are considered. These are VAMPind and VAMPire, which use individual variances throughout the entire iteration, as well as VAMPia and VAMPii, the algorithms with average variance in the channel-constrained estimation. This comparison

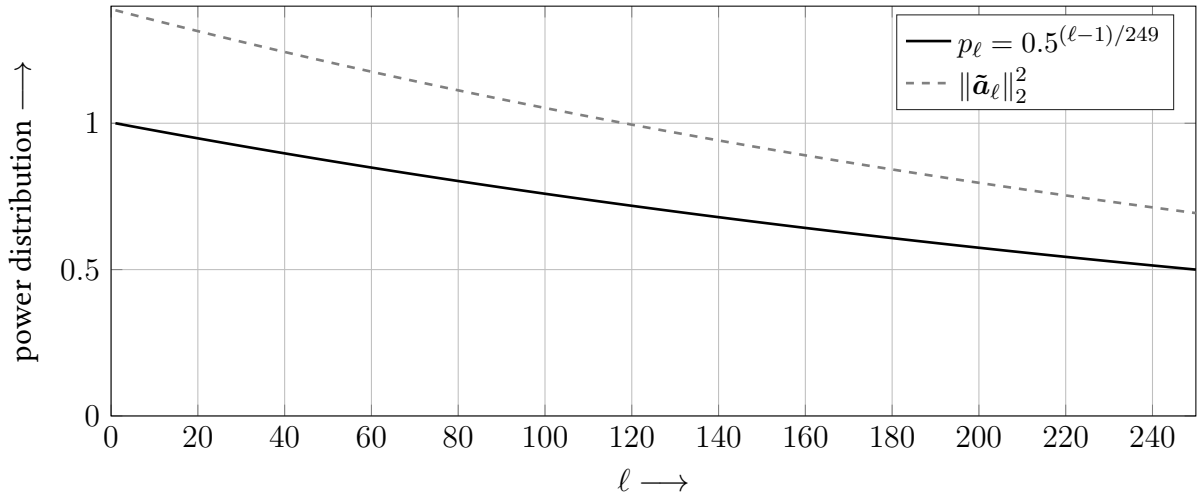


Figure 6.7: Power distribution (6.8) for $N = 250$, $v = 0.5$, i.i.d. Gaussian matrices before and after Frobenius normalization.

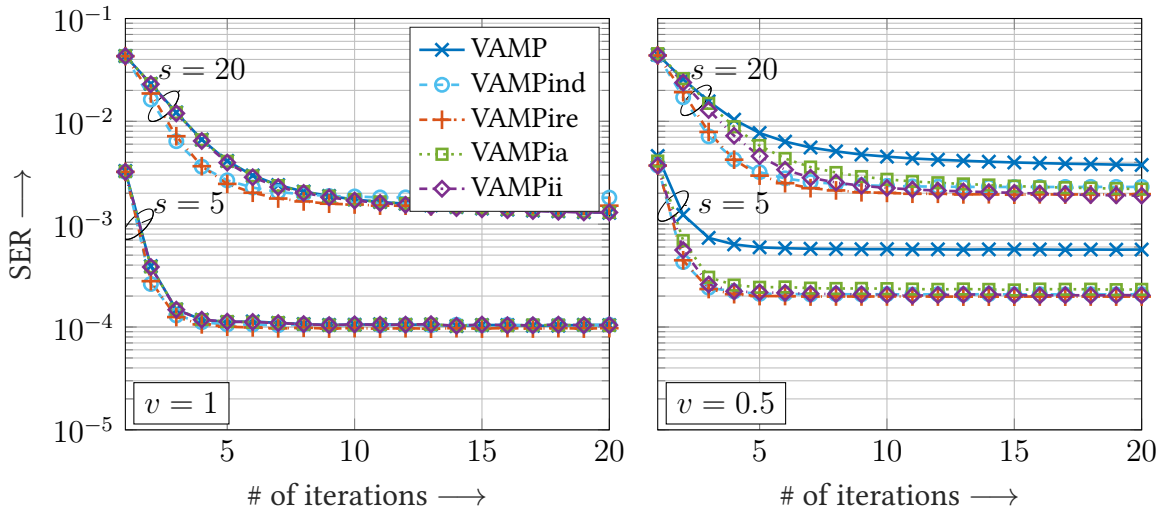


Figure 6.8: Performance over the iterations. $N = 250$, $M = 125$, i.i.d. Gaussian sensing matrices with power profile (6.8), DT prior, $10\log_{10}(1/\sigma_{\mathsf{n}}^2) = 16\,\mathrm{dB}$.

has first been considered in [SF22], the scenario with power distribution has slightly different earlier been used in [SF21].

Assuming simple binary transmission of the sensors in the WSN, translates into the DT prior (2.4) being a suitable choice for the signal model. In this section, we stick to signal dimension $N = 250$, observation dimension $M = 125$ and SNR $10\log_{10}(1/\sigma_{\mathsf{n}}^2) = 16\,\mathrm{dB}$ if not stated otherwise.

First, the convergence over the iterations is depicted in Fig. 6.8. The left side compares VAMP and the variants for the usual case where the columns are normalized to unit ℓ_2-norm, which coincides with the power distribution (6.8) with $v = 1$. One can see that the variants VAMPia and VAMPii perform exactly the same as VAMP. To understand why this is, regard

the end-to-end cascade $\boldsymbol{K}$ from (4.44) with $\boldsymbol{\Phi}_{\mathsf{x}} = \sigma_{\mathsf{x}}^2 \boldsymbol{I}_N$

$$\boldsymbol{K} = \sigma_{\mathsf{x}}^2 \boldsymbol{A}^\top (\sigma_{\mathsf{x}}^2 \boldsymbol{A}\boldsymbol{A}^\top + \sigma_{\mathsf{n}}^2 \boldsymbol{I}_M)^{-1} \boldsymbol{A} = (\boldsymbol{A}^\top \boldsymbol{A} + \frac{\sigma_{\mathsf{n}}^2}{\sigma_{\mathsf{x}}^2} \boldsymbol{I}_N)^{-1} \boldsymbol{A}^\top \boldsymbol{A} \,. \tag{6.9}$$

First, note that the Gram matrix of $\boldsymbol{A}$ can be written as

$$\boldsymbol{A}^\top \boldsymbol{A} = \begin{bmatrix} \sum_{i=1}^M a_{i1}^2 & \sum_{i=1}^M a_{i1} a_{i2} & \dots & \sum_{i=1}^M a_{i1} a_{iN} \\ \sum_{i=1}^M a_{i1} a_{i2} & \sum_{i=1}^M a_{i2}^2 & \dots & \sum_{i=1}^M a_{i2} a_{iN} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^M a_{i1} a_{iN} & \sum_{i=1}^M a_{i2} a_{iN} & \dots & \sum_{i=1}^M a_{iN}^2 \end{bmatrix} , \tag{6.10}$$

which shows that the diagonal contains the column energies (squared ℓ_2-norms), which are in the case $v = 1$ all equal to 1, i.e., $\sum_{i=1}^M a_{ij}^2 = 1$ for $j \in \{1, \dots, N\}$. Sensing matrices are usually chosen such that the measurements are as independent as possible from each other, in order to gain as much information as possible from each individual measurement. This means that the sensing matrices should have a low *coherence* [EK12, Def. 1.5]. Luckily, i.i.d. Gaussian matrices meet this requirement relatively well, if the dimensions are large enough [Don06][1], leading to $\sum_{i=1}^M a_{ij} a_{ij'} \approx 0$ or at least $|\sum_{i=1}^M a_{ij} a_{ij'}| \ll 1$. With this knowledge, one finds that $\boldsymbol{M} = \boldsymbol{A}^\top \boldsymbol{A} + \frac{\sigma_{\mathsf{n}}^2}{\sigma_{\mathsf{x}}^2} \boldsymbol{I}_N$ can in this case be written as

$$\boldsymbol{M} = \begin{bmatrix} 1 + \frac{\sigma_{\mathsf{n}}^2}{\sigma_{\mathsf{x}}^2} & \epsilon_{12} & \epsilon_{13} & \dots & \epsilon_{1N} \\ \epsilon_{12} & 1 + \frac{\sigma_{\mathsf{n}}^2}{\sigma_{\mathsf{x}}^2} & \epsilon_{23} & \dots & \epsilon_{2N} \\ \epsilon_{13} & \epsilon_{23} & 1 + \frac{\sigma_{\mathsf{n}}^2}{\sigma_{\mathsf{x}}^2} & & \vdots \\ \vdots & \vdots & & \ddots & \epsilon_{N-1N} \\ \epsilon_{1N} & \epsilon_{2N} & \dots & \epsilon_{N-1N} & 1 + \frac{\sigma_{\mathsf{n}}^2}{\sigma_{\mathsf{x}}^2} \end{bmatrix} , \tag{6.11}$$

with $|\epsilon_{jj'}| \ll 1$ for $j' \neq j$, $j, j' \in \{1, \dots, N\}$. Hence, one can deduce that in the inverse $\boldsymbol{M}^{-1}$ mostly the diagonal elements $1 + \frac{\sigma_{\mathsf{n}}^2}{\sigma_{\mathsf{x}}^2}$ contribute and, consequently, the diagonal entries of $\boldsymbol{K}$ are dominated by equal or close to equal scaling factors. Compensating the effects on average ignores the small variation but does not result in significant errors. Thus, the effect of average bias compensation is in this case very close to individual bias compensation. The other two algorithms, VAMPind and VAMPire, show a somewhat faster convergence due to their utilization of individual variances for the channel-constrained estimation.

By decreasing v, variations between the column energies appear due to the different scaling factors of the power distribution. So, the effect of individual and average bias compensation after the channel-constrained estimation differs. The convergence over iterations for $v = 0.5$ is shown on the right side of Fig. 6.8. Comparing to the left reveals that the steady-state performance is in general worse, which is due to the fact that the estimation problem overall gets harder under these conditions. It is clearly visible that the performance of VAMP degrades most for the considered algorithms. This can be explained by the stronger variations that appear in the diagonal entries of the end-to-end cascade $\boldsymbol{K}$; the effect can no longer be compensated on average.

[1]There are ways to generate matrices, that fulfill the property of low coherence even better [Spa19], but for our purposes, the Gaussian matrices suffice. For a definition of the coherence, see also Appendix A.

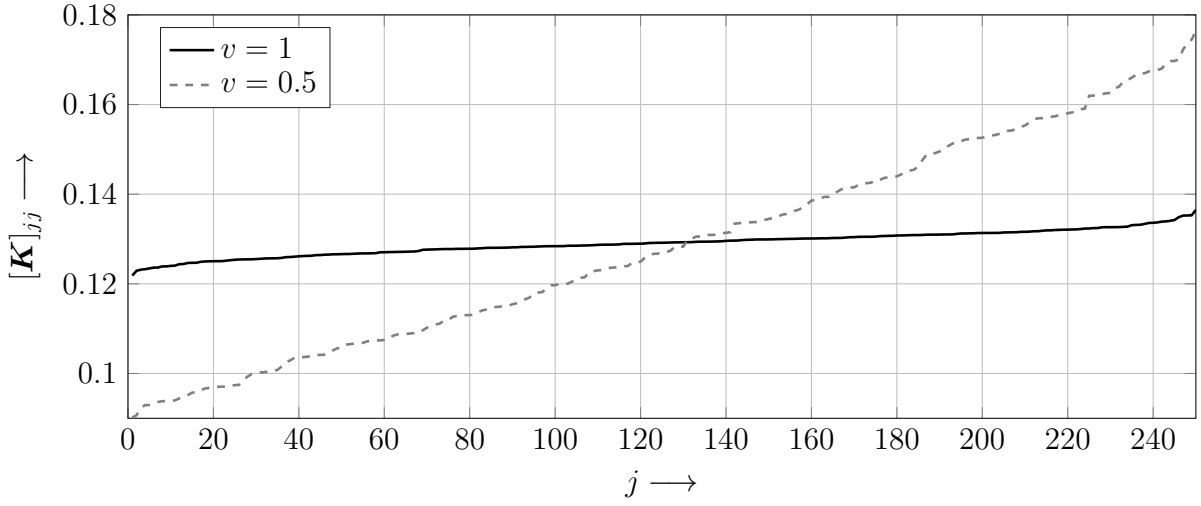


Figure 6.9: Example for variations in diagonal of $\boldsymbol{K}$ for an i.i.d. Gaussian matrix of size 125×250, $\sigma_{\mathsf{x}}^2 = 1/25$, $\sigma_{\mathsf{n}}^2 = 0.2$, with ($v = 0.5$) and without ($v = 1$) power distribution. The diagonal entries are sorted in ascending order for the plot.

An example for the variations that result in the diagonal of $\boldsymbol{K}$ is displayed for an i.i.d. Gaussian matrix of size 125×250, whose columns are normalized to unit ℓ_2-norm in one case and compared to the case with applied power distribution (6.8) with $v = 0.5$. For the calculation of $\boldsymbol{K}$, $\sigma_{\mathsf{x}}^2 = 1/25$ is used, which corresponds to the initialization with $s = 10$, and $\sigma_{\mathsf{n}}^2 = 0.2$, which represents an SNR of roughly $10 \log_{10}(1/\sigma_{\mathsf{n}}^2) = 17\,\mathrm{dB}$. The diagonal entries are sorted in ascending order for better visibility in the plot. One can see that with employed power distribution, the variations between diagonal elements increase.

In order to see the entire picture, the simulations over the parameter v of the power distribution are shown in Fig. 6.10. One can see that the performance of VAMP degrades much more than the other algorithms when v decreases and, thus, more power variations are present. Considering the algorithms with individual bias compensation and average variance in the channel-constrained estimation, VAMPii is preferable over VAMPia, because VAMPii shows slightly better or equal steady-state error. This means that the usage of individual variances in the signal-constrained estimation can yield some insight to lower the steady-state error. For $s = 20$, the algorithms with individual variances are inferior to the ones with average variances when $v \to 1$, which coincides with the slightly worse compressibility of the algorithms for larger variances in Fig. 6.4(b). In this case, the individual variance algorithms show their benefit only for very small values of v, i.e., they are generally less impaired by large variations in the column powers, because of more insight to reliabilities of the channel-constrained estimates.

In Fig. 6.11 the steady-state error in terms of SER over the SNR, given by the inverse noise variance in dB, is considered. On the left, the results for $v = 1$ are shown, where the curves are very close. Noteworthy, the algorithms with individual variances are generally better for higher SNR, except for $s = 20$, where they generally deteriorate as already discussed. On the right, one can see the results for $v = 0.5$; it is again apparent that the performance of VAMP

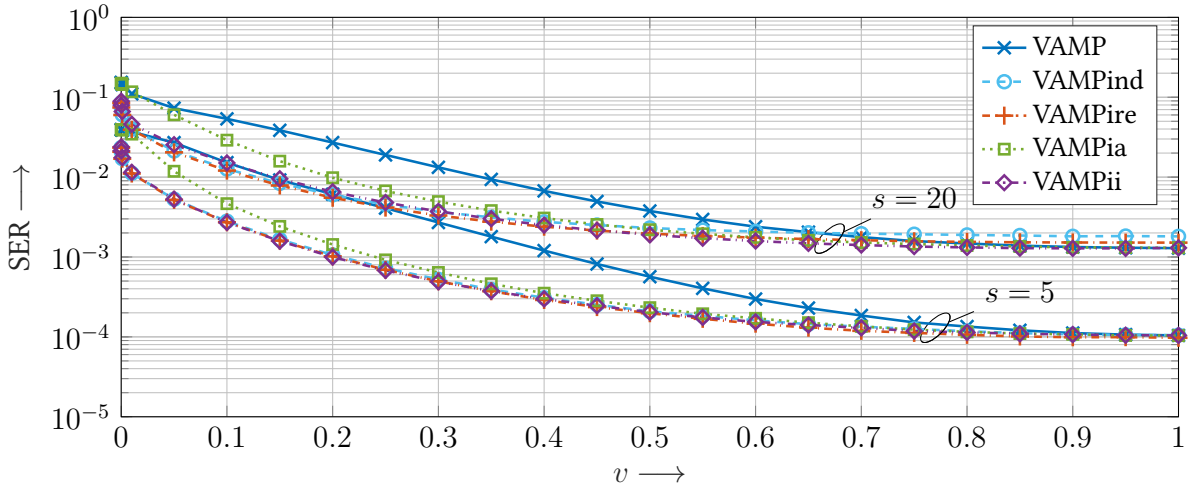


Figure 6.10: Performance over parameter v of the power profile (6.8). $N = 250$, $M = 125$, i.i.d. Gaussian sensing matrices, DT prior, $10\log_{10}(1/\sigma_{\mathsf{n}}^2) = 16\,\mathrm{dB}$, evaluation after 20 iterations.

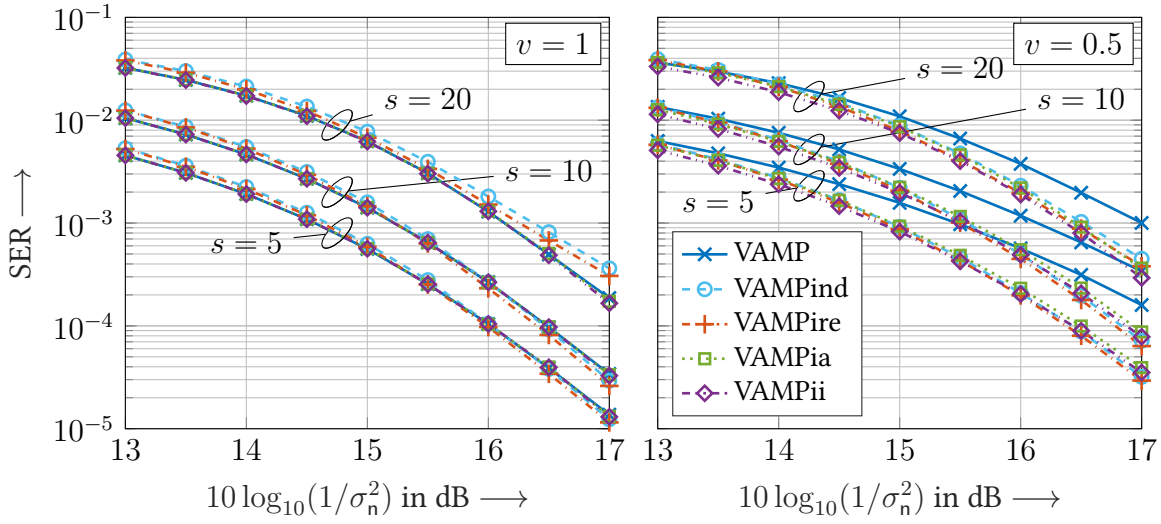


Figure 6.11: Performance over SNR. $N = 250$, $M = 125$, i.i.d. Gaussian sensing matrices with power profile (6.8), DT prior for different sparsities, evaluation after 20 iterations.

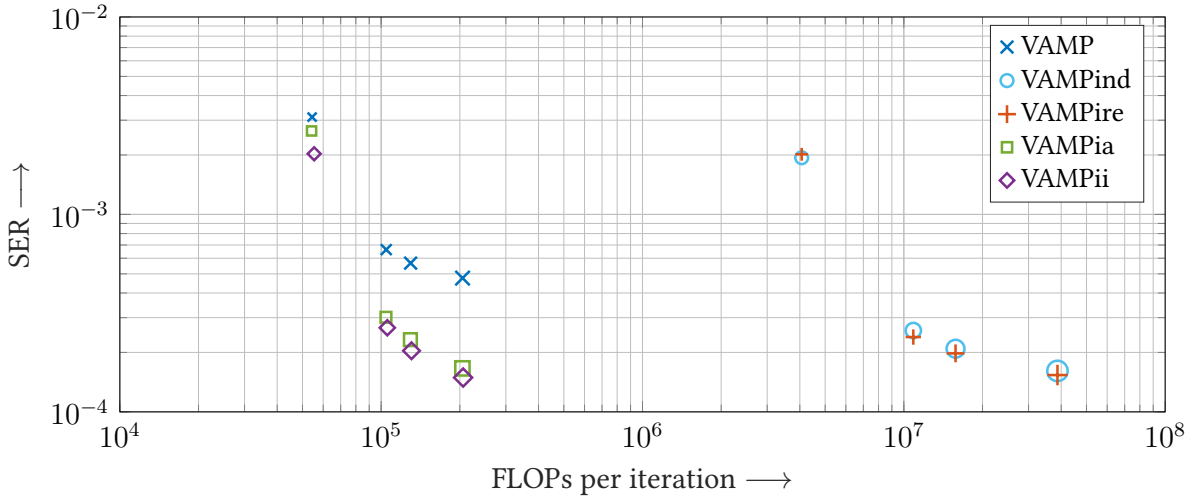


Figure 6.12: Performance over FLOPs per iteration. $N = 250$, i.i.d. Gaussian sensing matrices with power profile (6.8), $v = 0.5$, DT prior with sparsity $s = 5$, $M \in \{50, 100, 125, 200\}$ with increasing mark size, $10\log_{10}(1/\sigma_{\mathsf{n}}^2) = 16\,\mathrm{dB}$, evaluation after 20 iterations.

degrades more than all other algorithms with VAMPii being mostly the best.

Up to now, it has become clear that the algorithms VAMPia and VAMPii can very well compete with the algorithms with individual variances. In order to see the actual benefit over the strategies with individual variances, the computational complexity of the algorithms is compared by counting the floating-point operations (FLOPs) per iteration. The results are plotted in Fig. 6.12; the simulations and computations were executed for a sparsity $s = 5$ and observation dimension $M \in \{50, 100, 125, 200\}$. For the algorithms with average variance in the channel-constrained estimation, the SVD implementation is utilized for the inverse in the LMMSE [RSF19, Algorithm 2]. The benefit of the SVD implementation is obvious, as the individual variance algorithms need up to two orders of magnitude more FLOPs per iteration. Since the algorithms combine both, the low complexity of VAMP as well as a good performance due to the individual bias compensation after the channel-constrained estimation, they are the algorithms to be preferred in such a scenario. Since VAMPii is due to the usage of individual variances in the signal-constrained estimation usually slightly better than VAMPia, it is preferable over VAMPia.

VAMP is usually promoted by its stability in ill-conditioned sensing scenarios, where the sensing matrix has high condition number κ [RSF19], for the definition of the condition number in the form that it is used here, see Appendix A. The comparison of the algorithms in such a scenario[2], see Fig. 6.13, shows that the algorithms utilizing individual variances can improve over VAMP, as well as the algorithms with individual bias compensation and average variance at the channel-constrained estimation. Noteworthy, VAMPii supersedes partially even the algorithms with individual variances, which is remarkable considering the much lower complexity. The explanation for that is the average variance in the channel-constrained estimation, which brings more stability into the algorithm, thereby improving the overall convergence.

[2]The sensing matrix $\boldsymbol{A}$ is generated as in [RSF19, Sec. VI.A], which is explained in Appendix A.

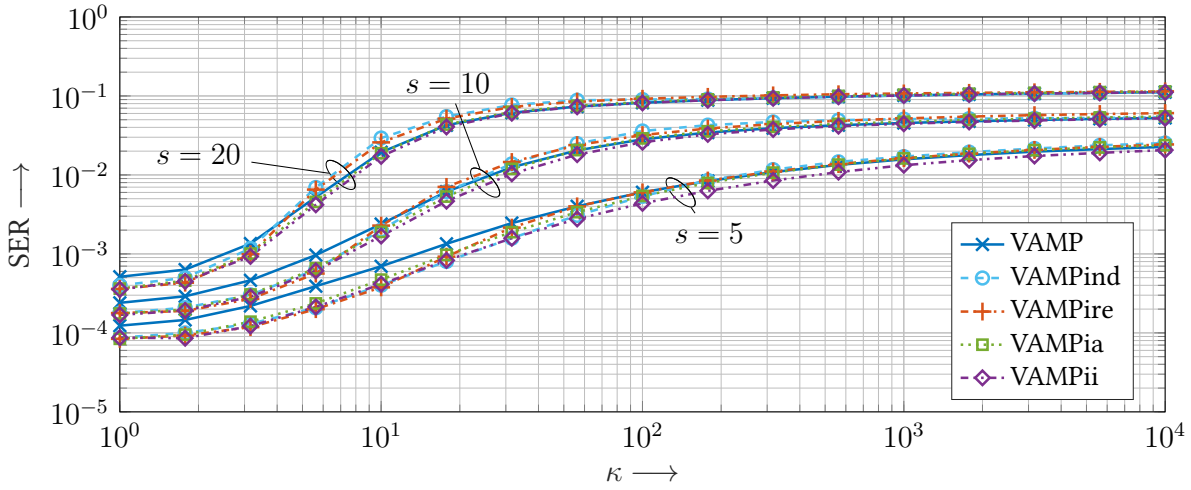


Figure 6.13: Performance over condition number κ of sensing matrix $\boldsymbol{A}$. $N = 250$, $M = 125$, DT prior, $10\log_{10}(1/\sigma_{\mathsf{n}}^2) = 16\,\mathrm{dB}$, evaluation after 20 iterations.

6.5 Schedules for the Sequential Processing of Variables

In this section, simulation results for the algorithms that utilize a sequential processing of the signal components and were introduced in Sec. 5.1.4 and 5.1.5 are presented. They are compared to the Turbo-type algorithms VAMP, VAMPind, and VAMPire, since they are closest to them in terms of estimation strategy with VAMP being furthest apart, because it employs average variances as reliabilities for the estimations.

The scenario is the same as in the section above, i.e., i.i.d. Gaussian sensing matrices of size 125×250 are drawn and the power distribution (6.8) is applied. As signal model, the DT prior (2.4) with $s = 5$, if not stated otherwise, is assumed. The signal-to-noise ratio is set to $10\log_{10}(1/\sigma_{\mathsf{n}}^2) = 16\,\mathrm{dB}$ throughout the simulations. Similar results have been shown in [SF21].

The simulations in Sec. 6.3 have already shown that seqVAMPire usually performs close to VAMPire and improves over seqVAMPind in the standard setting utilizing i.i.d. Gaussian matrices with columns normalized to unit ℓ_2-norm. The comparison in the section at hand focuses mostly on the difference in convergence speed, but before turning to that, the performance over the parameter v of the power distribution (6.8) is considered, in order to get a picture of the performance. The results are shown in Fig. 6.14. For small values of v, the performance of the sequential algorithms is very close to the Turbo-type algorithms with individual variances from which they are derived. For $v \to 1$, the performance tends to get worse than the parallel algorithms, which is especially visible for the larger sparsity value $s = 20$. The sequential algorithms utilize the $\sigma^2\nearrow$ schedule, which will be explained below. Noteworthy, this schedule was also used in the simulations over the observation dimensions in Sec. 6.3.

In the following, the simulations are restricted to $s = 5$ and $v \in \{0.5,\, 1\}$.

Opposing the Turbo-type algorithms, where each processing step is prescribed, the sequential processing of the signal components enables a degree of freedom in choosing among the signal components the one to be processed in the next step. This opens the possibility for

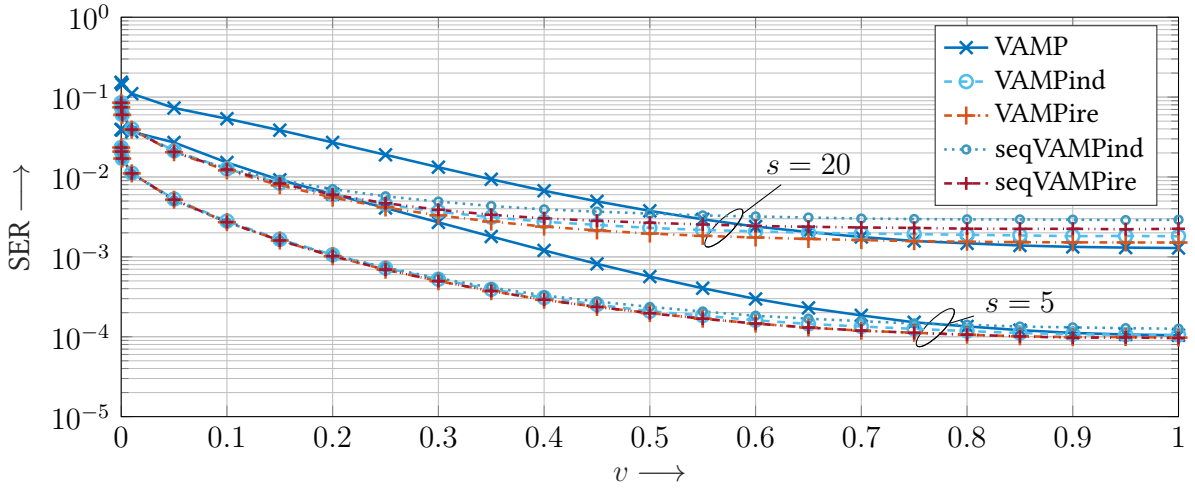


Figure 6.14: Performance over parameter v of the power profile (6.8). $N = 250$, $M = 125$, i.i.d. Gaussian sensing matrices, DT prior, $10\log_{10}(1/\sigma_{\mathrm{n}}^2) = 16\,\mathrm{dB}$, evaluation after 20 iterations. Comparison between Turbo-type and sequential (processing variables) algorithms. The sequential algorithms utilize the $\sigma^2\nearrow$ schedule, which will be explained below.

a variety of schedules. The schedules that are discussed here were first introduced in [SF21].

In order to be able to properly define the schedules, a distinction between *inner iterations* and *outer iterations* is employed. One *inner iteration* comprises the computation steps that are described in Fig. 5.2, i.e., the signal-constrained estimation of one signal component x_j and the subsequent broadcasting of the result to the other signal components. An *outer iteration* describes the processing of N inner iterations, i.e., the number of inner iterations that are necessary to process each signal component at least once. These outer iterations are used to compare to the Turbo-type algorithms, which process all signal components together.

At first, schedules that sweep entirely through the signal components are considered, so that no component is left out and none is processed twice in one outer iteration. Therefore, one can directly compare over the outer iterations. As schedules, a random choice and two deterministic choices are introduced. The random choice (random) draws the order to be processed before each outer iteration, i.e., permutes the sequence $[1, \ldots, N]$ and runs through it. For the deterministic choices, the variances $\sigma_{\mathrm{c},j}^2$ ($j \in \{1, \ldots, N\}$) that represent the reliability in the channel-constrained estimation of the respective signal component need to be considered. The variances are either sorted in ascending ($\sigma^2\nearrow$), or in descending ($\sigma^2\searrow$) order and the order is kept throughout the outer iteration.

The comparison of these schedules over the iteration is depicted in Fig. 6.15, on the left for $v = 1$, on the right for $v = 0.5$. In the unit-column-norm case ($v = 1$), there is almost no difference between the schedules. Nevertheless, it is visible that the convergence of the sequential algorithms is slightly faster than the Turbo-type algorithms in terms of outer iterations. Furthermore, in contrast to seqVAMPire, whose steady-state error ends up at the same level as VAMPire, the sequential version of VAMPind (called seqVAMPind) results in a steady-state error worse than all the compared Turbo-type algorithms. This shows that the estimation-theoretic view on bias compensation becomes even more important, when the view on the

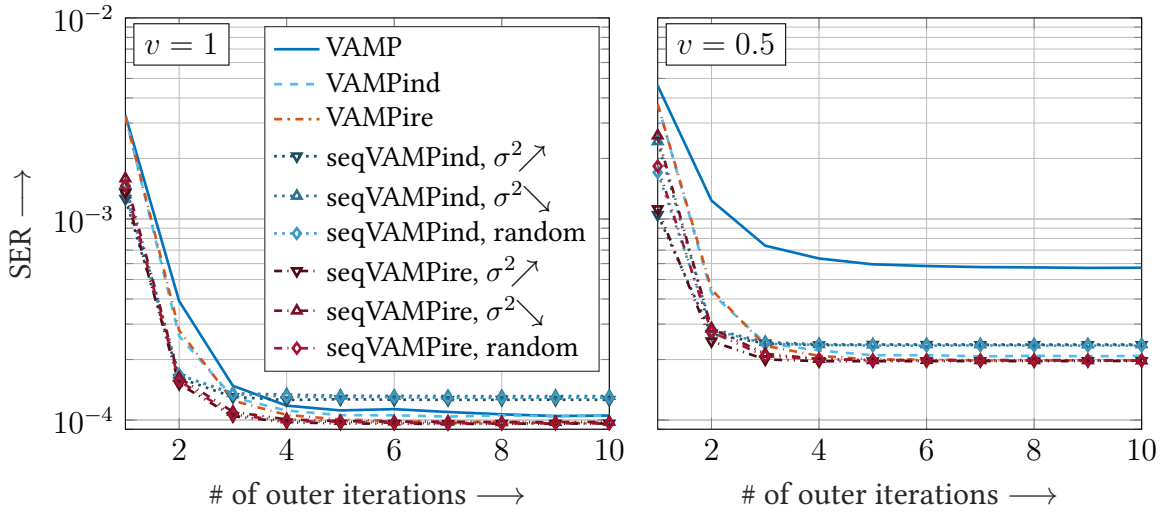


Figure 6.15: Comparison of schedules that encompass all signal elements per outer iteration. $N = 250$, $M = 125$, i.i.d. Gaussian sensing matrices with power distribution (6.8), DT prior with sparsity $s = 5$, $10\log_{10}(1/\sigma_{\mathrm{n}}^2) = 16\,\mathrm{dB}$.

estimation is even more detailed as in this sequential processing case. The faster convergence of the sequential algorithms can be explained by the fact that the knowledge gained by processing a single variable is immediately fed back to the other variables, so that within an outer iteration already more knowledge can be gained as when processing all variables together.

For $v = 0.5$, one can see a difference between the schedules; especially after the first outer iteration. Later the schedules tend to the same level in terms of SER. The schedules that exceeds the other two in the first few iterations is $\sigma^2\nearrow$, i.e., the processing in order of ascending variances. The schedule $\sigma^2\searrow$ is worst and the random schedule is between the other two.

The reason for this order is as follows. Since the variance $\sigma_{\mathrm{c},j}^2$ represents the reliability in the estimates, starting with low variances (as in $\sigma^2\nearrow$) means to start with the ones one is most sure about, which leads to more confidence in subsequent estimations of signal components and thereby to faster convergence. The second deterministic schedule $\sigma^2\searrow$ works oppositely and therefore performs worst, although still being better than the Turbo-type algorithms.

The schedules examined so far followed the strategy of choosing the order once before an outer iteration and kept this order throughout the outer iteration. However, it might be worthwhile to reconsider and perhaps change the order of processing within one outer iteration. For that purpose, the index set $\mathcal{J} \subseteq \{1, \ldots, N\}$ of cardinality $|\mathcal{J}| = T$ is introduced, which specifies the T variables that are processed before reconsidering the order.

There are plenty of ways to choose the index set $\mathcal{J}$. Again two deterministic and one random strategy are examined. The deterministic approaches follow the deterministic choices introduced earlier, i.e., one picks the positions corresponding to the T smallest variances, the other goes for the T highest variances.

For the random approach, a strategy that prefers high variances is chosen. The motivation therefor is that the recovery algorithm needs to make sure that the reliability in the

estimation increases over the iterations, i.e., the variances need to decrease. The variance of a signal component generally decreases most, when the signal component itself is processed in the signal-constrained estimation, because the signal-constrained (non-linear) estimation yields more accurate estimates than the channel-constrained (linear) one. Signal components with high variance $\sigma^2_{\mathrm{c},j}$ therefore indicate that they require to be processed via the signal-constrained estimation.

In order to focus on high variances, the positions are drawn according to a respective probability distribution. The distribution is created such that the probability for a position j to be drawn is proportional to its corresponding variance $\sigma^2_{\mathrm{c},j}$. Since unique positions in the set $\mathcal{J}$ are necessary, it is built successively. Assume that $\ell < T$ positions have already been drawn, i.e., define $\mathcal{J}^{(\ell)} = \{j_1, \ldots, j_\ell\}$. Then, the probability for the other positions to be drawn is given by

$$\Pr\{\mathrm{j}_{\ell+1} = j\} = \frac{\sigma^2_{\mathrm{c},j}}{\sum_{j'=1, j' \notin \mathcal{J}^{(\ell)}}^{N} \sigma^2_{\mathrm{c},j'}} , \qquad j \in \{1, \ldots, N\} \setminus \mathcal{J}^{(\ell)} . \tag{6.12}$$

This procedure is denoted by “rv”.

In order to make sure that all the positions are processed equally often, one can track the number of processings conducted for each position j with a counter c_j. For later reference, the set of positions with minimal counters is denoted by

$$\mathcal{J}_{\mathrm{c}} = \{j \mid c_j \leq c_{j'} \; \forall j, j' \in \{1, \ldots, N\}\} . \tag{6.13}$$

This procedure is especially necessary for the procedure minv, which chooses always the positions with lowest variances (see below), because, as explained before, processing a signal component leads to a decrease of the respective variance, which means that this procedure is likely to pick signal components that it has chosen in the first iteration over and over again, without ever processing the high-variance components which actually needs to be processed. When considering the deterministic strategies, we will additionally always keep track of the processing by the counter.

Furthermore, one can deliberately switch between the strategies before choosing the next $\mathcal{J}$. When the index set $\mathcal{J}$ is specified, the order of processing the set can be chosen as one of the previously considered strategies, namely, $\sigma^2\nearrow$, $\sigma^2\searrow$, and random.

In the simulations, we restrict to the extreme case, opposing the strategies shown before, i.e., $T = 1$ so that after each inner iteration, the processing order is reconsidered. In this case, there is of course no need to order the set, which consists of one position only, i.e., $\mathcal{J} = \{j\}$. In summary, the following strategies are considered

- maxv: choose $j = \operatorname{argmax}_{j' \in \mathcal{J}_{\mathrm{c}}} \sigma^2_{\mathrm{c},j'}$ with $\mathcal{J}_{\mathrm{c}}$ from (6.13),
- minv: choose $j = \operatorname{argmin}_{j' \in \mathcal{J}_{\mathrm{c}}} \sigma^2_{\mathrm{c},j'}$ with $\mathcal{J}_{\mathrm{c}}$ from (6.13),
- rv: draw position from $\{1, \ldots, N\}$ randomly with probability given by (6.12),
- minv + maxv,
- minv + rv,

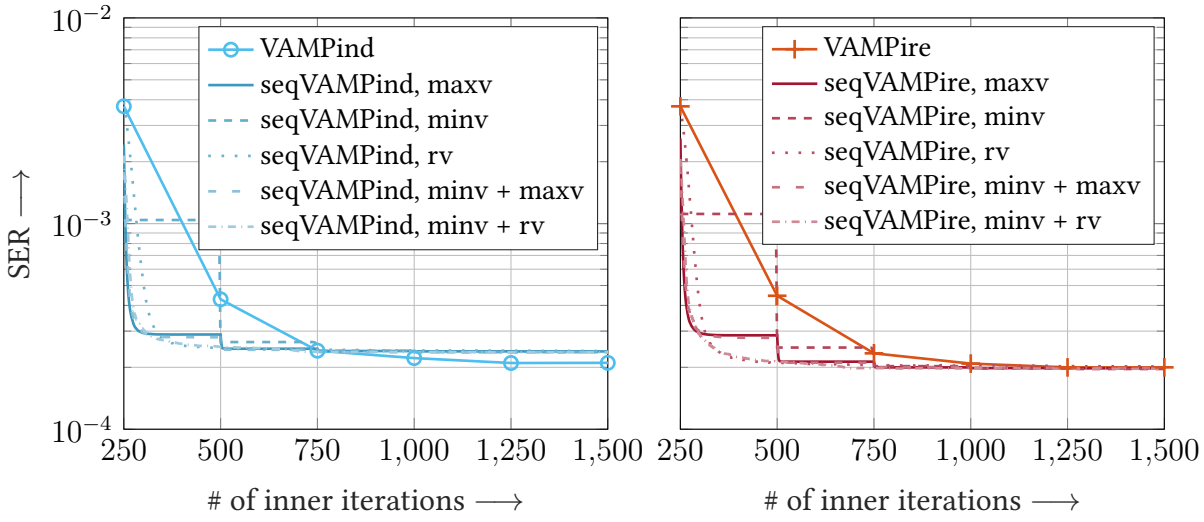


Figure 6.16: Comparison of schedules with $T = 1$ over inner iterations. $N = 250$, $M = 125$, power distribution (6.8) with $v = 0.5$, DT prior with $s = 5$, $10\log_{10}(1/\sigma_{\mathrm{n}}^2) = 16\,\mathrm{dB}$.

where the plus means that the strategy is switched after every processed signal component. Because the schedules show differences, especially when the power distribution is applied, the focus lies now on simulations with $v = 0.5$.

The simulation results are shown in Fig. 6.16 over the inner iterations, where on the left side the individual variance version of VAMP (VAMPind) is compared to its sequential version (seqVAMPind) and on the right the respective algorithms improved by estimation-theoretic bias compensation (VAMPire and seqVAMPire) are depicted. The Turbo-type algorithms are evaluated at multiples of $N = 250$, because they process all N signal components at once. As can be seen, the general behavior of the schedules is for both sequential algorithms, seqVAMPind and seqVAMPire the same, only the steady-state performance differs, which is a property of the algorithm, as already discussed.

For better visibility, the results of estimation-theoretically improved algorithms is again shown in Fig. 6.17, so that one can describe and interpret the behavior of the schedules. Since the counter is used for the deterministic strategies, they process all signal components once between multiples of N, i.e., per outer iteration. When choosing always the largest variance (maxv) from the (per outer iteration) not yet processed signal components, a fast convergence at the beginning of the outer iteration results, because the respective positions require processing the most. Towards the end of the loop, the performance flattens out, because the positions that need less attention are addressed. This leads to a staircase behavior. The staircase is better visible in the opposite case (minv), which chooses the position with lowest variance (among the per outer iteration not yet processed signal components). The behavior is contrary to maxv in the sense that the performance stays equal at the beginning of the outer iteration and shows a drop at the end. Noteworthy, with this drop it gets below the maxv strategy, which is consistent with the results from Fig. 6.15, where the strategy going for low variances first ($\sigma^2\nearrow$) improved over the opposite strategy ($\sigma^2\searrow$).

The probabilistic approach does not make use of the counter and therefore converges

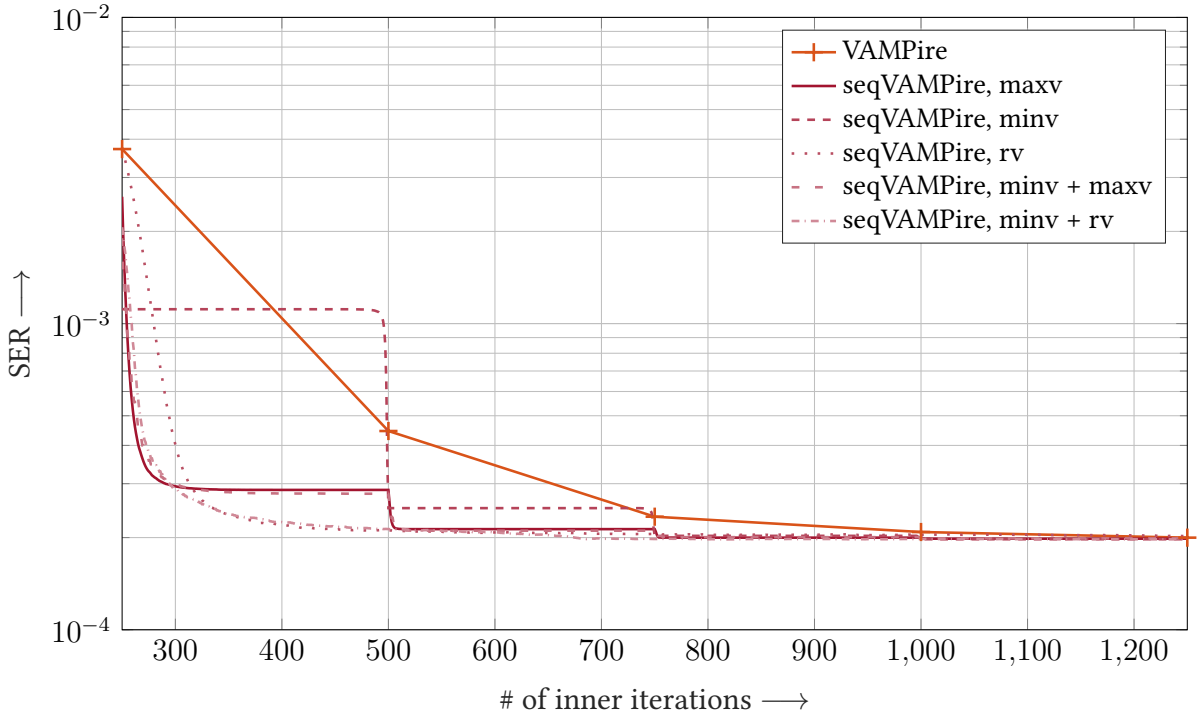


Figure 6.17: Comparison of schedules with $T = 1$ over inner iterations for the estimation-theoretically adapted algorithms. $N = 250$, $M = 125$, power distribution (6.8) with $v = 0.5$, DT prior with sparsity $s = 5$, $10 \log_{10}(1/\sigma_{\mathsf{n}}^2) = 16\,\mathrm{dB}$.

smoothly, but the convergence is not as steep as for maxv at the beginning.

With the switching between opposing strategies, it is possible to combine the advantages of the respective strategies and yield both fast convergence and well performance at convergence; in the given comparison, the combination of rv and minv is the preferable schedule, since it combines the good convergence of the minv schedule with the smoothness (no steps) of the rv schedule.

6.6 Variable Noise Scenario

In this last scenario, the case of non-i.i.d. noise with independent but individual noise variances as introduced in Sec. 5.2 is considered, i.e.,

$$\mathbf{n} \sim \mathcal{N}\left(\mathbf{0},\, \mathbf{diag}\left(\sigma_{\mathsf{n},i}^2\right)\right) . \tag{6.14}$$

This scenario was used to motivate the sequential processing of observations, which contrasts VAMP in the sense that VAMP considers all observations at once, i.e., parallel. The algorithm seqVAMPobs is therefore compared to two versions of VAMP. Naturally, the known noise variances $\sigma_{\mathsf{n},1}^2, \ldots, \sigma_{\mathsf{n},M}^2$ can simply be considered in the VAMP algorithm by computing the channel-constrained estimation by

$$\boldsymbol{m}_{\mathrm{c}} = \tilde{\boldsymbol{x}}_{\mathrm{c}} + \sigma_{\mathsf{x}}^2 \boldsymbol{A}^{\mathsf{T}} \left(\sigma_{\mathsf{x}}^2 \boldsymbol{A}\boldsymbol{A}^{\mathsf{T}} + \mathbf{diag}\left(\sigma_{\mathsf{n},i}^2\right)\right)^{-1} \left(\boldsymbol{y} - \boldsymbol{A}\tilde{\boldsymbol{x}}_{\mathrm{c}}\right) , \tag{6.15}$$

with respective error covariance matrix

$$\boldsymbol{\Phi}_{\mathrm{c}} = \left(\boldsymbol{A}^{\top} \mathbf{diag}\left(1/\sigma_{\mathsf{n},i}^2\right) \boldsymbol{A} + \frac{1}{\sigma_{\mathsf{x}}^2} \boldsymbol{I}_N \right)^{-1} . \tag{6.16}$$

Note that our implementation uses the direct computation of the unbiased variable $\tilde{\boldsymbol{x}}_{\mathrm{s}}$ by employing the scaling matrix $\boldsymbol{W}$ appropriately, so that $\boldsymbol{\Phi}_{\mathrm{c}}$ does not have to be computed directly and, thus, the computation of the $N \times N$ inverse can be spared. The other variant of VAMP averages the noise variances $\sigma_{\mathsf{n}}^2 = \frac{1}{N} \sum_{i=1}^{M} \sigma_{\mathsf{n},i}^2$ and uses this single value for the noise variance as usual.

If not stated otherwise, a scenario with $N = 250$, $M = 125$, the DT prior with various sparsities, and $10 \log_{10}(1/\sigma_{\mathsf{n}}^2) = 16\,\mathrm{dB}$ is simulated.

The different noise variances are obtained by scaling the noise variance σ_{n}^2, which is specified by the SNR, with factors that are drawn uniformly from the interval $[1 - b/2,\, 1 + b/2]$, i.e., the random variable ς_{n}^2 representing the noise variance σ_{n}^2, is uniformly distributed around its mean value σ_{n}^2 with width $b\sigma_{\mathsf{n}}^2$ by

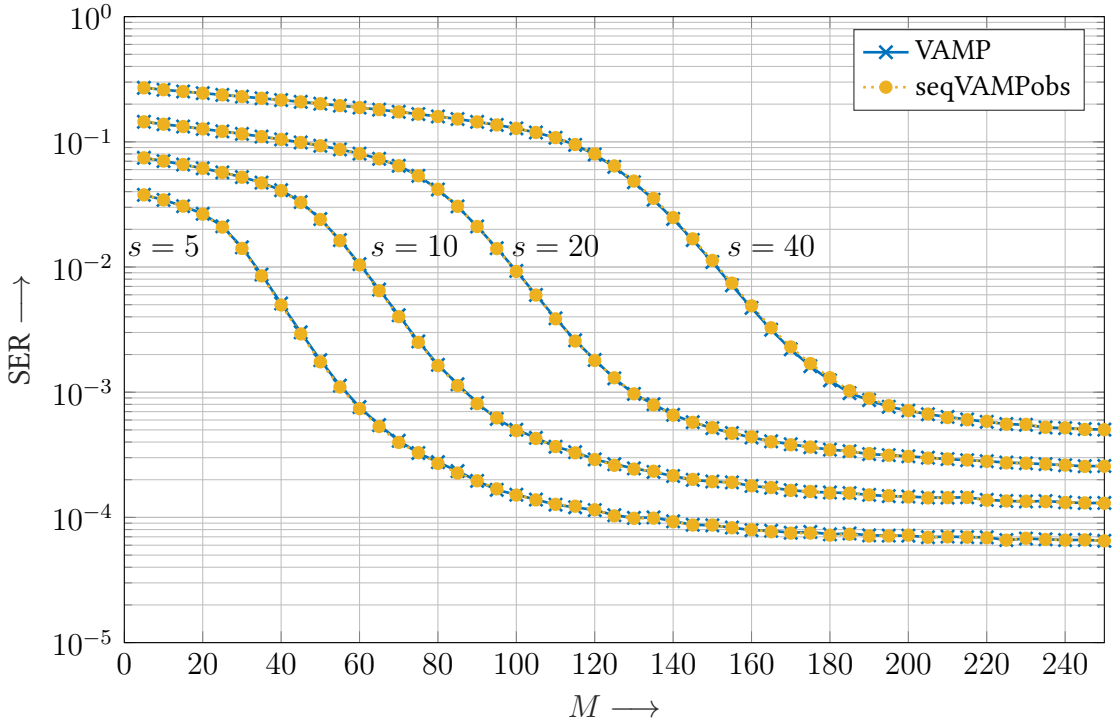
$$\varsigma_{\mathsf{n}}^2 \sim \mathcal{U}\Big([\sigma_{\mathsf{n}}^2(1 - b/2),\, \sigma_{\mathsf{n}}^2(1 + b/2)] \Big) . \tag{6.17}$$

Since negative variances are not possible, this naturally yields a maximum of $b \leq 2$ on the width parameter b of the uniform distribution. For $b = 0$, the i.i.d.-noise case is retrieved.

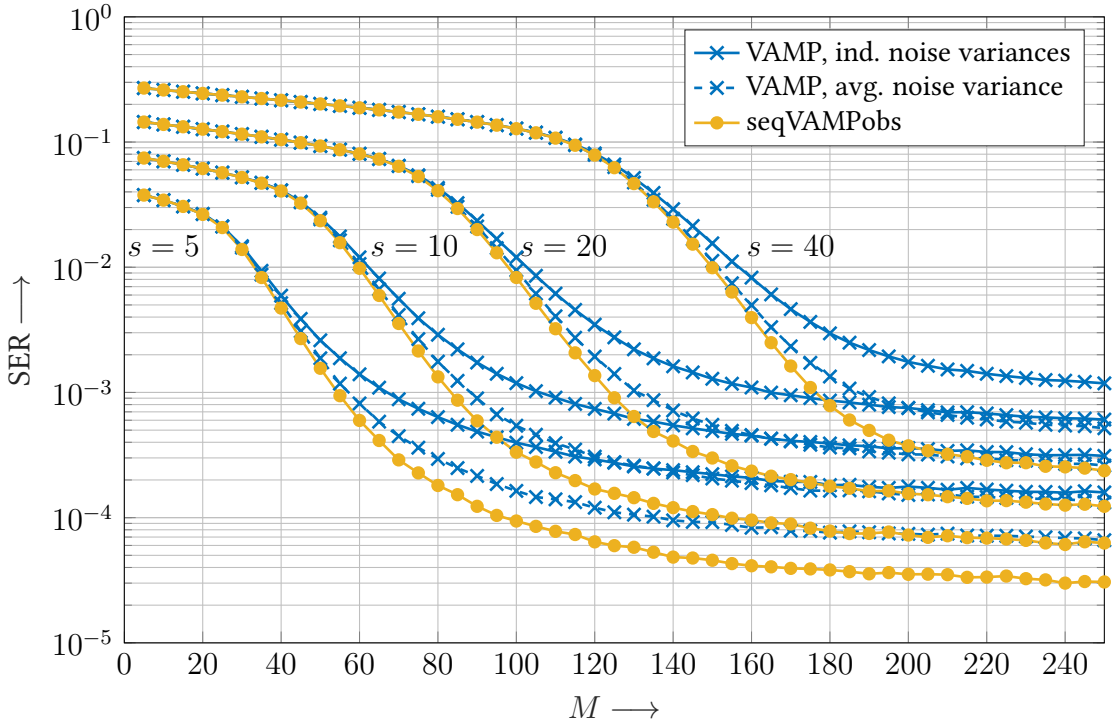
First, the standard case with i.i.d. noise is considered. As before, i.i.d. Gaussian sensing matrices with columns normalized to unit ℓ_2-norm are used. VAMP and seqVAMPobs are compared over the observation dimension M in Fig. 6.18(a). The simulation shows that in this standard case, both algorithms perform exactly the same, which is no surprise since they are based on the same framework.

In contrast, the same plot is shown for non-i.i.d. noise with noise variance uniformly distributed around σ_{n}^2 with parameter $b = 1$ in Fig. 6.18(b), where one can see that seqVAMPobs improves over VAMP both, in terms of compressibility and resulting steady-state error for $M \to N$. Interestingly, VAMP performs better, when the average of the noise variances is used in the algorithms, but still not getting the performance of seqVAMPobs. The worse performance of VAMP with individual noise variances can be explained by the fact that the scaled identity matrix of using a single noise variance has an averaging effect in the processing, thereby making the inverse of the LMMSE much more stable, especially compared to the case when small values occur on the diagonal of the matrix that has to be inverted. The sequential approach benefits over these approaches because it can make use of the knowledge about the individual noise variances without getting impairments by the less stable matrix inverse.

In order to see the impact of the noise variance model on the behavior of the algorithms, they are compared over the value of b that parameterizes the width of the uniform distribution. Recall that for $b = 0$ the i.i.d.-noise case is recovered and the maximum is naturally given by $b = 2$. The results are shown in Fig. 6.19 for $M = 125$. On the left side, one can see that since $b = 0$ recovers the i.i.d. case, VAMP and seqVAMPobs perform the same. Towards the right, the performance of VAMP that utilizes the individual noise variances decreases, due to the



(a) Usual case with i.i.d. noise.



(b) Non-i.i.d.-noise case (6.17) with $b = 1$.

Figure 6.18: Comparison of VAMP and seqVAMPobs over the observation dimension M. $N = 250$, i.i.d. Gaussian sensing matrices with columns normalized to unit ℓ_2-norm, DT prior, $10\log_{10}(1/\sigma_{\mathsf{n}}^2) = 16\,\mathrm{dB}$, evaluation after 20 iterations.

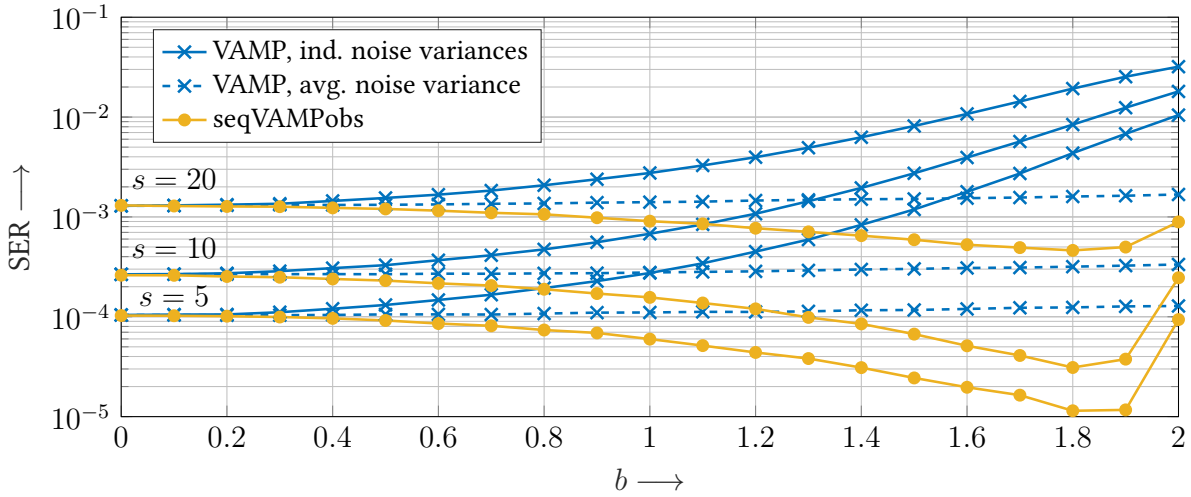


Figure 6.19: Performance over width parameter b of the uniform distribution (6.17) for the noise variance. $N = 250$, $M = 125$, i.i.d. Gaussian sensing matrices with columns normalized to unit ℓ_2-norm, DT prior, $10\log_{10}(1/\sigma_{\mathsf{n}}^2) = 16\,\mathrm{dB}$, evaluation after 20 iterations.

missing averaging effect inside the LMMSE inverse, as already mentioned. When the average of the noise variances is used, the performance of VAMP stays almost constant. In contrast, the sequential version seqVAMPobs can actually improve performance w.r.t. the i.i.d.-noise case, because it handles the observations individually, which only gets less stable for $b \approx 2$.

The sequential processing of the observations enables the use of certain schedules. As in the previous section, a random schedule is compared to two deterministic schedules with opposite ordering. For the deterministic sorting, a measure is needed that characterizes the reliability of the respective factor. In case of the channel-constrained factors $\mathsf{f}_{\mathsf{y}_i|\mathsf{x}}(\boldsymbol{x})$, this is the ℓ_2-norm of the row $\boldsymbol{a}_i$, which tells us how much the signal is amplified by the measurement, divided by the respective noise variance $\sigma_{\mathsf{n},i}^2$. Additionally, there is the signal-constrained factor $\mathsf{f}_{\mathsf{x}}(\boldsymbol{x})$, which is characterized by $\sigma_{\mathsf{x}}^2 = s/N$. Since these measures do not change throughout the iterations of the algorithm, the sorting is done once at the beginning and stays until the end. The random approach permutes the $M+1$ possibilities before each iteration through the factors.

The sequential approach for the signal components from Sec. 5.1 showed that the processing of the signal-constrained estimation is very crucial for the progress of the algorithm. One may therefore emphasize on the factor $\mathsf{f}_{\mathsf{x}}(\boldsymbol{x})$, when designing schedules for the sequential processing of observations. In order to stay somewhat fair to VAMP, an iteration for the sequential algorithm seqVAMPobs is defined to be the processing of exactly $M+1$ factors. In the simulations below, versions of the previously introduced schedules are shown that process the factor $\mathsf{f}_{\mathsf{x}}(\boldsymbol{x})$ every second time; so after one channel-constrained factor $\mathsf{f}_{\mathsf{y}_i|\mathsf{x}}(\boldsymbol{x})$ always $\mathsf{f}_{\mathsf{x}}(\boldsymbol{x})$ is processed, before the next channel-constrained factor is chosen. All in all, the following schedules are considered (the index s is used to denote the signal factor)

- $\nearrow$: sort $\{\|\boldsymbol{a}_i\|_2^2/\sigma_{\mathsf{n},i}^2 \mid i \in \{1,\ldots,M\}\} \cup \{s/N\}$ in ascending order
- $\searrow$: sort $\{\|\boldsymbol{a}_i\|_2^2/\sigma_{\mathsf{n},i}^2 \mid i \in \{1,\ldots,M\}\} \cup \{s/N\}$ in descending order

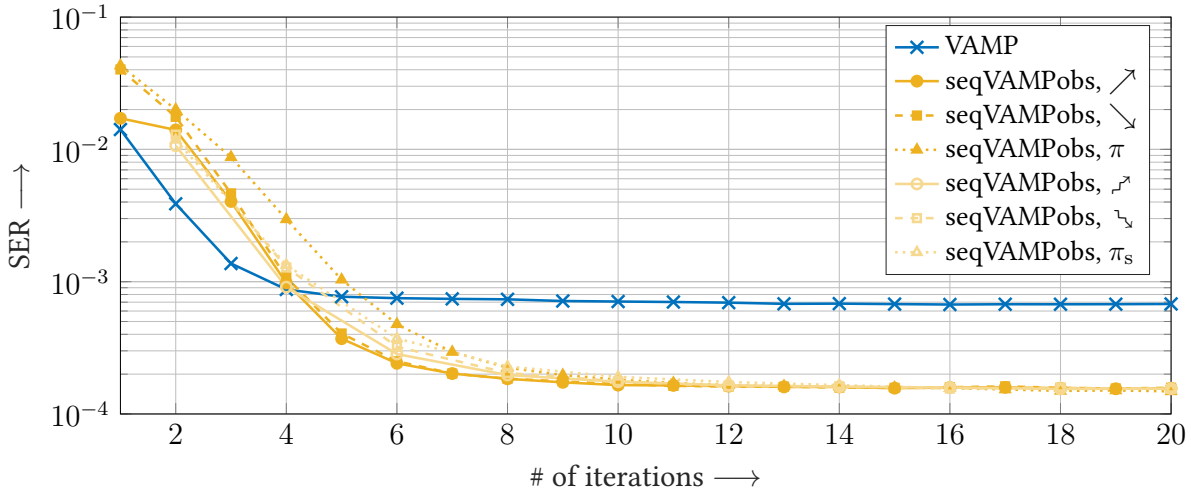


Figure 6.20: Performance over iterations with comparison of schedules. $N = 250$, $M = 125$, i.i.d. Gaussian sensing matrices with columns normalized to unit ℓ_2-norm, non-i.i.d. noise with uniformly distributed variance according to (6.17) with $b = 1$, DT prior with sparsity $s = 10$, $10\log_{10}(1/\sigma_{\mathrm{n}}^2) = 16\,\mathrm{dB}$.

- π: random permutation of $[\mathrm{s},\, 1,\, \ldots,\, M]$ before each iteration
- $\nearrow\!\!\!\!\nearrow$: sort $\|\boldsymbol{a}_i\|_2^2 / \sigma_{\mathrm{n},i}^2$ $\nearrow$ and interleave indices with s
- $\searrow\!\!\!\!\searrow$: sort $\|\boldsymbol{a}_i\|_2^2 / \sigma_{\mathrm{n},i}^2$ $\searrow$ and interleave indices with s
- π_{s}: permutation of $[1,\, \ldots,\, M]$ interleaved with s, yielding $[\mathrm{s},\, \pi(1),\, \mathrm{s},\, \pi(2),\, \ldots,\, \mathrm{s},\, \pi(M)]$

The simulations shown so far used the scheduling that starts with the most reliable observation and proceeds towards the least reliable one, i.e., $\nearrow$.

The comparison of the schedules over the iterations are shown in Fig. 6.20 for $b = 1$. One can see that VAMP converges faster, but has higher steady-state error, as expected from the previous results. The schedules considering every factor once per iteration are shown with filled markers and the schedules emphasizing on the signal-constrained factor $\mathrm{f}_{\mathsf{x}}(\boldsymbol{x})$ are shown lighter and with non-filled markers. The sequential algorithm is evaluated each time when the schedule is run through entirely. Because the schedules emphasizing on s process $2M$ factors, instead of $M + 1$ as the others, they are only plotted every second iteration.

Except for the first iteration the two deterministic procedures yield very close results; also the emphasis on the signal-constrained estimation does not gain much, if at all. The approach with ascending variances yields the most benefit in the first iteration, which means the most insight from reliable observations is obtained at the beginning. The random and the descending variance schedule start similar in the first iteration, but the random approach, converges a little slower; a deterministic approach seems to be the means of choice. The steady-state error of the different schedules is the same.

7. Conclusions and Future Work

The thesis at hand considered recovery algorithms for compressed sensing (CS), i.e., the reconstruction of high-dimensional signals from linearly compressed measurements in noise. Turbo-type and message-passing-type algorithms were derived from the framework of minimizing the Kullback–Leibler divergence under expectation constraints tailored to the respective factor graph representation, which shows the common basis for both types of algorithms. Furthermore, sequential algorithms were developed under the same framework and comparisons of all algorithms were documented via numerical simulations.

7.1 Conclusions

In Chapter 2, the CS model was introduced and the reconstruction task was characterized by two different kinds of estimations that resulted from independently treating the two basic constraints, which are imposed by the problem. These constraints were distinguished as the *channel-constrained* part, resulting from the linear compression with known (sensing) matrix, and the *signal-constrained* part, which is defined by the inherent structure of the signal, i.e., its statistical properties (in form of a probability density function) and especially its *sparsity*. The exact form of the channel-constrained estimation depended on the considered factorizations of the problem. Different representations of the problem were introduced in the form of factor graphs and the approach of minimizing the Kullback–Leibler divergence was motivated.

The most fine-granular representation of the problem led in Chapter 3 to message-passing algorithms, which simplify the estimation to the treatment of scalars (instead of vectors). This simplification of the estimation required extensive exchange (messages) between the many factors and variables. In the exchange between the estimations, the well-known concept of *extrinsic* was found to be crucial for the processing of the algorithm. In its usual form, the resulting algorithm suffer in the CS scenario from its fine-granularity because of the high number of messages that have to be processed individually [KMS$^+$12]. The expense could be reduced by exploiting the benefits of *post-processing* the extrinsic, which requires less estimations because the estimation becomes edge-independent, instead of *pre-processing* it. In

Chapter 3 it was shown that post-processing and pre-processing the extrinsic is mathematically the same for the channel-constrained estimation due to its linearity. Numerical results in Chapter 6 pointed out that the usage of post-processed extrinsic for the signal-constrained estimation do not yield a significant drawback.

The Turbo approach of Chapter 4 contrasted the message-passing approach as it denoted the factorization with least possible fragmentation, i.e., signal- and channel-constrained part oppose each other directly, resulting in two separated estimations. The derivation, again, motivated the use of extrinsic and led to the famous VAMP algorithm [RSF19] and a version of it utilizing individual variances. The estimation-theoretic analysis of the extrinsic computation revealed the connection to *bias compensation*. The optimization-based derivation linked the treatment of individual variances directly with individual bias compensation, as well as average variances and average bias compensation. For the bias compensation after the channel-constrained estimation it was proposed to cut the tie and combine individual bias compensation with the use of average variances. The algorithm with individual variances, which was directly derived from the optimization framework showed the drawback of not preventing the computation of negative variances. With the estimation-theoretic approach an adapted update was derived for the processing after the signal-constrained estimation, which guarantees positive unbiased variances. Utilizing these methods, several variants of the so far existing algorithms were proposed.

Both, the message-passing and the Turbo approach, elucidated the well-known importance of the computation of *extrinsic* in the exchange of parameters between different estimations.

The constrained optimization framework was also used to derive algorithms that process either signal components or measurements in a sequential manner in Chapter 5. Since the sequential processing of variables requires individual variances, the estimation-theoretic adaptions could be used to enhance the algorithm resulting directly from the optimization-based derivation.

In summary, four different factorizations of the posterior representing the CS problem were considered. One framework was used to provide derivations of algorithms for the different factorizations. Estimation-theoretical bias compensation was used to improve the derived algorithms.

The simulation results in Chapter 6 showed that the estimation-theoretically improved algorithms could generally beat the algorithms directly derived from the optimization-based approach in terms of compressibility and steady-state error. The model for wireless sensor networks showed a drawback of VAMP for unequally scaled sensing matrix columns, which could be overcome by individual bias compensation after the channel-constrained estimation without increasing the computational complexity significantly. The sequential algorithms have the freedom of varying the schedule of processing either the signal components or the observations. For the algorithms that process signal components sequentially, the possible schedules were examined in detail. It could be shown by numerical simulations that the sequential processing of measurements can defeat VAMP in scenarios with non-i.i.d. noise.

All in all, the alternative approaches (leading to sequential algorithms) and the estimation-theoretic enhancements turned out to be able to improve VAMP in certain specific scenarios.

Generally, on the one hand, the algorithms with individual variances play to their strengths for small signal dimensions N, where the advance to VAMP is stronger and the restraint of their high complexity does not (yet) kick in. On the other hand, one could see that the average variance strategy of VAMP inherits useful stability properties on both sides of noise and prior variances. Nevertheless, the usage of average variance can be beneficially combined with individual bias compensation.

7.2 Future Work

In order to focus on the estimation-theoretic concepts within the algorithms, the thesis was restricted to relatively basic assumptions on the scenario, such as independent and identically distributed signal components with known sparsity and measurements obtained by a known sensing matrix in Gaussian noise. Possible future work could therefore encompass to test and adapt the proposed algorithms in less restrictive scenarios, closer to real-world applications; a keyword being *block sparsity* [BCDH10] or multidimensional priors [BG19, Bir19] that model dependencies between the signal components. It would also be interesting to see how robust the algorithms are when there is a mismatch to the prior assumption [Ver10, VS13]. Additionally, also the performance in the multiple measurement vector (MMV) problem [EK12, CREK05] or the distributed compressed sensing (DCS) [SBW$^+$05] problem could be examined. Both problems work with multiple measurements, where DCS utilizes different sensing matrices per measurement vector; while the MMV problem considers the same sensing matrix in each measurement. The multiple measurements may be exploited to gain more knowledge from temporal or spatial correlations.

The algorithm, which processes measurements sequentially has not been investigated very thoroughly yet. Possible applications could be PAR reduction schemes for OFDM, which have to handle varying noise variances [KPN$^+$14, LHD20, LHL$^+$19]. Moreover, the trade-off between computing the channel-constrained estimate based on a single observation (as in the sequential algorithm), some part of the observation vector, or the entire observation vector (as in VAMP) might be worthwhile to consider, both in terms of performance and complexity.

Furthermore, the estimation-theoretic adaptions in the bias compensation strategies might be beneficially applicable in inference problems outside the CS scenario, as long as they inherit minimum mean-squared error estimators.

A. Vector Spaces

In this appendix, the notion of a vector space is discussed because the entire thesis is based on the treatment of vectors and matrices. Furthermore, the notion is used in Chapter 4 for a geometrical interpretation of estimation and bias compensation. The appendix comprises a summary of the necessary definitions, e.g., norms, and necessary properties of vectors and matrices.

A vector space $\mathcal{V}$ over a certain (scalar) field $\mathbb{F}$ is a set of objects that can be linearly combined, i.e., scaled and added, without leaving the set. Mathematically this means the set is closed under addition and multiplication with elements of the corresponding scalar field, i.e., with $a, b \in \mathbb{F}$, it holds for all $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{V}$ that $a\boldsymbol{x} + b\boldsymbol{y} \in \mathcal{V}$. The scalar field is in this thesis always the set of real numbers $\mathbb{R}$. For mathematical details and more properties, the reader is referred to [HJ09, Sec. 0.1.2].

A.1 Vectors

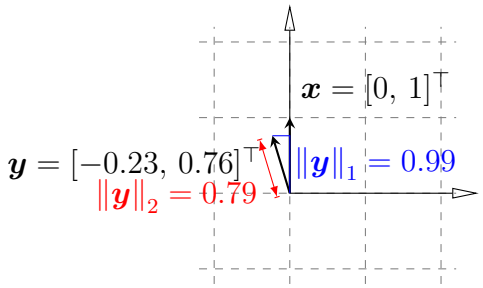
In this thesis, the objects are mostly vectors, which are represented by a list of entries, e.g.,

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} = [0,\, 1]^\top , \qquad \boldsymbol{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} -0.23 \\ 0.76 \end{bmatrix} , \tag{A.1}$$

as in Ex. 2.1. Note that vectors are represented as column vectors. Other examples for the objects are functions and thereby random variables, which is discussed in detail in Sec. 4.4.2.

The objects *span* the space. In the above example, the space is of dimension two, because there are two entries in the vectors and the vectors are linearly independent (one cannot get the other vector by simply scaling one of them). The same space is spanned by $\boldsymbol{x}$ and $[1,\, 0]^\top$ as well, which together form an *orthonormal basis*. This means that the vectors are *orthogonal*, i.e., the *inner product* is zero. The inner product of vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{V}$ of dimension N is here given by the sum of the component-wise multiplied entries

$$\boldsymbol{x}^\top \boldsymbol{y} = \sum_{j=1}^{N} x_j \cdot y_j \ . \tag{A.2}$$



$$\|\boldsymbol{x}\|_2 = 1 \qquad \|\boldsymbol{y}\|_2 = \sqrt{(-0.23)^2 + 0.76^2} = 0.79$$
$$\|\boldsymbol{x}\|_1 = 1 \qquad \|\boldsymbol{y}\|_1 = |-0.23| + |0.76| = 0.99$$
$$\|\boldsymbol{x}\|_0 = 1 \qquad \|\boldsymbol{y}\|_0 = 2$$

Figure A.1: Depiction of $\boldsymbol{x}$ and $\boldsymbol{y}$ from Ex. 2.1 in the space spanned by $\boldsymbol{x}$ and $[1,\ 0]^\top$.

In the example, it is $\boldsymbol{x}^\top \boldsymbol{y} = 0.76$ and $\boldsymbol{x}^\top [1,\ 0]^\top = 0 \cdot 1 + 1 \cdot 0 = 0$. The orthogonality of the basis makes a graphical representation convenient and the inner product allows for geometric notions as length and angles. In Fig. A.1 a coordinate system is used that is obtained by scaling the vectors $\boldsymbol{x}$ and $[1,\ 0]^\top$ to represent vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ from Ex. 2.1. The vectors are represented as arrows that point to the coordinates that define them. This shows, as already mentioned, that the vectors are defined by a direction and a length; $\boldsymbol{x}$ points in the direction of one of the coordinate axes, because the axis is defined based on it. In the following, it is considered what is meant by "length".

If the inner product is specified as in (A.2), the vector space is called a *Euclidean vector space*, because the norm corresponding to the inner product is the so-called *Euclidean norm* or ℓ_2-norm, which for is $\boldsymbol{x} \in \mathcal{V}$ (dimensionality N) defined by [GV96, HJ09]

$$\|\boldsymbol{x}\|_2 \stackrel{\text{def}}{=} \sqrt{\boldsymbol{x}^\top \boldsymbol{x}} = \sqrt{\sum_{j=1}^{N} x_j^2}\ . \tag{A.3}$$

Another example for a norm is the ℓ_1-norm or Manhattan norm, which is defined as [GV96, HJ09]

$$\|\boldsymbol{x}\|_1 \stackrel{\text{def}}{=} \sum_{j=1}^{N} |x_j|\ . \tag{A.4}$$

Both norms are special cases of the ℓ_p-norm with definition [GV96, HJ09]

$$\|\boldsymbol{x}\|_p \stackrel{\text{def}}{=} \left(\sum_{j=1}^{N} |x_j|^p \right)^{1/p}\ . \tag{A.5}$$

Noteworthy, for $p < 1$, the definition of the ℓ_p-norm does not yield a norm in the mathematical sense anymore and for $p \to 0$ the result yields the *sparsity*, i.e., the number of non-zero elements, of the vector [EK12]

$$\|\boldsymbol{x}\|_0 \stackrel{\text{def}}{=} |\{x_j \neq 0 \mid j \in \{1, \ldots, N\}\}|\ . \tag{A.6}$$

A comparison of the different norms is given in Fig. A.1 for $\boldsymbol{x}$ and $\boldsymbol{y}$ from Ex. 2.1. The Euclidean norm is usually associated as length of a vector.

Generally, the vectors of one vector space can be represented in other vector spaces, cf. [Kay93, PP02]. In Sec. 4.4.2, this is used to represent random variables in the Euclidean space. Note that the vector space of the random variables itself does not use the Euclidean norm as inner product.

A.2 Matrices

By appending vectors column-wise, one obtains matrices, e.g., in Ex. 2.1, it is

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = \begin{bmatrix} 0.75 & -0.25 \\ -0.5 & 0.75 \end{bmatrix} , \tag{A.7}$$

which consists of the vectors $\tilde{\boldsymbol{a}}_1 = [a_{11},\, a_{21}]^\top = [0.75,\, -0.5]^\top$ and $\tilde{\boldsymbol{a}}_2 = [a_{12},\, a_{22}]^\top = [-0.25,\, 0.75]^\top$.

An important property of matrices that is used in this thesis is the so-called *Frobenius norm* [GV96], which is for $\boldsymbol{A} \in \mathbb{R}^{M \times N}$ given by

$$\|\boldsymbol{A}\|_{\mathrm{F}} \overset{\text{def}}{=} \sqrt{\sum_{i=1}^{M} \sum_{j=1}^{N} a_{ij}^2} . \tag{A.8}$$

The Frobenius norm of the matrix given above is, thus,

$$\|\boldsymbol{A}\|_{\mathrm{F}} = \sqrt{2 \cdot 0.75^2 + (-0.25)^2 + (-0.5)^2} = 0.9345 . \tag{A.9}$$

A.2.1 Coherence

The *coherence* [EK12], which is mentioned in Chapters 2 and 6, is defined as

$$c(\boldsymbol{A}) = \max_{\substack{i,j \in \{1,\,\ldots,\,N\} \\ i \neq j}} \frac{|\tilde{\boldsymbol{a}}_i^\top \tilde{\boldsymbol{a}}_j|}{\|\tilde{\boldsymbol{a}}_i\|_2 \, \|\tilde{\boldsymbol{a}}_j\|_2} , \tag{A.10}$$

where $|\cdot|$ denotes the absolute value. A matrix satisfying $c(\boldsymbol{A}) = 0$ is called an *orthogonal matrix*, because it implies that every column is orthogonal to each other column; noteworthy, an orthogonal matrix needs to be square, i.e., $M = N$.

A.2.2 Condition Number

Furthermore, the notion of a condition number of a matrix is required. The condition number of $\boldsymbol{A}$ gives the chance to bound the relative error obtained when inverting the matrix to solve a system of equations $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ for $\boldsymbol{x}$ based on the relative error in $\boldsymbol{b}$ [HJ09]. In this thesis, we restrict to the condition number based on the ℓ_2-norm, so that the condition number is basically the ratio between largest and smallest singular value of the matrix. With matrix $\boldsymbol{A} \in \mathbb{R}^{M \times N}$ (where $M < N$) having singular value decomposition (SVD) given by $\boldsymbol{A} = \boldsymbol{U}\mathbf{diag}(s_j)\boldsymbol{V}^\top$, i.e., singular values $s_1,\, \ldots,\, s_M > 0$ and $s_{M+1} = \cdots = s_N = 0$, where the singular values are

sorted in descending order, meaning $s_1 \geq s_2 \geq \cdots \geq s_M$, then the condition number of $\boldsymbol{A}$ is defined as [HJ09, Sec. 5.7 Problem 1]

$$\kappa(\boldsymbol{A}) = \frac{s_1}{s_M} \, . \tag{A.11}$$

If the condition number is large, $\boldsymbol{A}$ is said to be *ill conditioned* or *poorly conditioned*, meaning that errors are generally amplified. For $\kappa(\boldsymbol{A})$ close or equal to one, one may say that $\boldsymbol{A}$ is *well conditioned*.

When constructing sensing matrices with specified condition number κ, the procedure from [RSF19, Sec. VI. A] is chosen, where the matrix $\boldsymbol{A}$ is constructed via an SVD $\boldsymbol{A} = \boldsymbol{U}\mathbf{diag}(s_j)\boldsymbol{V}^\top$ with non-zero singular values being specified by

$$s_j = \kappa^{(M-j+1)/(M-1)} \, , \quad j \in \{1, \ldots, M\} \, , \tag{A.12}$$

as well as $\boldsymbol{U} \in \mathbb{R}^{M \times M}$ and $\boldsymbol{V} \in \mathbb{R}^{N \times N}$ being drawn randomly from the group of orthogonal matrices.

B. Convex Optimization

This appendix comprises the notions of optimization theory that are necessary to follow the derivation of the algorithms in Chapter 3 through 5.

The following definitions are according to [BV04]. A set $\mathcal{A}$ is *convex* if for any two elements of the set, let's call them x_1 and x_2, and $\gamma \in [0,\, 1]$, the convex combination is in the set, i.e.,

$$\gamma x_1 + (1-\gamma)x_2 \in \mathcal{A}\ . \tag{B.1}$$

This holds as well for vector spaces, i.e., $\mathcal{X}$ is convex if for $\boldsymbol{x}_1,\, \boldsymbol{x}_2 \in \mathcal{X}$, $\gamma \in [0,\, 1]$, it is $\gamma\boldsymbol{x}_1 + (1-\gamma)\boldsymbol{x}_2 \in \mathcal{X}$. This means that line segment between $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ is always inside the set $\mathcal{X}$. A *convex function* is a function with convex domain satisfying

$$f(\gamma\boldsymbol{x}_1 + (1-\gamma)\boldsymbol{x}_2) \leq \gamma f(\boldsymbol{x}_1) + (1-\gamma)f(\boldsymbol{x}_2)\ , \tag{B.2}$$

i.e., the line segment connecting the points $f(\boldsymbol{x}_1)$ and $f(\boldsymbol{x}_2)$ is always above the function, cf. Fig. B.1. A function $f(\boldsymbol{x})$ is *concave*, if $-f(\boldsymbol{x})$ is convex.

Constrained optimization problems are usually formulated in the following way

$$\begin{aligned} &\text{minimize} && f(\boldsymbol{x}) \\ &\text{subject to} && f_k(\boldsymbol{x}) = 0\ , \quad k \in \{1,\, \ldots,\, K\}\ . \end{aligned} \tag{B.3}$$

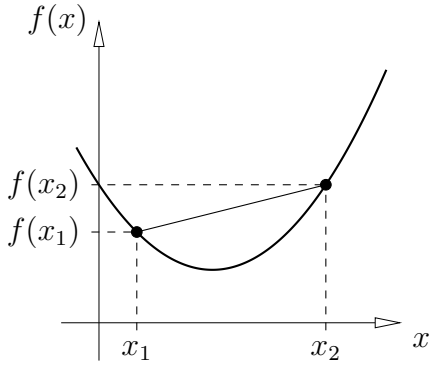


Figure B.1: Example for a convex function and the line segment according to the definition.

The function $f(\boldsymbol{x})$ is called the *objective function* or *cost function*, $\boldsymbol{x}$ is the *optimization variable*, and $f_k(\boldsymbol{x}) = 0\,,\ k \in \{1, \ldots, K\}$ are the *equality constraints*. We do not define inequality constraints since they do not appear in the thesis. In the given case, the optimization problem is called convex, if the cost function $f(\boldsymbol{x})$ and the constraints are convex.

A point $\boldsymbol{x}$ that satisfies all constraints, while also being part of the respective domains of the functions, is called *feasible*. A point that minimizes the problem (in the sense of an infimum, if the minimum can not be attained) is called *optimal point* or *optimizer* $\boldsymbol{x}^*$ and $p^* = f(\boldsymbol{x}^*)$ is the *optimal (minimal) value*.

In order to fulfill one equality constraint, an optimal point must lie where the contour line of the objective function and the respective constraint function is tangential, because otherwise that means the objective function changes along the constraint function, which in turn means the objective function can still further be minimized without violating the constraint [Gra14, HF14]. Mathematically, this can be formulated as collinearity of the gradients[1] [Gra14], meaning that the differentiation w.r.t. every possible direction of the constraint function and the objective function must be multiples of each other. If several constraints are imposed, the same reasoning leads to the insight that the gradient of the objective function must be a linear combination of the equality constraints. By introducing respective multipliers $\boldsymbol{\mu} = [\mu_1, \ldots, \mu_K]^\top$, this yields a system of equations

$$\nabla f(\boldsymbol{x}^*) \stackrel{!}{=} -\sum_{k=1}^{K} \mu_k \nabla f_k(\boldsymbol{x}^*)\ . \tag{B.4}$$

The reason for placing the minus here, stems from alternatively writing the functions on one side and setting the sum to zero, i.e.,

$$\nabla f(\boldsymbol{x}^*) + \sum_{k=1}^{K} \mu_k \nabla f_k(\boldsymbol{x}^*) \stackrel{!}{=} 0 \tag{B.5}$$

$$\Leftrightarrow \qquad \nabla \left(f(\boldsymbol{x}^*) + \sum_{k=1}^{K} \mu_k f_k(\boldsymbol{x}^*) \right) \stackrel{!}{=} 0\ . \tag{B.6}$$

Then, it is only one step left to summarizing the procedure by additively combining objective function and the (weighted) constraint functions into a new function. The result is the so-called *Lagrangian*

$$\mathcal{L}(\boldsymbol{x},\, \boldsymbol{\mu}) = f(\boldsymbol{x}) + \sum_{k=1}^{K} \mu_k f_k(\boldsymbol{x}) \tag{B.7}$$

with $\boldsymbol{\mu} = [\mu_1, \ldots, \mu_K]^\top$ being called *Lagrange multipliers*.

This way, the optimization problem is converted to finding the stationary points of the Lagrangian. An example in two dimensions is shown for illustration purposes.

Example B.1:

Here, Ex. 2.1 is extended for the purpose of showing the idea of the Lagrangian. Let $\mathrm{p}_{\boldsymbol{y}}(\boldsymbol{x}) = \frac{1}{\sqrt{(2\pi\sigma_\mathrm{n}^2)^M}} \exp\!\left(-\frac{1}{2\sigma_\mathrm{n}^2} \left\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\right\|_2^2\right)$ with the values of $\boldsymbol{A}$, $\boldsymbol{y}$, and σ_n^2 taken from Ex. 2.1

[1] Gradients summarize the differentiation with respect to the elements of the argument, i.e., if $\boldsymbol{x} \in \mathbb{R}^N$, then $\nabla f(\boldsymbol{x}) = [\frac{\partial}{\partial x_1} f(\boldsymbol{x}), \ldots, \frac{\partial}{\partial x_N} f(\boldsymbol{x})]^\top$. We may use $\nabla f(\boldsymbol{x}^*)$ as shorthand for $\nabla f(\boldsymbol{x})|_{\boldsymbol{x}=\boldsymbol{x}^*}$.

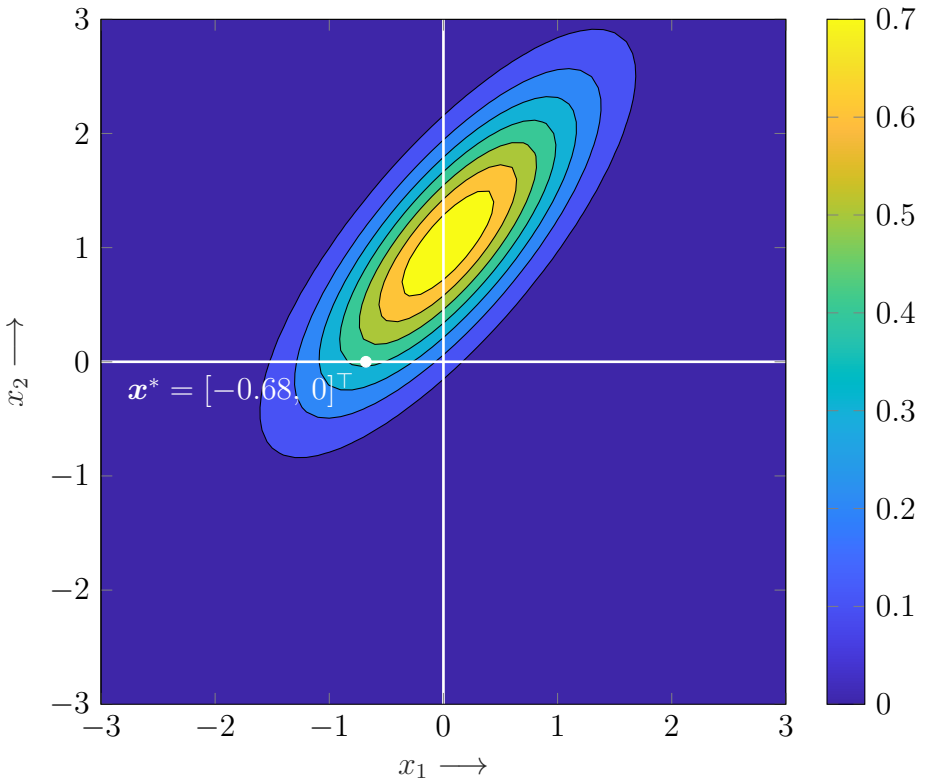


Figure B.2: Contour plot of $\mathrm{p}_{\boldsymbol{y}}(\boldsymbol{x})$ from Ex. 2.1 with constraints $x_1 = 0$ and $x_2 = 0$ as white lines and the solution $\boldsymbol{x}^* = [-0.68,\ 0]^\top$ of the minimization with constraint $x_2 = 0$.

be the negative cost function, i.e., $f(\boldsymbol{x}) = -\mathrm{p}_{\boldsymbol{y}}(\boldsymbol{x})$ should be minimized. Further, assume that it is known that $x_2 \stackrel{!}{=} 0$, so an equality constraint $f_1(\boldsymbol{x}) = x_2$ is added. $\mathrm{p}_{\boldsymbol{y}}(\boldsymbol{x})$ is shown as contour plot in Fig. B.2, the equality constraint is the horizontal white line. The Lagrangian reads $\mathcal{L}(\boldsymbol{x},\, \mu_1) = -\mathrm{p}_{\boldsymbol{y}}(\boldsymbol{x}) + \mu_1 f_1(\boldsymbol{x})$. Differentiation w.r.t. x_1, x_2, and μ_1 and subsequent zeroing yields three equations, which can be use to solve for the variables.

Differentiation w.r.t. μ_1 retrieves the constraint $f_1(\boldsymbol{x})$, i.e.,

$$\frac{\partial}{\partial \mu_1}\mathcal{L}(\boldsymbol{x}, \mu_1) = \frac{\partial}{\partial \mu_1}(-\mathrm{p}_{\boldsymbol{y}}(\boldsymbol{x}) + \mu_1 f_1(\boldsymbol{x})) = f_1(\boldsymbol{x}) = x_2 \stackrel{!}{=} 0\ . \tag{B.8}$$

Together with the other equations, the missing values of x_1 and μ_1 can be obtained.

First note that

$$\frac{\partial}{\partial x_j}\mathrm{p}_{\boldsymbol{y}}(\boldsymbol{x}) = \mathrm{p}_{\boldsymbol{y}}(\boldsymbol{x}) \cdot \left(-\frac{1}{\sigma_{\mathsf{n}}^2}\sum_{i=1}^{M}(y_i - \boldsymbol{a}_i\boldsymbol{x}) \cdot (-a_{ij})\right)\ . \tag{B.9}$$

Therefore,

$$\begin{aligned}\frac{\partial}{\partial x_1}\mathcal{L}(\boldsymbol{x},\, \mu_1) &= -\frac{\partial}{\partial x_1}\mathrm{p}_{\boldsymbol{y}}(\boldsymbol{x}) + \mu_1 \frac{\partial}{\partial x_1}x_2 \\ &= -\mathrm{p}_{\boldsymbol{y}}(\boldsymbol{x}) \cdot \left(\frac{1}{\sigma_{\mathsf{n}}^2}\sum_{i=1}^{M} a_{i1}(y_i - \boldsymbol{a}_i\boldsymbol{x})\right) + 0 \stackrel{!}{=} 0 \end{aligned} \tag{B.10}$$

$$\Rightarrow \qquad \sum_{i=1}^{M} a_{i1}(y_i - \boldsymbol{a}_i\boldsymbol{x}) \stackrel{!}{=} 0\ , \tag{B.11}$$

which yields together with $x_2 = 0$

$$a_{11}(y_1 - a_{11}x_1) + a_{21}(y_2 - a_{21}x_1) \stackrel{!}{=} 0 \tag{B.12}$$

$$\Rightarrow \qquad x_1 = \frac{a_{11}y_1 + a_{21}y_2}{a_{11}^2 + a_{21}^2} = -0.68\ . \tag{B.13}$$

With the values $x_1 = -0.68$ and $x_2 = 0$, the last equation yields the value for μ_1

$$\begin{aligned}\frac{\partial}{\partial x_2}\mathcal{L}(\boldsymbol{x},\, \mu_1) &= -\frac{\partial}{\partial x_2}\mathrm{p}_{\boldsymbol{y}}(\boldsymbol{x}) + \mu_1\frac{\partial}{\partial x_2}x_2 \\ &= -\mathrm{p}_{\boldsymbol{y}}(\boldsymbol{x}) \cdot \left(\frac{1}{\sigma_{\mathrm{n}}^2}\sum_{i=1}^{M} a_{i2}(y_i - \boldsymbol{a}_i\boldsymbol{x})\right) + \mu_1 \stackrel{!}{=} 0 \end{aligned} \tag{B.14}$$

$$\Rightarrow \qquad \mu_1 = \mathrm{p}_{\boldsymbol{y}}(\boldsymbol{x}) \cdot \left(\frac{1}{\sigma_{\mathrm{n}}^2}\sum_{i=1}^{M} a_{i2}(y_i - \boldsymbol{a}_i\boldsymbol{x})\right)\ . \tag{B.15}$$

Hence, the solution is given by $\boldsymbol{x}^* = [-0.68,\, 0]^\top$, which fulfills the equality constraint $x_2 = 0$ and maximizes $\mathrm{p}_{\boldsymbol{y}}(\boldsymbol{x})$, given the constraint. As one can see, the constraint $x_2 = 0$ and the contour lines are tangential in the solution point.

Note that $\boldsymbol{y}$ was generated by $\boldsymbol{x} = [0,\, 1]^\top$, so the constraint $x_1 = 0$ is much closer to the actual maximum of $\mathrm{p}_{\boldsymbol{y}}(\boldsymbol{x})$.

It can be shown that the optimal point *minimizes* the Lagrangian. This property is used in the *Lagrange dual function*. It is defined as

$$\mathcal{L}_{\mathrm{D}}(\boldsymbol{\mu}) = \inf_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{x},\, \boldsymbol{\mu})\ . \tag{B.16}$$

Noteworthy, the dual function is concave, even if the problem itself is not convex. This concavity ensures that the dual function yields lower bounds on the optimal value p^*, i.e.,

$$\mathcal{L}_{\mathrm{D}}(\boldsymbol{\mu}) \leq p^*\ . \tag{B.17}$$

If the optimal value can be attained, i.e.,

$$\max_{\boldsymbol{\mu}} \mathcal{L}_{\mathrm{D}}(\boldsymbol{\mu}) = p^*\ , \tag{B.18}$$

one speaks of *strong duality*, otherwise of *weak duality*. Note that this leads the way for an approach to solve the optimization problem from a different point of view. The resulting formulation of the problem is called *Lagrange dual problem*

$$\text{maximize} \quad \mathcal{L}_{\mathrm{D}}(\boldsymbol{\mu})\ .$$

Noteworthy, *Slater's condition* states that the optimal value can be attained via this approach, i.e., (B.18) holds in this case, if there exists a feasible point.

C. Functional Derivatives

A *functional* describes a function which has itself functions as arguments; the differential entropy (2.23) is an example for a functional, since it depends on the form of the density $\mathsf{q}(\boldsymbol{x})$. The term *functional derivative* or *variational derivative* stems from the *calculus of variations* [GFS63], which addresses the search for extrema in such functionals.

Here, functionals of the form

$$F(\mathsf{q}(\boldsymbol{x})) = \int f(\mathsf{q}(\boldsymbol{x}),\, \boldsymbol{x})\, \mathrm{d}\boldsymbol{x} \tag{C.1}$$

are considered, with $f(\cdot\,,\,\cdot)$ being a function of the two variables. The task of variational calculus is to find the function $\mathsf{q}(\boldsymbol{x})$ that maximizes or minimizes the functional. For the derivation of the form of the functional derivative in this case, the necessary conditions to obtain an extremum in such a functional are described. The explanations follow the exposition given in [Gra14] as it focuses on a rather intuitive understanding. An entirely mathematical description is given in [GFS63].

In order to derive conditions for an extremum of such a functional, one considers the optimal function $\mathsf{q}^*(\boldsymbol{x})$ and a *variation* $\delta\mathsf{q}(\boldsymbol{x})$ thereof, i.e., any solution $\mathsf{q}(\boldsymbol{x})$ may be represented as

$$\mathsf{q}(\boldsymbol{x}) = \mathsf{q}^*(\boldsymbol{x}) + \epsilon\delta\mathsf{q}(\boldsymbol{x})\ , \tag{C.2}$$

with scalar parameter ϵ quantifying the amount of variation. Obviously, for $\epsilon \to 0$, the optimal solution is obtained. This also means that at an extremum, the derivative w.r.t. ϵ must vanish, as otherwise $\mathsf{q}^*(\boldsymbol{x})$ would not be the optimal solution. Mathematically, it is

$$\delta F \overset{\text{def}}{=} \frac{\mathrm{d}F(\mathsf{q}^*(\boldsymbol{x}) + \epsilon\delta\mathsf{q}(\boldsymbol{x}))}{\mathrm{d}\epsilon}\bigg|_{\epsilon=0} \overset{!}{=} 0\ , \tag{C.3}$$

where δF is called the *(first) variation* or *differential*. By use of the chain rule[1] the differentia-

[1]The chain rule states that a derivative of nested functions $f(a(\epsilon))$ w.r.t. the variable ϵ can be computed by differentiating the outer function $f(\cdot)$ w.r.t. the inner function as argument and multiplying it to the derivative of the inner function $a(\epsilon)$, i.e., $\frac{\mathrm{d}}{\mathrm{d}\epsilon}f(a(\epsilon)) = \frac{\mathrm{d}}{\mathrm{d}a}f(a) \cdot \frac{\mathrm{d}}{\mathrm{d}\epsilon}a(\epsilon)$.

tion w.r.t. ϵ yields in the given case

$$\begin{aligned}\delta F &= \int \frac{\mathrm{d}}{\mathrm{d}\epsilon} f(\mathsf{q}^*(\boldsymbol{x}) + \epsilon\delta\mathsf{q}(\boldsymbol{x}),\, \boldsymbol{x})\Big|_{\epsilon=0} \mathrm{d}\boldsymbol{x} \\ &= \int \frac{\partial}{\partial \mathsf{q}} f(\mathsf{q}(\boldsymbol{x}),\, \boldsymbol{x}) \cdot \frac{\mathrm{d}}{\mathrm{d}\epsilon}(\epsilon\delta\mathsf{q}(\boldsymbol{x}))\, \mathrm{d}\boldsymbol{x} \\ &= \int \frac{\partial}{\partial \mathsf{q}} f(\mathsf{q}(\boldsymbol{x}),\, \boldsymbol{x})\delta\mathsf{q}(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x}\ . \end{aligned} \tag{C.4}$$

At this point, one needs to make use of the fundamental lemma of the calculus of variations [GFS63, Lemma 1], which states that for every continuous function $\delta\mathsf{q}(\boldsymbol{x})$ from

$$\int_{\mathcal{X}} f(\boldsymbol{x})\delta\mathsf{q}(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} = 0 \tag{C.5}$$

it follows that $f(\boldsymbol{x}) = 0$ for any $\boldsymbol{x} \in \mathcal{X}$.

Since $\delta F = 0$ is required for all variations $\delta\mathsf{q}(\boldsymbol{x})$, it can be deduced that

$$\frac{\partial}{\partial \mathsf{q}} f(\mathsf{q}(\boldsymbol{x}),\, \boldsymbol{x}) \stackrel{!}{=} 0 \tag{C.6}$$

must hold at an extremum, i.e., the integral can be dropped for these considerations. Ultimately, the left hand-side of (C.6) yields the desired way of computing the functional derivative.

Noteworthy, the differentiation in (C.6) is carried out as if $\mathsf{q}(\boldsymbol{x})$ were a simple variable instead of a function. Hence, the classical differentiation rules as sum, product, and chain rule, hold as usual. All in all, the procedure to differentiate a functional as in (C.1) w.r.t. $\mathsf{q}(\boldsymbol{x})$ is to drop the integral and differentiate the function(al) inside as if $\mathsf{q}(\boldsymbol{x})$ were a variable (not a function). In order to get familiar with this type of differentiation, the functional derivative of the differential entropy (2.23) and the Kullback–Leibler divergence (2.21) are computed exemplary.

Example C.1:

Recall that the differential entropy is defined as

$$\mathrm{h}(\mathsf{q}(\boldsymbol{x})) = -\int \mathsf{q}(\boldsymbol{x}) \log \mathsf{q}(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x}\ , \tag{C.7}$$

i.e., it is $f(\mathsf{q}(\boldsymbol{x}),\, \boldsymbol{x}) = -\mathsf{q}(\boldsymbol{x}) \log \mathsf{q}(\boldsymbol{x})$. Hence,

$$\frac{\partial \mathrm{h}}{\partial \mathsf{q}} = -\frac{\partial}{\partial \mathsf{q}} \mathsf{q}(\boldsymbol{x}) \log \mathsf{q}(\boldsymbol{x}) = -1 - \log \mathsf{q}(\boldsymbol{x})\ . \tag{C.8}$$

Example C.2:

The Kullback–Leibler divergence is defined as

$$\mathrm{D}_{\mathrm{KL}}(\mathsf{q}(\boldsymbol{x}) \,||\, \mathsf{f}_{\mathsf{x}}(\boldsymbol{x})) = \int \mathsf{q}(\boldsymbol{x}) \log \frac{\mathsf{q}(\boldsymbol{x})}{\mathsf{f}_{\mathsf{x}}(\boldsymbol{x})}\, \mathrm{d}\boldsymbol{x}\ , \tag{C.9}$$

i.e., $f(\mathsf{q}(\boldsymbol{x}),\, \boldsymbol{x}) = \mathsf{q}(\boldsymbol{x}) \log \frac{\mathsf{q}(\boldsymbol{x})}{\mathsf{f}_{\mathsf{x}}(\boldsymbol{x})}$ and, thus,

$$\frac{\partial}{\partial \mathsf{q}} \mathrm{D}_{\mathrm{KL}}(\mathsf{q}(\boldsymbol{x}) \,||\, \mathsf{f}_{\mathsf{x}}(\boldsymbol{x})) = \frac{\partial}{\partial \mathsf{q}} \left(\mathsf{q}(\boldsymbol{x}) \log \mathsf{q}(\boldsymbol{x}) - \mathsf{q}(\boldsymbol{x}) \log \mathsf{f}_{\mathsf{x}}(\boldsymbol{x})\right) = 1 + \log \mathsf{q}(\boldsymbol{x}) - \log \mathsf{f}_{\mathsf{x}}(\boldsymbol{x})\ . \tag{C.10}$$

D. Exponential Families

Exponential families denote a class of distributions which are entirely described by certain moments. The moments that specify the respective distribution are collected in the so-called *sufficient statistics* $\boldsymbol{g}(\boldsymbol{x})$. One can see the sufficient statistic as measure that needs to be known in order to fully characterize a given distribution [Fis22, PP02]. To point out the importance of the sufficient statistics, we derive the form of an exponential family as result of the minimization of the Kullback–Leibler divergence under a moment constraint.

D.1 Projection Property of the Kullback–Leibler Divergence

Throughout the thesis, the minimization of the Kullback–Leibler divergence plays an important role. In this section, we show the strong connection between exponential families and the minimization of the Kullback–Leibler divergence under a moment constraint by considering a constrained optimization problem and especially its Lagrangian. The task is to approximate an arbitrary distribution $\mathsf{f}_{\mathsf{x}}(\boldsymbol{x})$ by a substitute distribution $\mathsf{q}(\boldsymbol{x})$, while being consistent to the moments $\mathrm{E}_{\mathsf{x}\sim\mathsf{f}_{\mathsf{x}}}\{\boldsymbol{g}(\mathsf{x})\}$, i.e., a constrained optimization problem

$$\begin{aligned} &\text{minimize} && \mathrm{D}_{\mathrm{KL}}(\mathsf{q}(\boldsymbol{x}) \,||\, \mathsf{f}_{\mathsf{x}}(\boldsymbol{x})) \\ &\text{subject to} && \mathrm{E}_{\mathsf{x}\sim\mathsf{q}}\{\boldsymbol{g}(\mathsf{x})\} = \mathrm{E}_{\mathsf{x}\sim\mathsf{f}_{\mathsf{x}}}\{\boldsymbol{g}(\mathsf{x})\} \\ &&& \int \mathsf{q}(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} = 1 \end{aligned} \tag{D.1}$$

is given. The respective Lagrangian with multipliers $\boldsymbol{\theta}$ for the moment constraint and ν for the normalization constraint reads

$$\begin{aligned} \mathcal{L}(\mathsf{q}(\boldsymbol{x}),\, \boldsymbol{\theta},\, \nu) = \mathrm{D}_{\mathrm{KL}}(\mathsf{q}(\boldsymbol{x}) \,||\, \mathsf{f}_{\mathsf{x}}(\boldsymbol{x})) + \boldsymbol{\theta}^{\top} \left(\mathrm{E}_{\mathsf{x}\sim\mathsf{q}}\{\boldsymbol{g}(\mathsf{x})\} - \mathrm{E}_{\mathsf{x}\sim\mathsf{f}_{\mathsf{x}}}\{\boldsymbol{g}(\mathsf{x})\}\right) \\ + \nu \left(\int \mathsf{q}(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} - 1 \right) . \end{aligned} \tag{D.2}$$

For the functional derivative with respect to $\mathsf{q}(\boldsymbol{x})$, one gets

$$\frac{\partial \mathcal{L}}{\partial \mathsf{q}} = \log \mathsf{q}(\boldsymbol{x}) + 1 - \log \mathsf{f}_{\mathsf{x}}(\boldsymbol{x}) + \boldsymbol{\theta}^{\top} \boldsymbol{g}(\boldsymbol{x}) + \nu \stackrel{!}{=} 0 \,. \tag{D.3}$$

Hence, the form of the substitute distribution that minimizes the Kullback-Leibler divergence under the moment constraint is given by

$$\mathsf{q}(\boldsymbol{x}) = \mathsf{f}_{\mathsf{x}}(\boldsymbol{x}) \exp\Big(\boldsymbol{\theta}^{\top} \boldsymbol{g}(\boldsymbol{x}) - 1 - \nu\Big) \ . \tag{D.4}$$

For $\mathsf{q}(\boldsymbol{x})$ to fulfill the normalization constraint, the constant terms together with the multiplier ν must form the following equality

$$Z(\boldsymbol{\theta}) \stackrel{\text{def}}{=} \exp(1+\nu) = \int \mathsf{f}_{\mathsf{x}}(\boldsymbol{x}) \exp\Big(\boldsymbol{\theta}^{\top} \boldsymbol{g}(\boldsymbol{x})\Big) \, \mathrm{d}\boldsymbol{x} \tag{D.5}$$

and is called *partition function*.

All in all, an exponential family $\mathsf{f}_{\boldsymbol{\theta}}(\boldsymbol{x})$ is represented as[1]

$$\mathsf{f}_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{\theta})} \exp\Big(\boldsymbol{\theta}^{\top} \boldsymbol{g}(\boldsymbol{x}) + k(\boldsymbol{x})\Big) \ . \tag{D.6}$$

The elements of the Lagrangian multiplier $\boldsymbol{\theta}$ are called *natural parameters*. The pdf $\mathsf{f}_{\mathsf{x}}(\boldsymbol{x})$ is called *base measure* or *carrier density*, $\log \mathsf{f}_{\mathsf{x}}(\boldsymbol{x})$ is called *carrier measure* [NG09].

This also means that minimizing the Kullback–Leibler divergence of $\mathsf{q}(\boldsymbol{x})$ with respect to $\mathsf{f}_{\mathsf{x}}(\boldsymbol{x})$ forces the substitute distribution to be a member of an exponential family, if a moment constraint is given; the family is specified by the respective moments.

Note that the Boltzmann distribution (2.32) is also an exponential family with sufficient statistics $-U(\boldsymbol{x})$ and natural parameter β.

D.2 Sufficient Statistics

It has been shown above that the sufficient statistic plays an important role for an exponential family, as it specifies the measure that needs to be known for full characterization of the distribution. Since the thesis mainly works with the exponential family of Gaussian distributions, the corresponding specifications of the sufficient statistics are displayed here. For the first-order moments, we define

$$\boldsymbol{g}_{\boldsymbol{\lambda}}(\boldsymbol{x}) = \quad [x_1, \ldots, x_N]^{\top} \ , \qquad \text{i.e.,} \quad g_{\lambda_j}(x_j) = x_j \ , \tag{D.7}$$

with corresponding natural parameters

$$\boldsymbol{\lambda} = \quad [\lambda_1, \ldots, \lambda_N]^{\top} \ , \tag{D.8}$$

while for the second-order moment, either a vector parameter $\boldsymbol{\Lambda} = [\Lambda_1, \ldots, \Lambda_N]^{\top}$ with sufficient statistics

$$\boldsymbol{g}_{\boldsymbol{\Lambda}}(\boldsymbol{x}) = -\frac{1}{2}[x_1^2, \ldots, x_N^2]^{\top} = -\frac{1}{2}\mathbf{diag}\Big(\boldsymbol{x}\boldsymbol{x}^{\top}\Big) \ , \qquad \text{i.e.,} \quad g_{\Lambda_j}(x_j) = -\frac{1}{2}x_j^2 \tag{D.9}$$

[1] Use the definition $k(\boldsymbol{x}) \stackrel{\text{def}}{=} \log \mathsf{f}_{\mathsf{x}}(\boldsymbol{x})$ for consistency to the last representation

Table D.1: Summary of the parameter connections for Gaussian densities with individual and average variances.

	individual variances	average variances
$\boldsymbol{g}(\boldsymbol{x})$	$[\boldsymbol{x}^\top, -\frac{1}{2}\mathbf{diag}(\boldsymbol{x}\boldsymbol{x}^\top)^\top]^\top$	$[\boldsymbol{x}^\top, -\frac{1}{2}\boldsymbol{x}^\top\boldsymbol{x}]^\top$
$\boldsymbol{\lambda}$	$\boldsymbol{\Phi}_{\mathsf{x}}^{-1}\boldsymbol{m}_{\mathsf{x}}$	$\boldsymbol{m}_{\mathsf{x}}/\sigma_{\mathsf{x}}^2$
λ_j	$m_{\mathsf{x},j}/\sigma_{\mathsf{x},j}^2$	$m_{\mathsf{x},j}/\sigma_{\mathsf{x}}^2$
Λ_j	$1/\sigma_{\mathsf{x},j}^2$	–
Λ	–	$1/\sigma_{\mathsf{x}}^2$

or, alternatively, a single parameter Λ with corresponding statistics function

$$g_\Lambda(\boldsymbol{x}) = -\frac{1}{2}\sum_{j=1}^{N} x_j^2 = -\frac{1}{2}\boldsymbol{x}^\top\boldsymbol{x} \tag{D.10}$$

is utilized.

Together, the individual variances case is given by

$$\boldsymbol{g}(\boldsymbol{x}) = [\boldsymbol{g}_{\boldsymbol{\lambda}}(\boldsymbol{x})^\top, \boldsymbol{g}_{\boldsymbol{\Lambda}}(\boldsymbol{x})^\top]^\top\,, \quad \text{with} \quad \boldsymbol{\theta} = [\boldsymbol{\lambda}^\top, \boldsymbol{\Lambda}^\top]^\top \tag{D.11}$$

and in the average variance case

$$\boldsymbol{g}(\boldsymbol{x}) = [\boldsymbol{g}_{\boldsymbol{\lambda}}(\boldsymbol{x})^\top, g_\Lambda(\boldsymbol{x})]^\top\,, \quad \text{with} \quad \boldsymbol{\theta} = [\boldsymbol{\lambda}^\top, \Lambda]^\top\,. \tag{D.12}$$

The representation of Gaussian distributions as exponential families and the connection between the different parameterizations are explained in the examples below. The results are summarized in Table D.1.

Example D.1:

Consider the N-dimensional Gaussian random vector $\mathsf{x} \sim \mathcal{N}(\boldsymbol{m}_{\mathsf{x}}, \sigma_{\mathsf{x}}^2\boldsymbol{I})$ with mean $\boldsymbol{m}_{\mathsf{x}}$ and (isotropic) variance σ_{x}^2, i.e., the elements of x are independent but share the same variance. The probability density function reads

$$\mathsf{f}_{\mathsf{x}}(\boldsymbol{x}) = \frac{1}{\sqrt{(2\pi\sigma_{\mathsf{x}}^2)^N}}\exp\Big(-\frac{1}{2\sigma_{\mathsf{x}}^2}(\boldsymbol{x}-\boldsymbol{m}_{\mathsf{x}})^\top(\boldsymbol{x}-\boldsymbol{m}_{\mathsf{x}})\Big)\,. \tag{D.13}$$

The density can be expressed as exponential family with sufficient statistics $\boldsymbol{g}(\boldsymbol{x}) = [\boldsymbol{x}^\top, -\frac{1}{2}\boldsymbol{x}^\top\boldsymbol{x}]^\top$ and natural parameters $\boldsymbol{\lambda} = \boldsymbol{m}_{\mathsf{x}}/\sigma_{\mathsf{x}}^2$ and $\Lambda = \frac{1}{\sigma_{\mathsf{x}}^2}$, as well as partition function $Z(\boldsymbol{\lambda}, \Lambda) = \sqrt{(2\pi\sigma_{\mathsf{x}}^2)^N}\exp\Big(\frac{1}{2\Lambda}\boldsymbol{\lambda}^\top\boldsymbol{\lambda}\Big)$, since

$$\begin{aligned}
-\frac{1}{2\sigma_{\mathsf{x}}^2}(\boldsymbol{x}-\boldsymbol{m}_{\mathsf{x}})^\top(\boldsymbol{x}-\boldsymbol{m}_{\mathsf{x}}) &= \boldsymbol{m}_{\mathsf{x}}^\top\boldsymbol{x}/\sigma_{\mathsf{x}}^2 - \frac{1}{2\sigma_{\mathsf{x}}^2}\Big(\boldsymbol{m}_{\mathsf{x}}^\top\boldsymbol{m}_{\mathsf{x}} + \boldsymbol{x}^\top\boldsymbol{x}\Big) \\
&= \frac{1}{\sigma_{\mathsf{x}}^2}\boldsymbol{m}_{\mathsf{x}}^\top\boldsymbol{x} - \frac{1}{2}\left(\frac{1}{\sigma_{\mathsf{x}}^2}\boldsymbol{m}_{\mathsf{x}}^\top\frac{\sigma_{\mathsf{x}}^2}{\sigma_{\mathsf{x}}^2}\boldsymbol{m}_{\mathsf{x}} + \frac{1}{\sigma_{\mathsf{x}}^2}\boldsymbol{x}^\top\boldsymbol{x}\right) \\
&= \boldsymbol{\lambda}^\top\boldsymbol{x} - \frac{1}{2\Lambda}\boldsymbol{\lambda}^\top\boldsymbol{\lambda} - \frac{\Lambda}{2}\boldsymbol{x}^\top\boldsymbol{x}\,.
\end{aligned} \tag{D.14}$$

Using the denomination for exponential families, the pdf becomes

$$\mathsf{f}_{\boldsymbol{\lambda},\Lambda}(\boldsymbol{x}) = \frac{1}{Z(\boldsymbol{\lambda},\Lambda)} \exp\Big(\boldsymbol{\lambda}^\top \boldsymbol{x} - \frac{\Lambda}{2}\boldsymbol{x}^\top \boldsymbol{x}\Big) \ . \tag{D.15}$$

Example D.2:

Here, the N-dimensional Gaussian random vector $\mathbf{x} \sim \mathcal{N}(\boldsymbol{m}_\mathsf{x}, \boldsymbol{\Phi}_\mathsf{x})$ with mean $\boldsymbol{m}_\mathsf{x}$ and covariance matrix $\boldsymbol{\Phi}_\mathsf{x} = \mathbf{diag}(\sigma^2_{\mathsf{x},j})$ is considered, i.e., the elements of $\mathbf{x}$ are independent but not identically distributed, since each has an individual variance. The probability density function reads

$$\mathsf{f}_\mathbf{x}(\boldsymbol{x}) = \frac{1}{\sqrt{(2\pi)^N \det \boldsymbol{\Phi}_\mathsf{x}}} \exp\Big(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{m}_\mathsf{x})^\top \boldsymbol{\Phi}_\mathsf{x}^{-1}(\boldsymbol{x}-\boldsymbol{m}_\mathsf{x})\Big) \ . \tag{D.16}$$

With sufficient statistics $\boldsymbol{g}(\boldsymbol{x}) = [\boldsymbol{x}^\top, -\frac{1}{2}\mathbf{diag}(\boldsymbol{x}\boldsymbol{x}^\top)^\top]^\top$ and natural parameters $\boldsymbol{\lambda} = \boldsymbol{\Phi}_\mathsf{x}^{-1}\boldsymbol{m}_\mathsf{x}$ and $\boldsymbol{\Lambda} = \mathbf{diag}(\boldsymbol{\Phi}_\mathsf{x}^{-1}) = [(\sigma^2_{\mathsf{x},1})^{-1}, \ldots, (\sigma^2_{\mathsf{x},N})^{-1}]^\top$, i.e., $\lambda_j = m_{\mathsf{x},j}/\sigma^2_{\mathsf{x},j}$, respectively $\Lambda_j = 1/\sigma^2_{\mathsf{x},j}$. The partition function reads in this case $Z(\boldsymbol{\lambda},\boldsymbol{\Lambda}) = \sqrt{(2\pi)^N \det \boldsymbol{\Phi}_\mathsf{x}} \exp\Big(\frac{1}{2}\boldsymbol{m}_\mathsf{x}^\top \boldsymbol{\lambda}\Big)$. The reformulation of the exponent reads in this case

$$\begin{aligned} -\frac{1}{2}(\boldsymbol{x}-\boldsymbol{m}_\mathsf{x})^\top \boldsymbol{\Phi}_\mathsf{x}^{-1}(\boldsymbol{x}-\boldsymbol{m}_\mathsf{x}) &= \boldsymbol{m}_\mathsf{x}^\top \boldsymbol{\Phi}_\mathsf{x}^{-1}\boldsymbol{x} - \frac{1}{2}\Big(\boldsymbol{m}_\mathsf{x}^\top \boldsymbol{\Phi}_\mathsf{x}^{-1}\boldsymbol{m}_\mathsf{x} + \boldsymbol{x}^\top \boldsymbol{\Phi}_\mathsf{x}^{-1}\boldsymbol{x}\Big) \\ &= \boldsymbol{\lambda}^\top \boldsymbol{x} - \frac{1}{2}\Big(\boldsymbol{m}_\mathsf{x}^\top \boldsymbol{\lambda} + \boldsymbol{\Lambda}^\top \mathbf{diag}\Big(\boldsymbol{x}\boldsymbol{x}^\top\Big)\Big) \ . \end{aligned} \tag{D.17}$$

D.3 Derivative of the Log-Partition Function

One very helpful property of exponential families is that the differentiation of the logarithm of partition function $Z(\boldsymbol{\theta})$ (usually called log-partition function) with respect to the natural parameters $\boldsymbol{\theta}$ yields the moments (specified by the sufficient statistics) of the exponential family [WJ08]

$$\begin{aligned} \nabla \log Z(\boldsymbol{\theta}) &= \frac{\partial}{\partial \boldsymbol{\theta}} \log \int \exp\Big(\boldsymbol{\theta}^\top \boldsymbol{g}(\boldsymbol{x}) + k(\boldsymbol{x})\Big) \, \mathrm{d}\boldsymbol{x} \\ &= \frac{1}{Z(\boldsymbol{\theta})} \int \boldsymbol{g}(\boldsymbol{x}) \exp\Big(\boldsymbol{\theta}^\top \boldsymbol{g}(\boldsymbol{x}) + k(\boldsymbol{x})\Big) \, \mathrm{d}\boldsymbol{x} \\ &= \int \boldsymbol{g}(\boldsymbol{x}) \mathsf{f}_{\boldsymbol{\theta}}(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} = \mathrm{E}_{\mathbf{x}\sim\mathsf{f}_{\boldsymbol{\theta}}}\{\boldsymbol{g}(\mathbf{x})\} \ . \end{aligned} \tag{D.18}$$

E. Reconstruction Algorithms

On the following pages, all algorithms that are used within the thesis are summarized in pseudo code notation. The stopping criterion, mentioned in the algorithms below, can, e.g., be realized by comparing the squared (Euclidean) distance between the estimates $\boldsymbol{m}_{\mathrm{c}}$ and $\boldsymbol{m}_{\mathrm{s}}$ to a given threshold ϵ, i.e.,

$$\|\boldsymbol{m}_{\mathrm{c}} - \boldsymbol{m}_{\mathrm{s}}\|_2^2 \leq \epsilon \ . \tag{E.1}$$

In the simulations for this thesis, the algorithms always run for a given number of iterations before the result is evaluated.

E.1 Gaussian Message Passing (GMP)

This algorithm is close to an algorithm stated in [KMS+12] and results from the fine-granular view on the factor graph. Note that the extrinsic is here pre-processed. The notation here utilizes means and variances for comprehensibility; using natural parameters (scaled means and inverse variances) where appropriate instead can lead to better computational complexity, because some (scalar) inversions can be saved.

Algorithm E.1: $\boldsymbol{m}_\mathrm{s} = \mathtt{GMP}(\boldsymbol{y},\, \boldsymbol{A},\, \sigma_\mathsf{n}^2)$

$m_{\mathrm{s},j}^{\backslash i} = 0$, $\sigma_{\mathrm{s},j}^{2,\backslash i} = s/N \;\forall\, j \in \{1, \ldots, N\}, i \in \{1, \ldots, M\}$ `// initialization`

$m_{\mathrm{c},i,j}^{\backslash j} = 0$, $\sigma_{\mathrm{c},i,j}^{2,\backslash j} = s/N \;\forall\, j \in \{1, \ldots, N\}, i \in \{1, \ldots, M\}$

while *stopping criterion not met* **do**

  Choose $j \in \{1, \ldots, N\}$ and $i \in \{1, \ldots, M\}$

  $m_{\mathrm{c},i,j}^{\backslash j} = (y_i - \sum_{\substack{j'=1 \\ j' \neq j}}^{N} a_{ij'} m_{\mathrm{s},j'}^{\backslash i})/a_{ij}$ `// ‘‘upward’’ messages`

  $\sigma_{\mathrm{c},i,j}^{2,\backslash j} = (\sigma_\mathsf{n}^2 + \sum_{\substack{j'=1 \\ j' \neq j}}^{N} \sigma_{\mathrm{s},j'}^{2,\backslash i} a_{ij'}^2)/a_{ij}^2$

  $\tilde{\sigma}_{\mathrm{s},j}^{2,\backslash i} = \left(\sum_{\substack{i'=1 \\ i' \neq i}}^{M} 1/\sigma_{\mathrm{c},i',j}^{2,\backslash j} \right)^{-1}$ `// input to NLMMSE (edge-dependent)`

  $\tilde{x}_{\mathrm{s},j}^{\backslash i} = \tilde{\sigma}_{\mathrm{s},j}^{2,\backslash i} \sum_{\substack{i'=1 \\ i' \neq i}}^{M} m_{\mathrm{c},i',j}^{\backslash j} / \sigma_{\mathrm{c},i',j}^{2,\backslash j}$

  $m_{\mathrm{s},j}^{\backslash i} = \mathrm{E}_{\mathsf{x}_j}\{\mathsf{x}_j \mid \tilde{x}_{\mathrm{s},j}^{\backslash i},\, \tilde{\sigma}_{\mathrm{s},j}^{2,\backslash i}\}$ `// ‘‘downward’’ messages`

  $\sigma_{\mathrm{s},j}^{2,\backslash i} = \mathrm{E}_{\mathsf{x}_j}\{(\mathsf{x}_j - m_{\mathrm{s},j}^{\backslash i})^2 \mid \tilde{x}_{\mathrm{s},j}^{\backslash i},\, \tilde{\sigma}_{\mathrm{s},j}^{2,\backslash i}\}$

$\tilde{\sigma}_{\mathrm{s},j}^2 = \left(\sum_{i=1}^{M} 1/\sigma_{\mathrm{c},i',j}^{2,\backslash j} \right)^{-1} \quad \forall\, j \in \{1, \ldots, N\}$

$\tilde{x}_{\mathrm{s},j} = \tilde{\sigma}_{\mathrm{s},j}^2 \sum_{i=1}^{M} m_{\mathrm{c},i',j}^{\backslash j} / \sigma_{\mathrm{c},i',j}^{2,\backslash j} \quad \forall\, j \in \{1, \ldots, N\}$

$\boldsymbol{m}_\mathrm{s} = \mathrm{E}_{\mathsf{x}}\{\mathsf{x} \mid \tilde{\boldsymbol{x}}_\mathrm{s},\, \tilde{\sigma}_{\mathrm{s},1}^2, \ldots, \tilde{\sigma}_{\mathrm{s},N}^2\}$ `// NLMMSE`

E.2 GMP with Post-processed Extrinsic

In contrast to the previous algorithm, an algorithm that computes the extrinsics via post-processing is shown, which is computationally much simpler because the estimates do not have to be calculated per edge ($M \cdot N$), but per factor node ($M + N$). Here, scaled means $\lambda_{ij}^{\backslash j}$ and inverse variances $\Lambda_{ij}^{\backslash j}$ are used as upward messages, because they are summed up in the next step, i.e., a conversion to mean and variance would have to be converted back immediately.

Algorithm E.2: $\boldsymbol{m}_\mathrm{s} = \texttt{GMPpostExt}(\boldsymbol{y}, \boldsymbol{A}, \sigma_\mathsf{n}^2)$

$m_{\mathrm{s},j}^{\backslash i} = 0$, $\sigma_{\mathrm{s},j}^{2,\backslash i} = s/N \;\forall\, j \in \{1, \ldots, N\}, i \in \{1, \ldots, M\}$ `// initialization`

$m_{\mathrm{c},i,j}^{\backslash j} = 0$, $\sigma_{\mathrm{c},i,j}^{2,\backslash j} = s/N \;\forall\, j \in \{1, \ldots, N\}, i \in \{1, \ldots, M\}$

while *stopping criterion not met* **do**

  for $i \in \{1, \ldots, M\}$ **do**

    $\boldsymbol{m}_{\mathrm{c},i} = \boldsymbol{m}_\mathrm{s}^{\backslash i} + \mathbf{diag}(\sigma_{\mathrm{s},j}^{2,\backslash i})\boldsymbol{a}_i(y_i - \boldsymbol{a}_i^\top \boldsymbol{m}_\mathrm{s}^{\backslash i})/(\sigma_\mathsf{n}^2 + \sum_{j'=1}^N \sigma_{\mathrm{s},j'}^{2,\backslash i} a_{ij'}^2)$

    for $j \in \{1, \ldots, N\}$ **do**

      $\sigma_{\mathrm{c},i,j}^2 = \sigma_{\mathrm{s},j}^{2,\backslash i}(1 - \sigma_{\mathrm{s},j}^{2,\backslash i} a_{ij}^2/(\sigma_\mathsf{n}^2 + \sum_{j'=1}^N \sigma_{\mathrm{s},j'}^{2,\backslash i} a_{ij'}^2))$

      $\Lambda_{ij}^{\backslash j} = 1/\sigma_{\mathrm{c},i,j}^2 - 1/\sigma_{\mathrm{s},j}^{2,\backslash i}$ `// "upward" messages`

      $\lambda_{ij}^{\backslash j} = m_{\mathrm{c},i,j}/\sigma_{\mathrm{c},i,j}^2 - m_{\mathrm{s},j}^{\backslash i}/\sigma_{\mathrm{s},j}^{2,\backslash i}$

  for $j \in \{1, \ldots, N\}$ **do**

    $\tilde{\sigma}_{\mathrm{s},j}^2 = \left(\sum_{i'=1}^M \Lambda_{i'j}^{\backslash j}\right)^{-1}$ `// input to NLMMSE (not edge-dependent)`

    $\tilde{x}_{\mathrm{s},j} = \tilde{\sigma}_{\mathrm{s},j}^2 \sum_{i'=1}^M \lambda_{i'j}^{\backslash j}$

    $m_{\mathrm{s},j} = \mathrm{E}_{\mathsf{x}_j}\{\mathsf{x}_j \mid \tilde{x}_{\mathrm{s},j}, \tilde{\sigma}_{\mathrm{s},j}^2\}$ `// NLMMSE`

    $\sigma_{\mathrm{s},j}^2 = \mathrm{E}_{\mathsf{x}_j}\{(\mathsf{x}_j - m_{\mathrm{s},j})^2 \mid \tilde{x}_{\mathrm{s},j}, \tilde{\sigma}_{\mathrm{s},j}^2\}$

    for $i \in \{1, \ldots, M\}$ **do**

      $\sigma_{\mathrm{s},j}^{2,\backslash i} = (1/\sigma_{\mathrm{s},j}^2 - \Lambda_{ij}^{\backslash j})^{-1}$ `// "downward" messages`

      $m_{\mathrm{s},j}^{\backslash i} = \sigma_{\mathrm{s},j}^{2,\backslash i}(m_{\mathrm{s},j}/\sigma_{\mathrm{s},j}^2 - \lambda_{ij}^{\backslash j})$

$\boldsymbol{m}_\mathrm{s} = [m_{\mathrm{s},1}, \ldots, m_{\mathrm{s},N}]^\top$

E.3 Approximate Message Passing (AMP)

The AMP algorithm was first introduced in [DMM09] and rigorously analyzed in [BM11]. Here, the "Bayesian version" [DMM10] is displayed, i.e., the case with non-linear MMSE estimator as so-called *denoiser*, i.e., for the signal-constrained estimation. The algorithm is stated in the notation used throughout the thesis. In AMP, the residuum with the so-called *Onsager correction term* plays an important role, since it removes the correlations between estimates of different iterations, which is in the other algorithms done by bias compensation or computing extrinsic. Consequently, the processing omits the unbiased estimates, i.e., when comparing to VAMP one may set $\tilde{\boldsymbol{x}}_\mathrm{c} = \boldsymbol{m}_\mathrm{s}$ and $\tilde{\boldsymbol{x}}_\mathrm{s} = \boldsymbol{m}_\mathrm{c}$, as well as $\tilde{\sigma}_\mathrm{c}^2 = \sigma_\mathrm{s}^2$ and $\tilde{\sigma}_\mathrm{s}^2 = \sigma_\mathrm{c}^2$ before the estimations.

Algorithm E.3: $\boldsymbol{m}_\mathrm{s} = \mathtt{AMP}(\boldsymbol{y},\, \boldsymbol{A},\, \sigma_\mathsf{n}^2)$

$\boldsymbol{m}_\mathrm{s} = \mathbf{0}$, $\boldsymbol{r} = \boldsymbol{y}$, $\sigma_\mathrm{s}^2 = s/N$ `// initialization`

while *stopping criterion not met* **do**

  $\boldsymbol{m}_\mathrm{c} = \boldsymbol{m}_\mathrm{s} + \boldsymbol{A}^\top \boldsymbol{r}$ `// MF`

  $\sigma_\mathrm{c}^2 = \sigma_\mathsf{n}^2 + \frac{N}{M}\sigma_\mathrm{s}^2$

  $\boldsymbol{m}_\mathrm{s} = \mathrm{E}\{\mathsf{x} \mid \boldsymbol{m}_\mathrm{c}, \sigma_\mathrm{c}^2\}$ `// NLMMSE`

  $\sigma_\mathrm{s}^2 = \frac{1}{N}\sum_{j=1}^{N} \mathrm{E}_\mathsf{x}\{(\mathsf{x} - m_{\mathrm{s},j})^2 \mid m_{\mathrm{c},j}, \sigma_\mathrm{c}^2\}$

  $\boldsymbol{r} = \boldsymbol{y} - \boldsymbol{A}\boldsymbol{m}_\mathrm{s} + \frac{N}{M}\frac{\sigma_\mathrm{s}^2}{\sigma_\mathrm{c}^2} \cdot \boldsymbol{r}$ `// residuum with Onsager term`

$\boldsymbol{m}_\mathrm{s} = [m_{\mathrm{s},1},\, \ldots,\, m_{\mathrm{s},N}]^\top$

E.4 Generalized Approximate Message Passing (GAMP)

When referring to GAMP, what is meant is the algorithm stated in [Ran11], but restricted to additive Gaussian noise in the measurements. Closer to the notation here, is the same algorithm stated in [KMS+12, Sec. 3.2] or [MKTZ15, Sec. 2.2], where it is called relaxed Belief Propagation.

Algorithm E.4: $\boldsymbol{m}_\mathrm{s} = \texttt{GAMP}(\boldsymbol{y},\, \boldsymbol{A},\, \sigma_\mathsf{n}^2)$

1 $\boldsymbol{m}_\mathrm{s} = \boldsymbol{0}$, $\sigma_{\boldsymbol{A},i}^2 = \frac{s}{N}\boldsymbol{a}_i^\top \boldsymbol{a}_i$ $i \in \{1,\, \ldots,\, M\}$, $\boldsymbol{r} = \mathbf{diag}(\frac{1}{\sigma_\mathsf{n}^2 + \sigma_{\boldsymbol{A},i}^2})\boldsymbol{y}$ `// initialization`

2 **while** *stopping criterion not met* **do**

3 $\quad \sigma_{\mathrm{c},j}^2 = \left(\sum_{i=1}^M \frac{a_{ij}^2}{\sigma_\mathsf{n}^2 + \sigma_{\boldsymbol{A},i}^2}\right)^{-1}$ $\quad j \in \{1,\, \ldots,\, N\}$

4 $\quad \boldsymbol{m}_\mathrm{c} = \boldsymbol{m}_\mathrm{s} + \mathbf{diag}(\sigma_{\mathrm{c},j}^2)\boldsymbol{A}^\top \boldsymbol{r}$ `// MF with bias compensation`

5 $\quad$ **for** $j \in \{1,\, \ldots,\, N\}$ **do**

6 $\quad\quad m_{\mathrm{s},j} = \mathrm{E}_\mathsf{x}\{\mathsf{x} \mid m_{\mathrm{c},j},\, \sigma_{\mathrm{c},j}^2\}$ `// NLMMSE`

7 $\quad\quad \sigma_{\mathrm{s},j}^2 = \mathrm{E}_\mathsf{x}\{(\mathsf{x} - m_{\mathrm{s},j})^2 \mid m_{\mathrm{c},j},\, \sigma_{\mathrm{c},j}^2\}$

8 $\quad \sigma_{\boldsymbol{A},i}^2 = \boldsymbol{a}_i^\top \mathbf{diag}(\sigma_{\mathrm{s},j}^2)\boldsymbol{a}_i$ $\quad i \in \{1,\, \ldots,\, M\}$

9 $\quad \boldsymbol{r} = \mathbf{diag}(\frac{1}{\sigma_\mathsf{n}^2 + \sigma_{\boldsymbol{A},i}^2})(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{m}_\mathrm{s} + \mathbf{diag}(\sigma_{\boldsymbol{A},i}^2)\boldsymbol{r})$ `// res. with Onsager term`

10 $\boldsymbol{m}_\mathrm{s} = [m_{\mathrm{s},1},\, \ldots,\, m_{\mathrm{s},N}]^\top$

Note that GAMP imitates the LMMSE inverse from (2.10) by scaling the residuum with $\mathbf{diag}(1/(\sigma_\mathsf{n}^2 + \sigma_{\boldsymbol{A},i}^2))$, which is the inverse of the diagonal of $\boldsymbol{M} = \boldsymbol{A}\boldsymbol{\Phi}_\mathsf{x}\boldsymbol{A}^\top + \sigma_\mathsf{n}^2\boldsymbol{I}_M$. The scaling with $\mathbf{diag}(\sigma_{\mathrm{c},j}^2)$ in the linear estimation (Line 4) performs the respective individual bias compensation, since the entries of the end-to-end cascade between $\boldsymbol{x}$ and $\boldsymbol{m}_\mathrm{s} + \boldsymbol{A}^\top\boldsymbol{r}$ are here given by

$$[\boldsymbol{A}^\top \mathbf{diag}\left(\frac{1}{\sigma_\mathsf{n}^2 + \sigma_{\boldsymbol{A},i}^2}\right)\boldsymbol{A}]_{jj} = \sum_{i=1}^M \frac{a_{ij}^2}{\sigma_\mathsf{n}^2 + \sigma_{\boldsymbol{A},i}^2} = \frac{1}{\sigma_{\mathrm{c},j}^2}\,, \quad j \in \{1,\, \ldots,\, N\}\,. \tag{E.2}$$

Note that the usual unbiased variance is given by (4.56)

$$\begin{aligned}\sigma_{\mathrm{c,u},j}^2 &= \sigma_{\mathsf{x},j}^2([\boldsymbol{W}]_{jj} - 1) = \sigma_{\mathsf{x},j}^2(\frac{1}{[\boldsymbol{K}]_{jj}} - 1)\\ &= \sigma_{\mathsf{x},j}^2\left(\frac{1}{[\boldsymbol{\Phi}_\mathsf{x}\boldsymbol{A}^\top\boldsymbol{M}^{-1}\boldsymbol{A}]_{jj}} - 1\right)\\ &= \frac{1}{[\boldsymbol{A}^\top\boldsymbol{M}^{-1}\boldsymbol{A}]_{jj}} - \sigma_{\mathsf{x},j}^2\,,\end{aligned} \tag{E.3}$$

i.e., the variance $\sigma_{\mathrm{c},j}^2$ in the algorithm is an approximation of an unbiased variance, which should work at least for small variances $\sigma_{\mathsf{x},j}^2$. Compared to VAMP, one may interpret the Onsager correction term in Line 9 as a compensation for the neglection of the off-diagonal elements in the LMMSE inverse and the missing of the bias compensation procedure after the signal-constrained estimation.

E.5 Vector Approximate Message Passing (VAMP)

The VAMP algorithm was first introduced in [RSF17], derived from the expectation-consistent (EC) approximate inference framework [OW05]. A different approach leading to the same algorithm is shown in [Spa19, SF17].

Algorithm E.5: $\boldsymbol{m}_\mathrm{s} = \texttt{VAMP}(\boldsymbol{y},\, \boldsymbol{A},\, \sigma_\mathsf{n}^2)$

$\tilde{\sigma}_\mathrm{c}^2 = s/N,\ \tilde{\boldsymbol{x}}_\mathrm{c} = \mathbf{0}$ // initialization
while *stopping criterion not met* **do**
  $\boldsymbol{m}_\mathrm{c} = \tilde{\boldsymbol{x}}_\mathrm{c} + (\boldsymbol{A}^\top\boldsymbol{A} + \frac{\sigma_\mathsf{n}^2}{\tilde{\sigma}_\mathrm{c}^2}\boldsymbol{I}_N)^{-1}\boldsymbol{A}^\top(\boldsymbol{y} - \boldsymbol{A}\tilde{\boldsymbol{x}}_\mathrm{c})$ // LMMSE
  $\sigma_\mathrm{c}^2 = \mathrm{trace}(\sigma_\mathsf{n}^2(\boldsymbol{A}^\top\boldsymbol{A} + \frac{\sigma_\mathsf{n}^2}{\tilde{\sigma}_\mathrm{c}^2}\boldsymbol{I}_N)^{-1})/N$
  $\tilde{\sigma}_\mathrm{s}^2 = (\frac{1}{\sigma_\mathrm{c}^2} - \frac{1}{\tilde{\sigma}_\mathrm{c}^2})^{-1}$ // unbiasing
  $\tilde{\boldsymbol{x}}_\mathrm{s} = \tilde{\sigma}_\mathrm{s}^2(\frac{\boldsymbol{m}_\mathrm{c}}{\sigma_\mathrm{c}^2} - \frac{\tilde{\boldsymbol{x}}_\mathrm{c}}{\tilde{\sigma}_\mathrm{c}^2})$
  $\boldsymbol{m}_\mathrm{s} = \mathrm{E}_\mathsf{x}\{\mathsf{x} \mid \tilde{\boldsymbol{x}}_\mathrm{s},\, \tilde{\sigma}_\mathrm{s}^2\}$ // NLMMSE
  $\sigma_\mathrm{s}^2 = \frac{1}{N}\sum_{j=1}^{N} \mathrm{E}_\mathsf{x}\{(\mathsf{x} - m_{\mathrm{s},j})^2 \mid \tilde{x}_{\mathrm{s},j},\, \tilde{\sigma}_\mathrm{s}^2\}$
  $\tilde{\sigma}_\mathrm{c}^2 = (\frac{1}{\sigma_\mathrm{s}^2} - \frac{1}{\tilde{\sigma}_\mathrm{s}^2})^{-1}$ // unbiasing
  $\tilde{\boldsymbol{x}}_\mathrm{c} = \tilde{\sigma}_\mathrm{c}^2(\frac{\boldsymbol{m}_\mathrm{s}}{\sigma_\mathrm{s}^2} - \frac{\tilde{\boldsymbol{x}}_\mathrm{s}}{\tilde{\sigma}_\mathrm{s}^2})$

The representation here, shall show the connection to extrinsic calculation as subtraction of natural parameters. For the implementation, it is better suited to directly compute the *unbiased* channel-constrained estimate via

$$w = \left(\frac{1}{N}\mathrm{trace}\left(\boldsymbol{A}^\top(\boldsymbol{A}\boldsymbol{A}^\top + \frac{\sigma_\mathsf{n}^2}{\tilde{\sigma}_\mathrm{c}^2}\boldsymbol{I}_N)^{-1}\boldsymbol{A}\right)\right)^{-1}, \tag{E.4}$$

$$\tilde{\boldsymbol{x}}_\mathrm{s} = \tilde{\boldsymbol{x}}_\mathrm{c} + w\boldsymbol{A}^\top(\boldsymbol{A}\boldsymbol{A}^\top + \frac{\sigma_\mathsf{n}^2}{\tilde{\sigma}_\mathrm{c}^2}\boldsymbol{I}_M)^{-1}(\boldsymbol{y} - \boldsymbol{A}\tilde{\boldsymbol{x}}_\mathrm{c}), \tag{E.5}$$

$$\tilde{\sigma}_\mathrm{s}^2 = \tilde{\sigma}_\mathrm{c}^2(w - 1), \tag{E.6}$$

because it involves only the inversion of an $M \times M$ matrix, instead of an $N \times N$ matrix, which is beneficial, since $M < N$. Furthermore, one can simplify the algorithm by utilizing the SVD of sensing matrix $\boldsymbol{A}$ for the computation of the inverse. Let the SVD be given by $\boldsymbol{A} = \boldsymbol{U}\mathbf{diag}(s_i)\boldsymbol{V}^\top$, where $\boldsymbol{U} \in \mathbb{R}^{M\times M}$ is an orthonormal matrix (satisfying $\boldsymbol{U}^{-1} = \boldsymbol{U}^\top$), $s_1 \geq s_2 \geq \cdots \geq s_M > 0$, and $\boldsymbol{V} \in \mathbb{R}^{N\times M}$ is a matrix with orthonormal column vectors, which means $\boldsymbol{V}^\top\boldsymbol{V} = \boldsymbol{I}_M$. Then, the estimate $\tilde{\boldsymbol{x}}_\mathrm{s}$ can be written as

$$\tilde{\boldsymbol{x}}_\mathrm{s} = \tilde{\boldsymbol{x}}_\mathrm{c} + w\boldsymbol{V}\mathbf{diag}(s_i)\left(\mathbf{diag}\left(s_i^2\right) + \frac{\sigma_\mathsf{n}^2}{\tilde{\sigma}_\mathrm{c}^2}\boldsymbol{I}_M\right)^{-1}(\boldsymbol{U}^\top\boldsymbol{y} - \mathbf{diag}(s_i)\boldsymbol{V}^\top\tilde{\boldsymbol{x}}_\mathrm{c}), \tag{E.7}$$

where w is computed as

$$w = \left(\frac{1}{N}\mathrm{trace}\left(\boldsymbol{V}\mathbf{diag}(s_i)\left(\mathbf{diag}\left(s_i^2\right) + \frac{\sigma_\mathsf{n}^2}{\tilde{\sigma}_\mathrm{c}^2}\boldsymbol{I}_M\right)^{-1}\mathbf{diag}(s_i)\boldsymbol{V}^\top\right)\right)^{-1}. \tag{E.8}$$

The important simplification lies in the fact, that the matrix inverse is replaced by M scalar inverses, since the matrix to be inverted is diagonal. For details see [RSF19, Spa19].

E.6 VAMP with Individual Variances (VAMPind)

This algorithm results from VAMP, when simply replacing the averaged variances by individual ones without changing anything else in the processing. It was first proposed in [FSRS16]. For comparability, the same notation as in the VAMP algorithm above is used. Mathematically the same but better for the implementation of the LMMSE is the processing used in Algorithm E.7 below.

Algorithm E.6: $\boldsymbol{m}_{\mathrm{s}} = \texttt{VAMPind}(\boldsymbol{y},\, \boldsymbol{A},\, \sigma_{\mathsf{n}}^2)$

$\tilde{\boldsymbol{\Phi}}_{\mathrm{c}} = \frac{s}{N}\boldsymbol{I}_N,\ \tilde{\boldsymbol{x}}_{\mathrm{c}} = \mathbf{0}$ // initialization

while *stopping criterion not met* **do**

$\boldsymbol{m}_{\mathrm{c}} = \tilde{\boldsymbol{x}}_{\mathrm{c}} + (\boldsymbol{A}^{\top}\boldsymbol{A} + \sigma_{\mathsf{n}}^2\tilde{\boldsymbol{\Phi}}_{\mathrm{c}}^{-1})^{-1}\boldsymbol{A}^{\top}(\boldsymbol{y} - \boldsymbol{A}\tilde{\boldsymbol{x}}_{\mathrm{c}})$ // LMMSE

$\boldsymbol{\Phi}_{\mathrm{c}} = \sigma_{\mathsf{n}}^2(\boldsymbol{A}^{\top}\boldsymbol{A} + \sigma_{\mathsf{n}}^2\tilde{\boldsymbol{\Phi}}_{\mathrm{c}}^{-1})^{-1}$

for $j \in \{1, \ldots, N\}$ **do**

$\sigma_{\mathrm{c},j}^2 = [\boldsymbol{\Phi}_{\mathrm{c}}]_{jj}$

$\tilde{\sigma}_{\mathrm{s},j}^2 = (\frac{1}{\sigma_{\mathrm{c},j}^2} - \frac{1}{\tilde{\sigma}_{\mathrm{c},j}^2})^{-1}$ // unbiasing

$\tilde{x}_{\mathrm{s},j} = \tilde{\sigma}_{\mathrm{s},j}^2(\frac{m_{\mathrm{c},j}}{\sigma_{\mathrm{c},j}^2} - \frac{\tilde{x}_{\mathrm{c},j}}{\tilde{\sigma}_{\mathrm{c},j}^2})$

$m_{\mathrm{s},j} = \mathrm{E}_{\mathsf{x}}\{\mathsf{x} \mid \tilde{x}_{\mathrm{s},j},\, \tilde{\sigma}_{\mathrm{s},j}^2\}$ // NLMMSE

$\sigma_{\mathrm{s},j}^2 = \mathrm{E}_{\mathsf{x}}\{(\mathsf{x} - m_{\mathrm{s},j})^2 \mid \tilde{x}_{\mathrm{s},j},\, \tilde{\sigma}_{\mathrm{s},j}^2\}$

$\tilde{\sigma}_{\mathrm{c},j}^2 = (\frac{1}{\sigma_{\mathrm{s},j}^2} - \frac{1}{\tilde{\sigma}_{\mathrm{s},j}^2})^{-1},\ [\tilde{\boldsymbol{\Phi}}_{\mathrm{c}}]_{jj} = \tilde{\sigma}_{\mathrm{c},j}^2$ // unbiasing

$\tilde{x}_{\mathrm{c},j} = \tilde{\sigma}_{\mathrm{c},j}^2(\frac{m_{\mathrm{s},j}}{\sigma_{\mathrm{s},j}^2} - \frac{\tilde{x}_{\mathrm{s},j}}{\tilde{\sigma}_{\mathrm{s},j}^2})$

$\boldsymbol{m}_{\mathrm{s}} = [m_{\mathrm{s},1},\, \ldots,\, m_{\mathrm{s},N}]^{\top}$

E.7 VAMPire

The Algorithm E.6 shows significant drawbacks in the processing of the variances, since it allows for negative variances. By an estimation-theoretic point of view, this treatment can be corrected and leads to, what we call VAMPire; for the derivation see Sec. 4.4.2.1. The algorithm was first published in [FSG20].

Algorithm E.7: $\boldsymbol{m}_\mathrm{s} = \texttt{VAMPire}(\boldsymbol{y},\, \boldsymbol{A},\, \sigma_\mathsf{n}^2)$

$\tilde{\boldsymbol{\Phi}}_\mathrm{c} = \frac{s}{N}\boldsymbol{I}_N,\ \tilde{\boldsymbol{x}}_\mathrm{c} = \boldsymbol{0}$ // initialization

while *stopping criterion not met* **do**

  $\boldsymbol{K} = \tilde{\boldsymbol{\Phi}}_\mathrm{c}\boldsymbol{A}^\top(\boldsymbol{A}\tilde{\boldsymbol{\Phi}}_\mathrm{c}\boldsymbol{A}^\top + \sigma_\mathsf{n}^2\boldsymbol{I}_N)^{-1}\boldsymbol{A}$

  $[\boldsymbol{W}]_{jj} = 1/[\boldsymbol{K}]_{jj} \qquad j \in \{1,\, \ldots,\, N\}$

  $\tilde{\boldsymbol{x}}_\mathrm{s} = \tilde{\boldsymbol{x}}_\mathrm{c} + \boldsymbol{W}\tilde{\boldsymbol{\Phi}}_\mathrm{c}\boldsymbol{A}^\top(\boldsymbol{A}\tilde{\boldsymbol{\Phi}}_\mathrm{c}\boldsymbol{A}^\top + \sigma_\mathsf{n}^2\boldsymbol{I}_N)^{-1}(\boldsymbol{y} - \boldsymbol{A}\tilde{\boldsymbol{x}}_\mathrm{c})$ // unbiased LMMSE

  for $j \in \{1,\, \ldots,\, N\}$ **do**

    $\tilde{\sigma}_{\mathrm{s},j}^2 = \tilde{\sigma}_{\mathrm{c},j}^2([\boldsymbol{W}]_{jj} - 1)$

    $m_{\mathrm{s},j} = \mathrm{E}_\mathsf{x}\{\mathsf{x} \mid \tilde{x}_{\mathrm{s},j},\, \tilde{\sigma}_{\mathrm{s},j}^2\}$ // NLMMSE

    $\sigma_{\mathrm{s},j}^2 = \mathrm{E}_\mathsf{x}\{(\mathsf{x} - m_{\mathrm{s},j})^2 \mid \tilde{x}_{\mathrm{s},j},\, \tilde{\sigma}_{\mathrm{s},j}^2\}$

    $\sigma_j^2 = \mathrm{E}_{\tilde{\mathsf{x}}_\mathrm{s}}\{\sigma_{\mathrm{s},j}^2\}$ // $\tilde{\mathsf{x}}_\mathrm{s} = \mathsf{x} + \mathsf{n}_\mathrm{s},\ \mathsf{n}_\mathrm{s} \sim \mathcal{N}(0, \tilde{\sigma}_{\mathrm{s},j}^2)$

    $\tilde{\sigma}_{\mathrm{c},j}^2 = \sigma_{\mathrm{s},j}^2 + \left(\frac{\sigma_j^2}{\tilde{\sigma}_{\mathrm{s},j}^2 - \sigma_j^2}(m_{\mathrm{s},j} - \tilde{x}_{\mathrm{s},j})\right)^2$ // unbiasing

    $\tilde{x}_{\mathrm{c},j} = (1/\sigma_j^2 - 1/\tilde{\sigma}_{\mathrm{s},j}^2)^{-1}(\frac{m_{\mathrm{s},j}}{\sigma_j^2} - \frac{\tilde{x}_{\mathrm{s},j}}{\tilde{\sigma}_{\mathrm{s},j}^2})$

  $\tilde{\boldsymbol{\Phi}}_\mathrm{c} = \mathbf{diag}(\tilde{\sigma}_{\mathrm{c},j}^2),\, \tilde{\boldsymbol{x}}_\mathrm{c} = [\tilde{x}_{\mathrm{c},1},\, \ldots,\, \tilde{x}_{\mathrm{c},N}]$

$\boldsymbol{m}_\mathrm{s} = [m_{\mathrm{s},1},\, \ldots,\, m_{\mathrm{s},N}]^\top$

E.8 VAMP with Individually Unbiased LMMSE Estimator

The following algorithms result from VAMP by replacing the average bias compensation that is performed after the channel-constrained estimation with individual unbiasing. They differ in the treatment of the variances before the signal-constrained estimation. VAMPia averages the variances before the signal-constrained estimation; VAMPii uses the individual variances directly. The algorithms were first published in [SF22].

Algorithm E.8: $\boldsymbol{m}_\mathrm{s} = \mathtt{VAMPia}(\boldsymbol{y},\, \boldsymbol{A},\, \sigma_\mathsf{n}^2)$

$\tilde{\sigma}_\mathrm{c}^2 = s/N,\ \tilde{\boldsymbol{x}}_\mathrm{c} = \mathbf{0}$ // initialization

while *stopping criterion not met* **do**

$\quad \boldsymbol{K} = \boldsymbol{A}^\top(\boldsymbol{A}\boldsymbol{A}^\top + \frac{\sigma_\mathsf{n}^2}{\tilde{\sigma}_\mathrm{c}^2}\boldsymbol{I}_M)^{-1}\boldsymbol{A}$

$\quad \boldsymbol{W} = \mathbf{diag}(1/[\boldsymbol{K}]_{jj})$

$\quad \tilde{\boldsymbol{x}}_\mathrm{s} = \tilde{\boldsymbol{x}}_\mathrm{c} + \boldsymbol{W}\boldsymbol{A}^\top(\boldsymbol{A}\boldsymbol{A}^\top + \frac{\sigma_\mathsf{n}^2}{\tilde{\sigma}_\mathrm{c}^2}\boldsymbol{I}_M)^{-1}(\boldsymbol{y} - \boldsymbol{A}\tilde{\boldsymbol{x}}_\mathrm{c})$ // unbiased LMMSE

$\quad \tilde{\sigma}_\mathrm{s}^2 = \tilde{\sigma}_\mathrm{c}^2(\frac{1}{N}\sum_{j=1}^{N}\frac{1}{[\boldsymbol{K}]_{jj}} - 1)$

$\quad \boldsymbol{m}_\mathrm{s} = \mathrm{E}_\mathsf{x}\{\mathsf{x} \mid \tilde{\boldsymbol{x}}_\mathrm{s},\, \tilde{\sigma}_\mathrm{s}^2\}$ // NLMMSE

$\quad \sigma_\mathrm{s}^2 = \frac{1}{N}\sum_{j=1}^{N}\mathrm{E}_\mathsf{x}\{(\mathsf{x} - m_{\mathrm{s},j})^2 \mid \tilde{x}_{\mathrm{s},j},\, \tilde{\sigma}_\mathrm{s}^2\}$

$\quad \tilde{\sigma}_\mathrm{c}^2 = (\frac{1}{\sigma_\mathrm{s}^2} - \frac{1}{\tilde{\sigma}_\mathrm{s}^2})^{-1}$ // unbiasing

$\quad \tilde{\boldsymbol{x}}_\mathrm{c} = \tilde{\sigma}_\mathrm{c}^2(\frac{\boldsymbol{m}_\mathrm{s}}{\sigma_\mathrm{s}^2} - \frac{\tilde{\boldsymbol{x}}_\mathrm{s}}{\tilde{\sigma}_\mathrm{s}^2})$

Algorithm E.9: $\boldsymbol{m}_\mathrm{s} = \mathtt{VAMPii}(\boldsymbol{y},\, \boldsymbol{A},\, \sigma_\mathsf{n}^2)$

$\tilde{\sigma}_\mathrm{c}^2 = s/N,\ \tilde{\boldsymbol{x}}_\mathrm{c} = \mathbf{0}$ // initialization

while *stopping criterion not met* **do**

$\quad \boldsymbol{K} = \boldsymbol{A}^\top(\boldsymbol{A}\boldsymbol{A}^\top + \frac{\sigma_\mathsf{n}^2}{\tilde{\sigma}_\mathrm{c}^2}\boldsymbol{I}_M)^{-1}\boldsymbol{A}$

$\quad \boldsymbol{W} = \mathbf{diag}(1/[\boldsymbol{K}]_{jj})$

$\quad \tilde{\boldsymbol{x}}_\mathrm{s} = \tilde{\boldsymbol{x}}_\mathrm{c} + \boldsymbol{W}\boldsymbol{A}^\top(\boldsymbol{A}\boldsymbol{A}^\top + \frac{\sigma_\mathsf{n}^2}{\tilde{\sigma}_\mathrm{c}^2}\boldsymbol{I}_M)^{-1}(\boldsymbol{y} - \boldsymbol{A}\tilde{\boldsymbol{x}}_\mathrm{c})$ // unbiased LMMSE

$\quad \tilde{\sigma}_{\mathrm{s},j}^2 = \tilde{\sigma}_\mathrm{c}^2(1/[\boldsymbol{K}]_{jj} - 1) \qquad j \in \{1, \ldots, N\}$

$\quad m_{\mathrm{s},j} = \mathrm{E}_\mathsf{x}\{\mathsf{x} \mid \tilde{x}_{\mathrm{s},j},\, \tilde{\sigma}_{\mathrm{s},j}^2\} \qquad j \in \{1, \ldots, N\}$ // NLMMSE

$\quad \sigma_\mathrm{s}^2 = \frac{1}{N}\sum_{j=1}^{N}\mathrm{E}_\mathsf{x}\{(\mathsf{x} - m_{\mathrm{s},j})^2 \mid \tilde{x}_{\mathrm{s},j},\, \tilde{\sigma}_{\mathrm{s},j}^2\}$

$\quad \tilde{\sigma}_\mathrm{s}^2 = \frac{1}{N}\sum_{j=1}^{N}\tilde{\sigma}_{\mathrm{s},j}^2$

$\quad \tilde{\sigma}_\mathrm{c}^2 = (\frac{1}{\sigma_\mathrm{s}^2} - \frac{1}{\tilde{\sigma}_\mathrm{s}^2})^{-1}$ // unbiasing

$\quad \tilde{\boldsymbol{x}}_\mathrm{c} = \tilde{\sigma}_\mathrm{c}^2(\frac{\boldsymbol{m}_\mathrm{s}}{\sigma_\mathrm{s}^2} - \frac{\tilde{\boldsymbol{x}}_\mathrm{s}}{\tilde{\sigma}_\mathrm{s}^2})$

$\boldsymbol{m}_\mathrm{s} = [m_{\mathrm{s},1},\, \ldots,\, m_{\mathrm{s},N}]^\top$

E.9 VAMPind with Sequentially Processed Variables

The usage of individual variances enables the possibility for a sequential processing of the signal components to be recovered. The algorithm seqVAMPind is therefore a sequential version (in terms of processing the variables) of VAMPind. The idea for the algorithm was given by the algorithm stated in [OW05, Appendix D.], which differs from the algorithm here by the utilization of a binary prior. In [BMPP20, Algorithm 1], the algorithm is stated for Bernoulli-Gaussian prior.

Algorithm E.10: $\boldsymbol{m}_{\mathrm{s}} = \texttt{seqVAMPind}(\boldsymbol{y},\, \boldsymbol{A},\, \sigma_{\mathsf{n}}^2)$

$\boldsymbol{\Lambda}_{\mathrm{c}} = [\frac{N}{s},\, \ldots,\, \frac{N}{s}]_N^\top,\ \boldsymbol{\lambda}_{\mathrm{c}} = \mathbf{0}$ // initialization

$\boldsymbol{\Phi}_{\mathrm{c}} = (\frac{1}{\sigma_{\mathsf{n}}^2}\boldsymbol{A}^\top\boldsymbol{A} + \boldsymbol{\Lambda}_{\mathrm{c}})^{-1} = [\boldsymbol{\phi}_{\mathrm{c},1},\, \ldots,\, \boldsymbol{\phi}_{\mathrm{c},N}],\ \sigma_{\mathrm{c},j}^2 = [\boldsymbol{\Phi}_{\mathrm{c}}]_{jj}$ // LMMSE

$\boldsymbol{m}_{\mathrm{c}} = \boldsymbol{\Phi}_{\mathrm{c}}(\frac{1}{\sigma_{\mathsf{n}}^2}\boldsymbol{A}^\top\boldsymbol{y} + \boldsymbol{\lambda}_{\mathrm{c}})$

while *stopping criterion not met* **do**

  Choose index set $\mathcal{J} \subseteq \{1,\, \ldots,\, N\}$ with $|\mathcal{J}| = T$

  Specify sequence $s_{\mathcal{J}} = [j_1,\, j_2,\, \ldots,\, j_T] \,:\, j_i \in \mathcal{J}\ \forall i \in \{1,\, \ldots,\, T\}$

  for $j = s_{\mathcal{J}}$ **do**

    $\tilde{\sigma}_{\mathrm{s},j}^2 = (1/\sigma_{\mathrm{c},j}^2 - \Lambda_{\mathrm{c},j})^{-1}$ // unbiasing

    $\tilde{x}_{\mathrm{s},j} = \tilde{\sigma}_{\mathrm{s},j}^2(m_{\mathrm{c},j}/\sigma_{\mathrm{c},j}^2 - \lambda_{\mathrm{c},j})$

    $m_{\mathrm{s},j} = \mathrm{E}_{\mathsf{x}}\{\mathsf{x} \mid \tilde{x}_{\mathrm{s},j},\, \tilde{\sigma}_{\mathrm{s},j}^2\}$, // NLMMSE

    $\sigma_{\mathrm{s},j}^2 = \mathrm{E}_{\mathsf{x}}\{(\mathsf{x} - m_{\mathrm{s},j})^2 \mid \tilde{x}_{\mathrm{s},j},\, \tilde{\sigma}_{\mathrm{s},j}^2\}$

    $\lambda_{\mathrm{c},j} = m_{\mathrm{s},j}/\sigma_{\mathrm{s},j}^2 - \tilde{x}_{\mathrm{s},j}/\tilde{\sigma}_{\mathrm{s},j}^2$ // unbiasing

    $\Lambda_{\mathrm{c},j} = 1/\sigma_{\mathrm{s},j}^2 - 1/\tilde{\sigma}_{\mathrm{s},j}^2$

    $\boldsymbol{m}_{\mathrm{c}} = \boldsymbol{m}_{\mathrm{c}} + \frac{1}{\sigma_{\mathrm{c},j}^2}\,(m_{\mathrm{s},j} - m_{\mathrm{c},j})\,\boldsymbol{\phi}_{\mathrm{c},j}$ // rank-one update

    $\boldsymbol{\Phi}_{\mathrm{c}} = \boldsymbol{\Phi}_{\mathrm{c}} + \frac{1}{(\sigma_{\mathrm{c},j}^2)^2}\left(\sigma_{\mathrm{s},j}^2 - \sigma_{\mathrm{c},j}^2\right)\boldsymbol{\phi}_{\mathrm{c},j}\boldsymbol{\phi}_{\mathrm{c},j}^\top$

    for $j' = 1,\, \ldots,\, N$ **do**

      $\sigma_{\mathrm{c},j'}^2 = [\boldsymbol{\Phi}_{\mathrm{c}}]_{j'j'}$

$\boldsymbol{m}_{\mathrm{s}} = [m_{\mathrm{s},1},\, \ldots,\, m_{\mathrm{s},N}]^\top$

E.10 VAMPire with Sequentially Processed Variables

The algorithm shown here improves upon seqVAMPind by employing estimation-theoretic bias compensation after the signal-constrained estimation, i.e., it is the sequential version of VAMPire. It was introduced by the author in [SF21].

Algorithm E.11: $\boldsymbol{m}_\mathrm{s} = \texttt{seqVAMPire}(\boldsymbol{y},\, \boldsymbol{A},\, \sigma_\mathsf{n}^2)$

$\boldsymbol{\Lambda}_\mathrm{c} = [\frac{N}{s},\, \ldots,\, \frac{N}{s}]_N^\top,\ \boldsymbol{\lambda}_\mathrm{c} = \boldsymbol{0},$ `// initialization`

$\boldsymbol{\Phi}_\mathrm{c} = (\frac{1}{\sigma_\mathsf{n}^2}\boldsymbol{A}^\top\boldsymbol{A} + \boldsymbol{\Lambda}_\mathrm{c})^{-1} = [\boldsymbol{\phi}_{\mathrm{c},1},\, \ldots,\, \boldsymbol{\phi}_{\mathrm{c},N}],\ \sigma_{\mathrm{c},j}^2 = [\boldsymbol{\Phi}_\mathrm{c}]_{jj}\ j \in \{1,\, \ldots,\, N\}$ `// LMMSE`

$\boldsymbol{m}_\mathrm{c} = \boldsymbol{\Phi}_\mathrm{c}(\frac{1}{\sigma_\mathsf{n}^2}\boldsymbol{A}^\top\boldsymbol{y} + \boldsymbol{\lambda}_\mathrm{c})$

while *stopping criterion not met* **do**

  Choose index set $\mathcal{J} \subseteq \{1,\, \ldots,\, N\}$ with $|\mathcal{J}| = T$

  Specify sequence $s_\mathcal{J} = [j_1,\, j_2,\, \ldots,\, j_T] : j_i \in \mathcal{J}\ \forall i \in \{1,\, \ldots,\, T\}$

  for $j = s_\mathcal{J}$ **do**

    $\tilde{\sigma}_{\mathrm{s},j}^2 = (1/\sigma_{\mathrm{c},j}^2 - \Lambda_{\mathrm{c},j})^{-1}$ `// unbiasing`

    $\tilde{x}_{\mathrm{s},j} = \tilde{\sigma}_{\mathrm{s},j}^2(m_{\mathrm{c},j}/\sigma_{\mathrm{c},j}^2 - \lambda_{\mathrm{c},j})$

    $\tilde{x}_{\mathrm{s},j} = \lambda_{\mathrm{s},j}/\Lambda_{\mathrm{s},j},\, \tilde{\sigma}_{\mathrm{s},j}^2 = 1/\Lambda_{\mathrm{s},j}$

    $m_{\mathrm{s},j} = \mathrm{E}_\mathsf{x}\{\mathsf{x} \mid \tilde{x}_{\mathrm{s},j},\, \tilde{\sigma}_{\mathrm{s},j}^2\},$ `// NLMMSE`

    $\sigma_{\mathrm{s},j}^2 = \mathrm{E}_\mathsf{x}\{(\mathsf{x} - m_{\mathrm{s},j})^2 \mid \tilde{x}_{\mathrm{s},j},\, \tilde{\sigma}_{\mathrm{s},j}^2\}$

    $\sigma_j^2 = \mathrm{E}_{\tilde{\mathsf{x}}_\mathrm{s}}\{\sigma_{\mathrm{s},j}^2\}$ `//` $\tilde{\mathsf{x}}_\mathrm{s} = \mathsf{x} + \mathsf{n}_\mathrm{s},\, \mathsf{n}_\mathrm{s} \sim \mathcal{N}(0, \tilde{\sigma}_{\mathrm{s},j}^2)$

    $\tilde{\sigma}_{\mathrm{c},j}^2 = \sigma_{\mathrm{s},j}^2 + (\frac{\sigma_j^2}{\tilde{\sigma}_{\mathrm{s},j}^2 - \sigma_j^2}(m_{\mathrm{s},j} - \tilde{x}_{\mathrm{s},j}))^2$

    $\tilde{\sigma}_j^2 = (1/\sigma_j^2 - 1/\tilde{\sigma}_{\mathrm{s},j}^2)^{-1}$

    $\Delta\Lambda_{\mathrm{c},j} = 1/\tilde{\sigma}_{\mathrm{c},j}^2 - \Lambda_{\mathrm{c},j}$

    $\Lambda_{\mathrm{c},j} = 1/\tilde{\sigma}_{\mathrm{c},j}^2$

    $\Delta\lambda_{\mathrm{c},j} = \tilde{\sigma}_j^2/\tilde{\sigma}_{\mathrm{c},j}^2 \cdot (m_{\mathrm{s},j}/\sigma_j^2 - \lambda_{\mathrm{s},j}) - \lambda_{\mathrm{c},j}$

    $\lambda_{\mathrm{c},j} = \tilde{\sigma}_j^2/\tilde{\sigma}_{\mathrm{c},j}^2 \cdot (m_{\mathrm{s},j}/\sigma_j^2 - \lambda_{\mathrm{s},j})$

    $\boldsymbol{m}_\mathrm{c} = \boldsymbol{m}_\mathrm{c} + \frac{\Delta\lambda_{\mathrm{c},j} - \Delta\Lambda_{\mathrm{c},j}m_{\mathrm{c},j}}{1+\Delta\Lambda_{\mathrm{c},j}\sigma_{\mathrm{c},j}^2}\boldsymbol{\phi}_{\mathrm{c},j}$ `// rank-one update`

    $\boldsymbol{\Phi}_\mathrm{c} = \boldsymbol{\Phi}_\mathrm{c} - \frac{\Delta\Lambda_{\mathrm{c},j}}{1+\Delta\Lambda_{\mathrm{c},j}\sigma_{\mathrm{c},j}^2}\boldsymbol{\phi}_{\mathrm{c},j}\boldsymbol{\phi}_{\mathrm{c},j}^\top$

    for $j' = 1,\, \ldots,\, N$ **do**

      $\sigma_{\mathrm{c},j'}^2 = [\boldsymbol{\Phi}_\mathrm{c}]_{j'j'}$

$\boldsymbol{m}_\mathrm{s} = [m_{\mathrm{s},1},\, \ldots,\, m_{\mathrm{s},N}]^\top$

E.11 Sequential Processing of Observations

Here, the algorithm based on VAMP, that processes the observations in a sequential way, is stated. As in VAMP, an average variance is used to represent the reliability of the entire vector. Note that $\boldsymbol{a}_i$ is the ith row of sensing matrix $\boldsymbol{A}$.

The initialization is stated as in the implementation used for the simulations.

Algorithm E.12: $\boldsymbol{m} = \texttt{seqVAMPobs}(\boldsymbol{y},\, \boldsymbol{A},\, \sigma_{\mathsf{n}}^2)$

for $i = \mathrm{s},\, 1,\, \ldots,\, M$ **do**
  $\tilde{\boldsymbol{x}}^{(i)} = \boldsymbol{0},\, \sigma_{(i)}^2 = 10^8$ `// initialization`
$\boldsymbol{m} = \boldsymbol{0}, \sigma^2 = 5 \cdot 10^7$
while *stopping criterion not met* **do**
  Choose $i \in \{\mathrm{s},\, 1,\, \ldots,\, M\}$
  $\sigma_{\backslash i}^2 = (1/\sigma^2 - 1/\sigma_{(i)}^2)^{-1},\, \tilde{\boldsymbol{x}}^{\backslash i} = \sigma_{\backslash i}^2 (\boldsymbol{m}/\sigma^2 - \tilde{\boldsymbol{x}}^{(i)}/\sigma_{(i)}^2)$
  if $i \in \{1,\, \ldots,\, M\}$ **then**
    $w = \frac{1}{N} \mathrm{trace}(\boldsymbol{a}_i \left(\boldsymbol{a}_i^{\mathsf{T}} \boldsymbol{a}_i + \frac{\sigma_{\mathsf{n},i}^2}{\sigma_{\backslash i}^2} \right)^{-1} \boldsymbol{a}_i^{\mathsf{T}})$
    $\tilde{\boldsymbol{x}}^{(i)} = \tilde{\boldsymbol{x}}^{\backslash i} + \frac{1}{w} \boldsymbol{a}_i \left(\boldsymbol{a}_i^{\mathsf{T}} \boldsymbol{a}_i + \frac{\sigma_{\mathsf{n},i}^2}{\sigma_{\backslash i}^2} \right)^{-1} (y_i - \boldsymbol{a}_i^{\mathsf{T}} \tilde{\boldsymbol{x}}^{\backslash i})$ `// unbiased LMMSE`
    $\sigma_{(i)}^2 = \sigma_{\backslash i}^2 \left(\frac{1}{w} - 1 \right)$
    $\sigma^2 = \left(1/\sigma_{(i)}^2 + 1/\sigma_{\backslash i}^2 \right)^{-1}$
    $\boldsymbol{m} = \sigma^2 \left(\tilde{\boldsymbol{x}}^{(i)}/\sigma_{(i)}^2 + \tilde{\boldsymbol{x}}^{\backslash i}/\sigma_{\backslash i}^2 \right)$
  else
    $\boldsymbol{m} = \mathrm{E}_{\mathsf{x}}\{\mathsf{x} \mid \tilde{\boldsymbol{x}}^{\backslash \mathrm{s}},\, \sigma_{\backslash \mathrm{s}}^2\}$ `// NLMMSE`
    $\sigma^2 = \frac{1}{N} \sum_{j=1}^{N} \mathrm{E}_{\mathsf{x}}\{(\mathsf{x} - m_j)^2 \mid \tilde{x}_j^{\backslash \mathrm{s}},\, \sigma_{\backslash \mathrm{s}}^2\}$
    $\sigma_{(\mathrm{s})}^2 = (1/\sigma^2 - 1/\sigma_{\backslash \mathrm{s}}^2)^{-1},\, \tilde{\boldsymbol{x}}^{(\mathrm{s})} = \sigma_{(\mathrm{s})}^2 (\boldsymbol{m}/\sigma^2 - \tilde{\boldsymbol{x}}^{\backslash \mathrm{s}}/\sigma_{\backslash \mathrm{s}}^2)$

F. Notation

F.1 Abbreviations

Acronym	Meaning
a.k.a.	also known as
AMP	approximate message passing
avg.	average
GAMP	generalized AMP
BCJR	Bahl-Cocke-Jelinek-Raviv (algorithm)
BG	Bernoulli-Gauss (prior)
BP	belief propagation
CDMA	code-division multiple access
cf.	conferatur (compare)
CoSaMP	compressive sampling matching pursuit
CS	compressed sensing, also compressive sampling
dB	decibel
DT	discrete ternary (prior)
EC	expectation-consistent (approximate inference framework)
e.g.	exempli gratia (for example)
EP	expectation propagation
FLOP	floating-point operation
GBP	Gaussian belief propagation
GMP	Gaussian message passing

Acronym	Meaning
GMPpostExt	GMP with post-processed extrinsic
GMPpostExtLin	GMPpostExt at linear estimation
i.e.	id est (it is)
i.i.d.	independent and identically distributed
ind.	individual
IHT	iterative hard thresholding
ISF	iterative soft feedback
IST	iterative soft thresholding
LMMSE	linear minimum mean-squared error
LDPC	low-density parity-check
LLR	log-likelihood ratio
LUT	look-up table
MAP	maximum a-posteriori
MF	matched filter
MMSE	minimum mean-squared error
MP	message passing
OAMP	orthogonal approximate message passing
OFDM	orthogonal frequency-division multiplexing
OMP	orthogonal matching pursuit
PAR	peak-to-average power ratio
pdf	probablity density function
SER	symbol error rate
seqVAMPind	sequential VAMPind (sequential variable processing)
seqVAMPire	sequential VAMPire (sequential variable processing)
seqVAMPobs	sequential VAMP observation processing
SNR	signal-to-noise ratio
SVD	singular value decomposition
VAMP	vector approximate message passing
VAMPia	VAMP with individual unbiased LMMSE and average variance
VAMPii	VAMP with individual unbiased LMMSE and individual variance
VAMPind	VAMP with individual variances
VAMPire	VAMP with individual reliabilities enhanced

Acronym	Meaning
w.r.t.	with respect to
WSN	wireless sensor network

F.2 Mathematical Symbols

Throughout the thesis, scalars are denoted by lower-case letters, e.g., x, vectors by bold ones, e.g., $\boldsymbol{x}$, matrices by upper-case bold, e.g., $\boldsymbol{X}$, random variables in sans-serif font, e.g., x, random vectors in sans-serif bold, e.g., $\mathbf{x}$, and random matrices in upper-case bold font, e.g., $\mathbf{X}$. Vectors are always defined as column vectors.

F.2.1 System Parameters

Symbol	Meaning
i	row index of sensing matrix $\boldsymbol{A}$
j	column index of sensing matrix $\boldsymbol{A}$
M	observation dimension
N	signal dimension
s	sparsity, number of non-zero entries
σ_{n}^2	noise variance

F.2.2 System Model Variables

Symbol	Meaning
$\boldsymbol{A}$	sensing matrix
$\boldsymbol{a}_i^\top$	ith row of sensing matrix $\boldsymbol{A}$
$\tilde{\boldsymbol{a}}_j$	jth column of sensing matrix $\boldsymbol{A}$
a_{ij}	entry in sensing matrix $\boldsymbol{A}$ in ith row and jth column
$\boldsymbol{n}$	noise vector
n_i	ith element of noise vector $\boldsymbol{n}$
$\boldsymbol{x}$	signal vector
x_j	jth element of signal vector $\boldsymbol{x}$
$\boldsymbol{y}$	observation vector
y_i	ith element of observation vector $\boldsymbol{y}$

F.2.3 Estimation Variables

Symbol	Meaning
$\boldsymbol{m}_\mathrm{c}$	channel-constrained estimate / linear MMSE estimate
$m_{\mathrm{c},j}$	jth element of linear MMSE estimate $\boldsymbol{m}_\mathrm{c}$
$\boldsymbol{m}_\mathrm{c}^\circ$	updated linear MMSE estimate estimate
$m_{\mathrm{c},j}^\circ$	jth element of updated linear MMSE estimate $\boldsymbol{m}_\mathrm{c}^\circ$
$\boldsymbol{m}_{\mathrm{c},i}$	linear MMSE estimate based on ith observation y_i
$m_{\mathrm{c},i,j}$	jth element of linear MMSE estimate $\boldsymbol{m}_{\mathrm{c},i}$
$\boldsymbol{m}_\mathrm{s}$	signal-constrained estimate / non-linear MMSE estimate
$m_{\mathrm{s},j}$	jth element of $\boldsymbol{m}_\mathrm{s}$
$\boldsymbol{\Phi}_\mathrm{c}$	(conditional) covariance corresponding to $\boldsymbol{m}_\mathrm{c}$
$\boldsymbol{\Phi}_{\mathrm{c},i}$	(conditional) covariance corresponding to $\boldsymbol{m}_{\mathrm{c},i}$
$\boldsymbol{\phi}_{\mathrm{c},j}$	jth column of covariance matrix $\boldsymbol{\Phi}_\mathrm{c}$
$\sigma_{\mathrm{c},j}^2$	jth diagonal entry of covariance matrix $\boldsymbol{\Phi}_\mathrm{c}$
σ_c^2	average variance corresponding to $\boldsymbol{m}_\mathrm{c}$
$\sigma_{\mathrm{s},j}^2$	jth (conditional) variance corresponding to $m_{\mathrm{s},j}$
σ_s^2	average (conditional) variance corresponding to $\boldsymbol{m}_\mathrm{s}$
$\boldsymbol{m}_\mathsf{x}$	mean vector of signal $\mathbf{x}$
$m_{\mathsf{x},j}$	jth element of $\boldsymbol{m}_\mathsf{x}$
$\boldsymbol{\Phi}_\mathsf{x}$	(co)variance matrix of signal $\mathbf{x}$
$\sigma_{\mathsf{x},j}^2$	variance corresponding to x_j
σ_x^2	average variance corresponding to $\mathbf{x}$
$\boldsymbol{K}$	end-to-end cascade between $\boldsymbol{x}$ and $\boldsymbol{m}_\mathrm{c}$
$\boldsymbol{W}$	scaling matrix for bias compensation
w	scaling factor for bias compensation
$\tilde{\boldsymbol{x}}_\mathrm{c}$	input mean vector for channel-constrained estimation
$\tilde{x}_{\mathrm{c},j}$	jth element of $\tilde{\boldsymbol{x}}_\mathrm{c}$
$\tilde{\boldsymbol{x}}_\mathrm{s}$	input mean vector for signal-constrained estimation
$\tilde{x}_{\mathrm{s},j}$	jth element of $\tilde{\boldsymbol{x}}_\mathrm{s}$
$\tilde{\boldsymbol{\Phi}}_\mathrm{c}$	input (co)variance matrix corresponding to $\tilde{x}_{\mathrm{c},j}$
$\tilde{\sigma}_{\mathrm{c},j}^2$	jth input variance corresponding to $\tilde{x}_{\mathrm{c},j}$
$\tilde{\sigma}_\mathrm{c}^2$	average input variance corresponding to $\tilde{\boldsymbol{x}}_\mathrm{c}$

Symbol	Meaning
$\tilde{\sigma}^2_{\mathrm{s},j}$	jth input variance corresponding to $\tilde{x}_{\mathrm{s},j}$
σ^2_{s}	average input variance corresponding to $\tilde{\boldsymbol{x}}_{\mathrm{s}}$

F.2.4 Other Variables

Symbol	Meaning
$\mathrm{const.}$	arbitrary constant
$\boldsymbol{e}_j$	jth unit vector with zeros everywhere, except a 1 at jth position
$\boldsymbol{I}_m$	identity matrix of size $m \times m$
$\mathbf{0}$	null vector of respective dimension

F.2.5 Stochastics

Symbol	Meaning
$\mathsf{f}_{\mathsf{x}}(x)$	probability density function of random variable x
$\mathsf{f}_{\mathsf{x}\mid\mathsf{y}}(x)$	conditional probability density function of random variable x given a realization of random variable y
$\mathcal{N}(m_{\mathsf{x}},\, \sigma^2_{\mathsf{x}})$	Gaussian distribution with mean m_{x} and variance σ^2_{x}
$\mathcal{N}(\boldsymbol{m}_{\mathsf{x}},\, \boldsymbol{\Phi}_{\mathsf{x}})$	multi-dimensional Gaussian distribution with mean $\boldsymbol{m}_{\mathsf{x}}$ and covariance matrix $\boldsymbol{\Phi}_{\mathsf{x}}$
$\mathcal{U}([a,\, b])$	uniform distribution over the interval $[a,\, b]$
$\mathrm{E}\{\cdot\}$	expectation
$\mathrm{E}\{\cdot \mid \cdot\}$	conditional expectation
$\mathrm{E}_{\mathsf{x}\sim\mathsf{f}_{\mathsf{x}}}\{\cdot\}$	expectation of random variable x w.r.t. pdf $\mathsf{f}_{\mathsf{x}}(x)$
$\Pr\{\cdot\}$	probability

F.2.6 Operators

Symbol	Meaning
$\lvert\cdot\rvert$	absolute value of a scalar or cardinality of a set
$\lVert\cdot\rVert_p$	ℓ_p-norm
$\lVert\cdot\rVert_{\mathrm{F}}$	Frobenius norm
$[\boldsymbol{M}]_{jj}$	jth diagonal entry of matrix $\boldsymbol{M}$
$\mathrm{D}_{\mathrm{KL}}(\cdot\lVert\cdot)$	Kullback–Leibler divergence
$\mathbf{diag}(x_j)$	diagonal matrix with $x_1, \ldots, x_N$ as entries
$\mathbf{diag}(\boldsymbol{x})$	diagonal matrix with entries of vector $\boldsymbol{x}$ as entries
$\mathbf{diag}(\boldsymbol{M})$	vector with diagonal entries of matrix $\boldsymbol{M}$
$\delta(\cdot)$	Dirac delta function
$\mathrm{h}(\cdot)$	differential entropy of a distribution
$\kappa(\boldsymbol{A})$	condition number of matrix $\boldsymbol{A}$
$\mathcal{L}(\cdot)$	Lagrangian function
$\mathcal{L}_{\mathrm{D}}(\cdot)$	Lagrangian dual
$\log(\cdot)$	logarithmus naturalis (natural logarithm)
$\mathrm{M}_{\mathrm{A}}(x_j)$	arithmetic mean $\mathrm{M}_{\mathrm{A}}(x_j) = \frac{1}{N}\sum_{j=1}^{N} x_j$
$\mathrm{M}_{\mathrm{H}}(x_j)$	harmonic mean $\mathrm{M}_{\mathrm{H}}(x_j) = N/\sum_{j=1}^{N} 1/x_j$
$\nabla f(\boldsymbol{x})$	gradient of $f(\boldsymbol{x})$: $\nabla f(\boldsymbol{x}) = [\frac{\partial}{\partial x_1} f(\boldsymbol{x}), \ldots, \frac{\partial}{\partial x_N} f(\boldsymbol{x})]^\top$
$\mathcal{O}(\cdot)$	Landau notation, bound on growth of a function
$\pi(\cdot)$	permutation
$\mathrm{sgn}(\cdot)$	signum function
$\mathrm{trace}(\boldsymbol{M})$	trace of $N \times N$ matrix $\boldsymbol{M}$: $\mathrm{trace}(\boldsymbol{M}) = \sum_{j=1}^{N} [\boldsymbol{M}]_{jj}$

F.2.7 Exponential Families

Symbol	Meaning
$\boldsymbol{g}(\boldsymbol{x})$	sufficient statistic
$\boldsymbol{\theta}$	natural parameters
$Z(\boldsymbol{\theta})$	partition function
$k(\boldsymbol{x})$	carrier measure

F.2.8 Representatives

Symbol	Meaning
$\mathsf{q}(\boldsymbol{x})$	substitute distribution for posterior $\mathsf{f}_{\mathbf{x}\mid\mathbf{y}}(\boldsymbol{x})$
$\mathsf{q}_{\mathrm{c}}(\boldsymbol{x})$	representative for channel factor $\mathsf{f}_{\mathbf{y}\mid\mathbf{x}}(\boldsymbol{x})$
$\mathsf{q}_{\mathrm{c},i}(\boldsymbol{x})$	representative for $\mathsf{f}_{\mathsf{y}_i\mid\mathbf{x}}(\boldsymbol{x})$
$\mathsf{q}_{\mathrm{s}}(\boldsymbol{x})$	representative for signal factor $\mathsf{f}_{\mathbf{x}}(\boldsymbol{x})$
$\mathsf{q}_{\mathrm{s},j}(x_j)$	representative for $\mathsf{f}_{\mathsf{x}}(x_j)$, marginal of $\mathsf{q}_{\mathrm{s}}(\boldsymbol{x})$
$\mathsf{q}_{\mathrm{v}}(\boldsymbol{x})$	representative for variables $\boldsymbol{x}$
$\mathsf{q}_{\mathrm{v},j}(x_j)$	representative for variable x_j, marginal of $\mathsf{q}_{\mathrm{v}}(\boldsymbol{x})$

Bibliography

[Ban13] Paolo Banelli. Non-Linear Transformations of Gaussians and Gaussian-Mixtures with Implications on Estimation and Information Theory. *arXiv preprint arXiv:1111.5950*, 2013.

[BCDH10] Richard G. Baraniuk, Volkan Cevher, Marco F. Duarte, Chinmay Hegde. Model-Based Compressive Sensing. *IEEE Transactions on Information Theory*, vol. 56, no. 4, pp. 1982–2001, 2010.

[BD09] Thomas Blumensath, Mike E. Davies. Iterative Hard Thresholding for Compressed Sensing. *Applied and Computational Harmonic Analysis*, vol. 27, no. 3, pp. 265–274, 2009.

[Ber20] Ali Bereyhi. *Statistical Mechanics of Regularized Least Squares*. PhD thesis, Friedrich-Alexander-Universität Erlangen-Nürnberg, June 2020.

[Bet35] Hans A. Bethe. Statistical Theory of Superlattices. *Proceedings of the Royal Society of London A*, vol. 150, no. 871, pp. 552–575, 1935.

[BG96] Claude Berrou, Alain Glavieux. Near Optimum Error Correcting Coding and Decoding: Turbo-Codes. *IEEE Transactions on Communications*, vol. 44, no. 10, pp. 1261–1271, October 1996.

[BG18] Stefan C. Birgmeier, Norbert Goertz. Optimizing Approximate Message Passing for Variable Measurement Noise. In *European Signal Processing Conference (EUSIPCO)*, pp. 484–488, 2018.

[BG19] Stefan C. Birgmeier, Norbert Goertz. Exploiting General Multi-Dimensional Priors in Compressed-Sensing Reconstruction. In *12th International ITG Conference on Systems, Communications and Coding (SCC)*, pp. 1–6, 2019.

[BGT93] Claude Berrou, Alain Glavieux, Punya Thitimajshima. Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-Codes. In *Proceedings IEEE International Conference on Communications (ICC)*, pp. 1064–1070, Geneva, Switzerland, May 1993.

[Bir19] Stefan Birgmeier. *Message Passing For Multidimensional Inverse Problems*. PhD thesis, TU Wien, 2019.

[BM11] Mohsen Bayati, Andrea Montanari. The Dynamics of Message Passing on Dense Graphs, with Applications to Compressed Sensing. *IEEE Transactions on Information Theory*, vol. 57, no. 2, pp. 764–785, 2011.

[BMPP20] Alfredo Braunstein, Anna Paola Muntoni, Andrea Pagnani, Mirko Pieropan. Compressed Sensing Reconstruction using Expectation Propagation. *Journal of Physics A: Mathematical and Theoretical*, vol. 53, no. 18, 2020.

[Bro86] Lawrence D. Brown. *Fundamentals of Statistical Exponential Families: with Applications in Statistical Decision Theory*. Institute of Mathematical Statistics, 1986.

[BV04] Stephen P. Boyd, Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[CDS01] Scott Shaobing Chen, David L. Donoho, Michael A. Saunders. Atomic Decomposition by Basis Pursuit. *SIAM review*, vol. 43, no. 1, pp. 129–159, 2001.

[ÇLO22] Burak Çakmak, Yue M. Lu, Manfred Opper. Analysis of Random Sequential Message Passing Algorithms for Approximate Inference. *arXiv preprint arXiv:2202.08198*, 2022.

[CREK05] Shane F. Cotter, Bhaskar D. Rao, Kjersti Engan, Kenneth Kreutz-Delgado. Sparse Solutions to Linear Inverse Problems With Multiple Measurement Vectors. *IEEE Transactions on Signal Processing*, vol. 53, no. 7, pp. 2477–2488, 2005.

[CRT06] Emmanuel J. Candès, Justin Romberg, Terence Tao. Robust Uncertainty Principles: Exact Signal Reconstruction from Highly Incomplete Frequency Information. *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 489–509, 2006.

[CSD+17] Jun Won Choi, Byonghyo Shim, Yacong Ding, Bhaskar Rao, Dong In Kim. Compressed Sensing for Wireless Communications: Useful Tips and Tricks. *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1527–1550, 2017.

[CZK14] Francesco Caltagirone, Lenka Zdeborová, Florent Krzakala. On Convergence of Approximate Message Passing. In *Proceedings IEEE International Symposium on Information Theory (ISIT)*, pp. 1812–1816. IEEE, 2014.

[DB18] Guillaume Dehaene, Simon Barthelmé. Expectation Propagation in the Large Data Limit. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 80, no. 1, pp. 199–217, 2018.

[DDD04] Ingrid Daubechies, Michel Defrise, Christine De Mol. An Iterative Thresholding Algorithm for Linear Inverse Problems with a Sparsity Constraint. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, vol. 57, no. 11, pp. 1413–1457, 2004.

[DJB+95] Catherine Douillard, Michel Jézéquel, Claude Berrou, Annie Picart, Pierre Didier, Alain Glavieux. Iterative Correction of Intersymbol Interference: Turbo-Equalization. *European Transactions on Telecommunications*, vol. 6, no. 5, pp. 507–511, 1995.

[DMM09] David L. Donoho, Arian Maleki, Andrea Montanari. Message-Passing Algorithms for Compressed Sensing. *Proceedings of the National Academy of Sciences*, vol. 106, no. 45, pp. 18914–18919, 2009.

[DMM10] David L. Donoho, Arian Maleki, Andrea Montanari. Message Passing Algorithms for Compressed Sensing: I. Motivation and Construction. In *Proceedings IEEE Information Theory Workshop (ITW)*, pp. 1–5, 2010.

[Don06] David L. Donoho. Compressed Sensing. *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[DT97] G. B. Dantzig, M. N. Thapa. *Linear Programming 1: Introduction.* Springer Series in Operations Research. Springer, 1997.

[EK12] Yonina C. Eldar, Gitta Kutyniok. *Compressed Sensing: Theory and Applications.* Cambridge University Press, 2012.

[Fis22] Ronald A. Fisher. On the Mathematical Foundations of Theoretical Statistics. *Philosophical Transactions of the Royal Society of London. Series A, containing papers of a mathematical or physical character*, vol. 222, no. 594-604, pp. 309–368, 1922.

[Fis02] Robert F. H. Fischer. *Precoding and Signal Shaping for Digital Transmission.* John Wiley & Sons, 2002.

[Fis16] Robert F. H. Fischer. Notes on MMSE Estimation and Bias Compensation. Unpublished Manuscript, 2016.

[Fis18] Robert F. H. Fischer. Signal Theory. Lecture Notes, Ulm University, 2018.

[For01] G. David Forney. Codes on Graphs: Normal Realizations. *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 520–548, 2001.

[For03] G. David Forney. On the Role of MMSE Estimation in Approaching the Information-Theoretic Limits of Linear Gaussian Channels: Shannon meets Wiener. In *Allerton Conference on Communication, Control, and Computing*, pp. 430–439, 2003.

[For18] G. David Forney. Codes on graphs: Models for elementary algebraic topology and statistical physics. *IEEE Transactions on Information Theory*, vol. 64, no. 12, pp. 7465–7487, 2018.

[FS07] Robert F.H. Fischer, Cristian Siegl. Inflated Lattice Precoding, Bias Compensation, and the Uplink/Downlink Duality: The Connection. *IEEE Communications Letters*, vol. 11, no. 2, pp. 185–187, 2007.

[FS22] Robert F. H. Fischer, Carmen Sippel. Unbiasing in Iterative Reconstruction Algorithms for Discrete Compressed Sensing. In Gitta Kutyniok, Holger Rauhut, Robert J. Kunsch, editors, *Compressed Sensing in Information Processing*, Applied and Numerical Harmonic Analysis, chapter 6. Birkhäuser, 2022.

[FSG20] Robert F. H. Fischer, Carmen Sippel, Norbert Goertz. VAMP with Vector-Valued Diagonalization. In *Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, 2020.

[FSRS16] Alyson K. Fletcher, Mojtaba Sahraee-Ardakan, Sundeep Rangan, Philip Schniter. Expectation Consistent Approximate Inference: Generalizations and Convergence. In *Proceedings IEEE International Symposium on Information Theory (ISIT)*, pp. 190–194, 2016.

[Gal62] Robert Gallager. Low-Density Parity-Check Codes. *IRE Transactions on Information Theory*, vol. 8, no. 1, pp. 21–28, 1962.

[GFS63] Izrail Moiseevitch Gelfand, Sergej Vasilevitch Fomin, Richard A. Silverman. *Calculus of Variations*. Prentice-Hall, Inc., 1963.

[GH11] Qinghua Guo, Defeng David Huang. A Concise Representation for the Soft-In Soft-Out LMMSE Detector. *IEEE Communications Letters*, vol. 15, no. 5, pp. 566–568, 2011.

[Gra14] Knut Graichen. Methoden der Optimierung und optimalen Steuerung. Vorlesungsskript (in German), Universität Ulm, 2014.

[GSV05] Dongning Guo, Shlomo Shamai, Sergio Verdú. Mutual Information and Minimum Mean-Square Error in Gaussian Channels. *IEEE Transactions on Information Theory*, vol. 51, no. 4, pp. 1261–1282, 2005.

[GV96] Gene H. Golub, Charles F. Van Loan. *Matrix Computations*. John Hopkins University Press, 3 edition, 1996.

[GW06] Dongning Guo, Chih-Chun Wang. Asymptotic Mean-Square Optimality of Belief Propagation for Sparse Linear Systems. In *Proceedings IEEE Information Theory Workshop (ITW)*, pp. 194–198. IEEE, 2006.

[GWSV11] Dongning Guo, Yihong Wu, Shlomo Shamai Shitz, Sergio Verdú. Estimation in Gaussian Noise: Properties of the Minimum Mean-Square Error. *IEEE Transactions on Information Theory*, vol. 57, no. 4, pp. 2371–2385, 2011.

[HF14] Johannes B. Huber, Robert F.H. Fischer. Informationstheorie und deren Anwendungen zur Nachrichtenübertragung. Vorlesungsskript (in German), Friedrich-Alexander-Universität Erlangen-Nürnberg, 2014.

[HJ09] Roger A. Horn, Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, 2009.

[HOP96] Joachim Hagenauer, Elke Offer, Lutz Papke. Iterative Decoding of Binary Block and Convolutional Codes. *IEEE Transactions on Information Theory*, vol. 42, no. 2, pp. 429–445, 1996.

[HOW+05] Tom Heskes, Manfred Opper, Wim Wiegerinck, Ole Winther, Onno Zoeter. Approximate Inference Techniques with Expectation Constraints. *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2005, no. 11, 2005.

[HTP12] Jinping Hao, Filippo Tosato, Robert J. Piechocki. Sequential Compressive Sensing in Wireless Sensor Networks. In *IEEE 75th Vehicular Technology Conference (VTC Spring)*, pp. 1–5, 2012.

[Jay57] Edwin T. Jaynes. Information Theory and Statistical Mechanics. *Physical Review*, vol. 106, no. 4, pp. 620–630, 1957.

[Jay82] Edwin T. Jaynes. On the Rationale of Maximum-Entropy Methods. *Proceedings of the IEEE*, vol. 70, no. 9, pp. 939–952, 1982.

[Kay93] Steven M. Kay. *Fundamentals of Statistical Signal Processing: Estimation Theory*. Prentice-Hall, Inc., 1993.

[KF98] Frank R. Kschischang, Brendan J. Frey. Iterative Decoding of Compound Codes by Probability Propagation in Graphical Models. *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 219–230, 1998.

[KFL01] Frank R. Kschischang, Brendan J. Frey, H.-A. Loeliger. Factor Graphs and the Sum-Product Algorithm. *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 498–519, 2001.

[KL51] Solomon Kullback, Richard A. Leibler. On Information and Sufficiency. *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, March 1951.

[KMS+12] Florent Krzakala, Marc Mézard, Francois Sausset, Yifan Sun, Lenka Zdeborová. Probabilistic Reconstruction in Compressed Sensing: Algorithms, Phase Diagrams, and Threshold Achieving Matrices. *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2012, no. 08, 2012.

[KPN+14] Kee-Hoon Kim, Hosung Park, Jong-Seon No, Habong Chung, Dong-Joon Shin. Clipping Noise Cancelation for OFDM Systems Using Reliable Observations

Based on Compressed Sensing. *IEEE Transactions on Broadcasting*, vol. 61, no. 1, pp. 111–118, 2014.

[Lan09] Edmund Landau. *Handbuch der Lehre von der Verteilung der Primzahlen*, volume 1. B. G. Teubner, 1909.

[LCJ15] Markus Leinonen, Marian Codreanu, Markku Juntti. Sequential Compressed Sensing With Progressive Signal Reconstruction in Wireless Sensor Networks. *IEEE Transactions on Wireless Communications*, vol. 14, no. 3, pp. 1622–1635, 2015.

[LDH+07] Hans-Andrea Loeliger, Justin Dauwels, Junli Hu, Sascha Korl, Li Ping, Frank R. Kschischang. The Factor Graph Approach to Model-Based Signal Processing. *Proceedings of the IEEE*, vol. 95, no. 6, pp. 1295–1322, 2007.

[LHD20] Ning Li, Mengjia Huang, Zhongliang Deng. Elimination of Noise Distortion for OFDM Systems by Compressed Sensing Based on Distance Metric. *IEEE Access*, vol. 8, pp. 223700–223707, 2020.

[LHL+19] Xinghui Liu, Lingna Huy, Yue Liz, Lianghui Ding, Feng Yang, Cheng Zhi. Modified Peak Clipping and Compressed Sensing Recovery Scheme in OFDM System. In *2019 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, pp. 1–5. IEEE, 2019.

[LHLT15] Yingzhen Li, José Miguel Hernández-Lobato, Richard E. Turner. Stochastic Expectation Propagation. *Advances in Neural Information Processing Systems*, vol. 28, 2015.

[Loe04] H.-A. Loeliger. An Introduction to Factor Graphs. *IEEE Signal Processing Magazine*, vol. 21, no. 1, pp. 28–41, 2004.

[Mal11] Arian Maleki. *Approximate Message Passing Algorithms for Compressed Sensing*. PhD thesis, Stanford University, September 2011.

[MG98] Michael L. Moher, T. Aaron Gulliver. Cross-Entropy and Iterative Decoding. *IEEE Transactions on Information Theory*, vol. 44, no. 7, pp. 3097–3104, 1998.

[Min01] Thomas Peter Minka. Expectation Propagation for Approximate Bayesian Inference. In *Proceedings UAI-2001*, pp. 362–369, 2001.

[Min05] Thomas Peter Minka. Divergence Measures and Message Passing. Technical report, Microsoft Research, 2005.

[MKTZ15] Andre Manoel, Florent Krzakala, Eric Tramel, Lenka Zdeborovà. Swept Approximate Message Passing for Sparse Estimation. In *International Conference on Machine Learning*, pp. 1123–1132, 2015.

[MMC98] Robert J. McEliece, David J. C. MacKay, Jung-Fu Cheng. Turbo Decoding as an Instance of Pearl's "Belief Propagation" Algorithm. *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 140–152, 1998.

[Moh97a] Michael L. Moher. *Cross-Entropy and Iterative Detection.* PhD thesis, Carleton University, 1997.

[Moh97b] Michael L. Moher. Turbo-based Multiuser Detection. In *Proceedings IEEE International Symposium on Information Theory (ISIT)*, pp. 195–195, 1997.

[MP17] Junjie Ma, Li Ping. Orthogonal AMP. *IEEE Access*, vol. 5, pp. 2020–2033, 2017.

[MSW10] Dmitry M. Malioutov, Sujay R. Sanghavi, Alan S. Willsky. Sequential Compressed Sensing. *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 2, pp. 435–444, 2010.

[NG09] Frank Nielsen, Vincent Garcia. Statistical Exponential Families: A Digest with Flash Cards. *arXiv preprint arXiv:0911.4863*, 2009.

[NT09] Deanna Needell, Joel A. Tropp. CoSaMP: Iterative Signal Recovery from Incomplete and Inaccurate Samples. *Applied and Computational Harmonic Analysis*, vol. 26, no. 3, pp. 301–321, 2009.

[OW05] Manfred Opper, Ole Winther. Expectation Consistent Approximate Inference. *Journal of Machine Learning Research*, vol. 6, pp. 2177–2204, December 2005.

[Pea88] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann Publishers, INC., 1988.

[PP02] Athanasios Papoulis, S. Unnikrishna Pillai. *Probability, Random Variables, and Stochastic Processes.* McGraw-Hill, 2002.

[PRK93] Yagyensh Chandra Pati, Ramin Rezaiifar, Perinkulam Sambamurthy Krishnaprasad. Orthogonal Matching Pursuit: Recursive Function Approximation with Applications to Wavelet Decomposition. In *Proceedings of 27th Asilomar Conference on Signals, Systems and Computers*, pp. 40–44, 1993.

[Pro00] John G. Proakis. *Digital Communications.* McGraw-Hill, New York, NY, USA, 4th edition, 2000.

[PSC+15] Marcelo Pereyra, Philip Schniter, Emilie Chouzenoux, Jean-Christophe Pesquet, Jean-Yves Tourneret, Alfred O. Hero, Steve McLaughlin. A Survey of Stochastic Simulation and Optimization Methods in Signal Processing. *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 2, pp. 224–241, 2015.

[Ran11] Sundeep Rangan. Generalized Approximate Message Passing for Estimation With Random Linear Mixing. In *Proceedings IEEE International Symposium on Information Theory (ISIT)*, pp. 2168–2172, 2011.

[RFSK16] Sundeep Rangan, Alyson K. Fletcher, Philip Schniter, Ulugbek S. Kamilov. Inference for Generalized Linear Models via Alternating Directions and Bethe Free Energy Minimization. *IEEE Transactions on Information Theory*, vol. 63, no. 1, pp. 676–697, 2016.

[RSF17] Sundeep Rangan, Philip Schniter, Alyson K. Fletcher. Vector Approximate Message Passing. In *Proceedings IEEE International Symposium on Information Theory (ISIT)*, pp. 1588–1592, 2017.

[RSF19] Sundeep Rangan, Philip Schniter, Alyson K. Fletcher. Vector Approximate Message Passing. *IEEE Transactions on Information Theory*, vol. 65, no. 10, pp. 6664–6684, 2019.

[SBW+05] Shriram Sarvotham, Dror Baron, Michael Wakin, Marco F. Duarte, Richard G. Baraniuk. Distributed Compressed Sensing of Jointly Sparse Signals. In *Asilomar Conference on Signals, Systems, and Computers*, pp. 1537–1541, 2005.

[SD11] Henning F. Schepker, Armin Dekorsy. Sparse Multi-User Detection for CDMA Transmission Using Greedy Algorithms. In *8th International Symposium on Wireless Communication Systems*, pp. 291–295, 2011.

[SF14] Susanne Sparrer, Robert F. H. Fischer. Adapting Compressed Sensing Algorithms to Discrete Sparse Signals. In *18th International ITG Workshop on Smart Antennas (WSA)*, 2014.

[SF16] Susanne Sparrer, Robert F. H. Fischer. Enhanced Iterative Hard Thresholding for the Estimation of Discrete-Valued Sparse Signals. In *European Signal Processing Conference (EUSIPCO)*, pp. 71–75, 2016.

[SF17] Susanne Sparrer, Robert F. H. Fischer. Unveiling Bias Compensation in Turbo-Based Algorithms for (Discrete) Compressed Sensing. In *European Signal Processing Conference (EUSIPCO)*, pp. 2091–2095, 2017.

[SF18] Susanne Sparrer, Robert F. H. Fischer. Bias Compensation in Iterative Soft-Feedback Algorithms With Application to (Discrete) Compressed Sensing. In *IEEE Statistical Signal Processing Workshop (SSP)*, pp. 35–39, 2018.

[SF21] Carmen Sippel, Robert F. H. Fischer. VAMP with Individual Variances and Sequential Processing for Compressed Sensing. In *European Signal Processing Conference (EUSIPCO)*, Dublin, Ireland, August 2021.

[SF22] Carmen Sippel, Robert F. H. Fischer. Variants of VAMP for Signal Recovery in Wireless Sensor Networks. In *IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*, Athens, Greece, September 2022.

[Sha48] Claude E. Shannon. A Mathematical Theory of Communication. *Bell System Technical Journal*, vol. 27, pp. 379–423, July 1948.

[SJ80] John E. Shore, Rodney W. Johnson. Axiomatic Derivation of the Principle of Maximum Entropy and the principle of Minimum Cross-Entropy. *IEEE Transactions on Information Theory*, vol. 26, no. 1, pp. 26–37, 1980.

[SJ81] John Shore, Rodney Johnson. Properties of Cross-Entropy Minimization. *IEEE Transactions on Information Theory*, vol. 27, no. 4, pp. 472–482, 1981.

[SN08] Matthias W. Seeger, Hannes Nickisch. Compressed Sensing and Bayesian Experimental Design. In *Proceedings of the 25th International Conference on Machine Learning*, pp. 912–919, 2008.

[SN11] Matthias Seeger, Hannes Nickisch. Fast Convergent Algorithms for Expectation Propagation Approximate Bayesian Inference. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 652–660, 2011.

[Spa19] Susanne Sparrer. *Algorithms for (Discrete) Compressed Sensing—A Communications Engineering Perspective.* PhD thesis, Ulm University, September 2019.

[SSF13] Susanne Sparrer, Andreas Schenk, Robert F.H. Fischer. Communication over Impulsive Noise Channels: Channel Coding vs. Compressed Sensing. In *9th International ITG Conference on Systems, Communication and Coding (SCC)*, 2013.

[Tan81] R. Tanner. A Recursive Approach to Low Complexity Codes. *IEEE Transactions on Information Theory*, vol. 27, no. 5, pp. 533–547, 1981.

[Tib96] Robert Tibshirani. Regression Shrinkage and Selection via the LASSO. *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[TS11] Michael Tüchler, Andrew C. Singer. Turbo Equalization: An Overview. *IEEE Transactions on Information Theory*, vol. 57, no. 2, pp. 920–952, 2011.

[Ver10] Sergio Verdú. Mismatched estimation and relative entropy. *IEEE Transactions on Information Theory*, vol. 56, no. 8, pp. 3712–3720, 2010.

[VS13] Jeremy P. Vila, Philip Schniter. Expectation-Maximization Gaussian-Mixture Approximate Message Passing. *IEEE Transactions on Signal Processing*, vol. 61, no. 19, pp. 4658–4672, 2013.

[Wib96] Niclas Wiberg. *Codes and Decoding on General Graphs.* PhD thesis, Department of Electrical Engineering, Linköping University Sweden, 1996.

[WJ08] Martin J. Wainwright, Michael I. Jordan. Graphical Models, Exponential Families, and Variational Inference. *Foundations and Trends® in Machine Learning*, vol. 1, no. 1–2, pp. 1–305, 2008.

[Wu12] Nailong Wu. *The Maximum Entropy Method*, volume 32. Springer Verlag, 2012.

[WV12] Yihong Wu, Sergio Verdú. Optimal Phase Transitions in Compressed Sensing. *IEEE Transactions on Information Theory*, vol. 58, no. 10, pp. 6241–6263, 2012.

[XDS13] Tong Xue, Xiaodai Dong, Yi Shi. Multiple Access and Data Reconstruction in Wireless Sensor Networks Based on Compressed Sensing. *IEEE Wireless Communications (Magazine)*, vol. 12, no. 7, pp. 3399–3411, 2013.

[YFW05] Jonathan S. Yedidia, William T. Freeman, Yair Weiss. Constructing Free-Energy Approximations and Generalized Belief Propagation Algorithms. *IEEE Transactions on Information Theory*, vol. 51, no. 7, pp. 2282–2312, 2005.

[YMG08] Jennifer Yick, Biswanath Mukherjee, Dipak Ghosal. Wireless Sensor Network Survey. *Computer Networks*, vol. 52, no. 12, pp. 2292–2330, 2008.

List of Scientific Publications and Presentations

[1] Jonathan Bechter, Carmen Sippel, Christian Waldschmidt. Bats-inspired Frequency Hopping for Mitigation of Interference between Automotive Radars. In *IEEE MTT-S International Conference on Microwaves for Intelligent Mobility (ICMIM)*, San Diego, CA, USA, May 2016.

[2] Carmen Sippel, Sven Puchinger, Robert F.H. Fischer. Differential Linear Network Coding in Slowly-Varying Networks. Presentation at *33. Sitzung der ITG-Fachgruppe "Angewandte Informationstheorie", Codes, Lattices, and Decoding*, Ulm, Germany, March 2019.

[3] Carmen Sippel, Sven Puchinger, Robert F.H. Fischer. Differential Linear Network Coding in Slowly-Varying Networks. Poster at *the 19th Joint Workshop on Communications and Coding (JWCC)*, Kühtai, Austria, March 2019.

[4] Carmen Sippel, Sven Puchinger, Robert F.H. Fischer. Differential Linear Network Coding in Slowly-Varying Networks. Presentation at *Stuttgart Workshop on Coding and Communications (SWCC)*, Stuttgart, Germany, July 2019.

[5] Carmen Sippel, Cornelia Ott, Sven Puchinger, Martin Bossert. Reed–Solomon Codes over Fields of Characteristic Zero. In *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, pp. 1537–1541, July 2019.

[6] Carmen Sippel, Robert F.H. Fischer. On the Invariance of Recovery Algorithms for Compressed Sensing based on Expectation-Consistent Approximate Inference. In *Proceedings of the 24th International ITG Workshop on Smart Antennas (WSA)*, Hamburg, Germany, February 2020.

[7] Robert F.H. Fischer, Carmen Sippel, Norbert Goertz. VAMP with Vector-Valued Diagonalization. In *Proceedings of the IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP)*, Barcelona, Spain, May 2020.

[8] Carmen Sippel, Robert F.H. Fischer. VAMP with Individual Variances and Sequential Processing for Compressed Sensing. In *Proceedings of the 29th European Signal Processing Conference (EUSIPCO)*, pp. 1935–1939, Dublin, Ireland, August 2021.

[9] Carmen Sippel, Robert F.H. Fischer. Stabilization Techniques for Iterative Algorithms in Compressed Sensing. Online available at arxiv, `https://arxiv.org/abs/2109.14917`, 2021.

[10] Carmen Sippel, Robert F.H. Fischer. Variants of VAMP for Signal Recovery in Wireless Sensor Networks. In *Proceedings of the IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*, Athens, Greece, September 2022.

[11] Carmen Sippel, Robert F.H. Fischer. Comparison of Parallel and Sequential Algorithms for Compressed Sensing. In *Proceedings of the 7th International Conference on Frontiers of Signal Processing (ICFSP)*, Paris, France, September 2022.

[12] Carmen Sippel, Robert F.H. Fischer. Evaluation of the Fractional Approach for Iterative Algorithms in Compressed Sensing. In *Proceedings of the International ITG 26th Workshop on Smart Antennas (WSA) and 13th Conference on Systems, Communications, and Coding (SCC)*, Braunschweig, Germany, February 2023.

[13] Elena Sterk, Carmen Sippel, Robert F.H. Fischer. Comparison of Damping Approaches for AMP. In *Proceedings of the International ITG 26th Workshop on Smart Antennas (WSA) and 13th Conference on Systems, Communications, and Coding (SCC)*, Braunschweig, Germany, February 2023.

Curriculum Vitæ

For data protection reasons, the curriculum vitæ has been removed from the online version.

ISBN: 978-3-948303-38-9