This thesis has been submitted in fulfilment of the requirements for a postgraduate degree (e.g. PhD, MPhil, D Clin Psychol) at the University of Edinburgh. Please note the following terms and conditions of use:

- This work is protected by copyright and other intellectual property rights, which are retained by the thesis author, unless otherwise stated.
- A copy can be downloaded for personal non-commercial research or study, without prior permission or charge.
- This thesis cannot be reproduced or quoted extensively from without first obtaining permission in writing from the author.
- The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the author.
- When referring to this work, full bibliographic details including the author, title, awarding institution and date of the thesis must be given.
Upscaling Message Passing
Algorithms

Nikolajs Skuratovs

A thesis submitted for the degree of Doctor of Philosophy

The University of Edinburgh

November 2022
Dedicated to my parents and Anna Cvetkova
Abstract

The development of Approximate Message Passing (AMP) has become a precedent demonstrating the potential of Message Passing (MP) algorithms in solving large-scale linear inverse problems of the form $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{w}$. Not only AMP is provably convergent and Bayes-optimal, but it is also a first-order iterative method that can leverage Plug-and-Play (PnP) denoisers for recovering complex data like natural images. Unfortunately, all of these properties have been shown to hold assuming the measurement operator $\mathbf{A}$ is, roughly speaking, an i.i.d. random matrix, which highly limits the applicability of the algorithm. The promising extension of AMP, Vector AMP (VAMP), can handle a much broader range of $\mathbf{A}$ while preserving most advantages of AMP, but the algorithm requires inverting a large-scale matrix at each iteration, which makes it computationally intractable. As a result, a wide range of ideas has been proposed on upscaling VAMP while preserving its optimality and generality, and in this thesis we would like to share our contributions in this regard.

The first contribution is related to developing a stable and accelerated version of Conjugate Gradient (CG) VAMP (CG-VAMP) – the VAMP algorithm, where the matrix inversion is approximated by the CG algorithm. The originally proposed version of CG-VAMP exhibits unstable dynamics when even mildly large number of CG iterations is used and in those regimes where CG-VAMP is stable, the resulting fixed point of the algorithm might be much worse than that of VAMP. To allow CG-VAMP to use by an order more CG iterations and approximate VAMP with an almost arbitrary accuracy, we constructed a series of rigorous tools that have a negligible computational cost and that lead to stable performance of the algorithm. Additionally, we developed a combination of stopping criteria for CG that ensures efficient operation of CG-VAMP and faster time-wise convergence
without sacrificing the estimation accuracy.

Next, we considered an alternative way of pushing the performance of CG-VAMP closer to VAMP’s and developed the warm-started CG (WS-CG) that reuses the information generated at the previous outer-loop iterations of the MP algorithm. We show that when the matrix inverse in VAMP is approximated by WS-CG, a fixed point of WS-CG VAMP (WS-CG-VAMP) is a fixed point of VAMP and, therefore, is conjectured to be Bayes-optimal. Importantly, this result is invariant with respect to the number of WS-CG iterations and the resulting algorithm can have the computational cost of AMP while being general and optimal as VAMP. We extend the tools developed for CG to WS-CG and numerically demonstrate the stability and efficiency of WS-CG-VAMP.

The final contribution is the development of alternative methods for estimating the divergence of a PnP denoiser used within MP algorithms. This divergence plays a crucial role in stabilizing MP algorithms and ensuring its optimality and predictability. So far, the only suggested method for constructing an estimate of the divergence of a PnP denoiser has been the Black-Box Monte Carlo method [1]. The main drawback of this method is that it requires executing the denoiser an additional time, which, effectively, doubles the cost of most MP algorithms. In this thesis we propose two rigorous divergence estimation methods that avoid such a problem and utilize only the information circulated in every MP algorithm.
Lay summary

In the past decades, the resolution of visual information has been rapidly growing and many practical imaging applications require more intelligent algorithms to handle the increasing amount of data. Computer Tomography (CT) is one of such applications where the goal is to obtain a high quality image of a part of the body to do further diagnostic procedures. Unfortunately, scanning with CT has multiple adverse effects, including radiation-induced cancer and the more a person is exposed to the scanning beam, the more radiation he experiences. Such a drawback leads to the desire to keep a person in the CT scan for a shorter amount of time to reduce the damage. However, in this case we would obtain not a full image of the body, but only a part of the image. Roughly speaking, the fewer scanning beams are used, the fewer pixels of the image we obtain. Therefore, in such applications we need an algorithm that can recover the whole image using only small portion of it.

In this thesis we propose a series of such algorithms and additional tools to make them faster and more accurate. As a starting point, we use the algorithm called Vector Approximate Message Passing (VAMP) that was theoretically shown to be able to obtain the best accuracy of reconstructing a whole image from a small portion of samples. Unfortunately, the computational cost of this algorithm grows too quickly with the size of the image and recovering high-resolution data is computationally infeasible. Therefore, one of the main contribution of this thesis is the proposal of VAMP-like algorithms that can do accurate reconstruction of high-resolution images from a small portion of samples in tens of seconds. Additionally, we propose a series of tools for accelerating these algorithms and making them produce consistent results.
Declaration of originality

I declare that this thesis has been composed solely by myself and that it has not been submitted, in whole or in part, in any previous application for a degree. Except where states otherwise by reference or acknowledgment, the work presented is entirely my own.

Nikolajs Skuratovs
Acknowledgements

The four years of my PhD studies have allowed me to become a part of the research community and have been an amazing experience. That would not be possible without the support of the following people.

First and foremost, I am beyond grateful to professor Mike Davies for putting his trust in me and granting me an opportunity to learn under his supervision. From the very beginning of my PhD studies and until the completion of this thesis, Mike provided me with full support and was always willing to share constructive and fair feedback that helped me to grow. He was also able to create an atmosphere of trust, and gave me a chance to be more independent in my work and choose my own direction for the research. He has given me the freedom to make mistakes and learn from them. But whenever I felt like I had run out of ideas, I knew I could always talk to Mike and return with a new perspective and multiple ideas on how to move forward. This included not only the research related questions, but also the personal challenges that I experienced throughout my studies. To me, Mike has become a role model of a researcher and a leader that sincerely loves his job, values his colleagues, and inspires everyone around him, including myself.

I would also like to thank Dr. James Hopgood and Dr. Mehrdad Yaghoobi for their support and advice on overcoming the different challenges that come with undergoing a PhD programme. Apart from that, I truly appreciate James, Mehrdad and Mike trusting me with assisting them during lectures and in other parts of the teaching process. This experience has helped me improve my soft skills invaluable for my future career and allowed me to give something back to society.

Additionally, I want to thank my parents for their complete support in my decision to move to Edinburgh to broaden my horizons and learn more about other
people and myself. I highly appreciate all those hundreds of hours of conversation and thousands of messages between us that helped me tremendously through all the challenges in the last five years.

Last but not least, I am grateful to fate for having met Anna Cvetkova who has inspired me to learn more about the world, to be more open to different perspectives and ideas. Words cannot express my gratitude for your support, patience at times when I was being stubborn, for all the effort you have put into cheering me up during my lows and for all the great moments we have had together. My journey started with you, it has been completed with you and I am looking forward to moving into our future together.
Contents

Abstract ................................................................. iii
Lay summary ............................................................. iv
Declaration ............................................................... v
Acknowledgements ....................................................... vi
Contents ................................................................. viii
List of Figures .......................................................... xii
List of Tables ............................................................ xiv
Acronyms ................................................................. xv
Nomenclature ........................................................... xviii

1 Introduction ......................................................... 1
  1.1 Introduction ..................................................... 1
  1.2 Main Contributions ............................................. 4
  1.3 Publications .................................................... 5
  1.4 Organization of the thesis ..................................... 6

2 Background ......................................................... 8
  2.1 Low dimensional Signals, Sampling and Algorithms .......... 8
    2.1.1 Sparsity and Compressibility .............................. 9
    2.1.2 Sampling .................................................. 11
    2.1.3 Reconstruction algorithms for sparse signals .......... 16
  2.2 Message Passing for CS ....................................... 19
    2.2.1 Belief propagation as an instance of Expectation Propagation 20
    2.2.2 Approximate Message Passing ............................. 23
    2.2.3 Rigorous dynamics of AMP ............................... 25
2.2.4 State Evolution and other properties of AMP 30
2.2.5 Denoisers for image reconstruction 33
2.2.6 Limitations of AMP 35
2.3 Expectation Propagation and Vector AMP 36
2.3.1 Error Model and Asymptotic Properties of EP 41
2.3.2 Vector Approximate Message Passing 46
2.3.3 Limitation of VAMP 47
2.4 Upscaling VAMP and Conjugate Gradient VAMP 47
2.4.1 State Evolution and Onsager correction in CG-VAMP 50
2.4.2 Drawbacks and stability issues of CG-VAMP 52
2.5 Conclusions 53

3 Stable and efficient implementation of CG-VAMP 55
3.1 Practical estimation of $\gamma_{t,i}^A$ in CG-VAMP 56
3.2 Practical estimation of $\psi_{t,i}^A\rightarrow B$ in CG-VAMP 58
3.3 Adaptive CG in CG-VAMP 60
3.4 Simulation experiments 62
  3.4.1 Comparison of different CG-VAMP algorithms 64
  3.4.2 CG-VAMP with Adaptive CG 65
3.5 Conclusions 67

4 Warm-starting in MP 69
4.1 Warm-starting the CG algorithm in CG-VAMP 70
  4.1.1 Rigorous dynamics of Block A in WS-CG-VAMP 72
  4.1.2 Robust estimation of Block A parameters in WS-CG-VAMP 76
  4.1.3 Approximate update $x_{t}^{A\rightarrow B}$ in WS-CG-VAMP 79
4.2 Generalized Warm-Start Framework 79
  4.2.1 Equivalent SLE in VAMP 80
  4.2.2 Long-memory LMMSE approximation 81
  4.2.3 Memory AMP 83
  4.2.4 Asymptotic behaviour of WS methods within Message Passing 84
  4.2.5 Additional computational costs associated with Long-Memory MP 87
4.3 Numerical Experiments ............................................. 89
  4.3.1 Comparison of different WS-CG-VAMP algorithms .......... 89
  4.3.2 The SE for WS-CG in WS-CG-VAMP .......................... 91
  4.3.3 WS-CG-VAMP and WS-GD-VAMP ............................. 92
  4.3.4 Comparison with MAMP ................................. 95
4.4 Conclusions ..................................................... 96

5 Divergence estimation in Message Passing algorithms 99
  5.1 SMP algorithms and their main asymptotic properties ......... 102
  5.2 Efficient estimation of the divergence in SMP algorithms .... 105
    5.2.1 Intuition .................................................. 106
    5.2.2 Algebraic divergence estimator .......................... 108
    5.2.3 Polynomial divergence estimator .......................... 112
    5.2.4 Root identification for the polynomial estimator ......... 116
    5.2.5 Theoretical identification of the root .................... 118
    5.2.6 Implementation details of the polynomial estimator ....... 123
  5.3 Simulation experiments ........................................ 124
    5.3.1 Polynomial vs algebraic estimators ...................... 125
    5.3.2 Black Box Monte Carlo and the polynomial methods for di-
        vergence estimation ........................................ 127
    5.3.3 AMP ......................................................... 129
    5.3.4 CG-VAMP with Adaptive CG and the Polynomial diver-
        gence estimator ........................................... 130
  5.4 Conclusions ..................................................... 131

6 Conclusions and further work 133
  6.1 Further work .................................................. 135
    6.1.1 Warm-starting and LM MP algorithms ...................... 135
    6.1.2 Divergence estimation in MP ............................ 139
    6.1.3 Extending the range of inverse problems .................. 141

A Supplementary material 143
CONTENTS

B Proofs of Lemma 1 and Theorem 2
   B.1 Proof of Lemma 1 ........................................ 147
   B.2 Proof of Theorem 2 ...................................... 149

C Proof of Theorem 3 ............................................ 152

D Proofs of Theorem 4 and of Theorem 8
   D.1 Proof of Theorem 8 ........................................ 162

E Proofs of Theorem 5 and of Theorem 9
   E.1 Asymptotics of $a_l^j$ and $b_l^j$ in WS-CG-VAMP .... 167
   E.2 Proof of Theorem 9 ......................................... 169

F Proofs of Theorem 6 and Theorem 10
   F.1 Proof of Theorem 10 ...................................... 174

G Proof of Lemma .................................................. 176

H Proof of Lemma 3 ................................................ 179

I Proof of Theorem 13
   I.1 Analysis of (I.17) ......................................... 184
   I.2 Analysis of (I.18) ......................................... 184
   I.3 Roots of (I.16) ............................................ 189

J Proof of the asymptotic equivalence of (5.69) and (5.70)
   ................................................................. 192

K Proof of Theorem 14
   K.1 MF-OAMP .................................................. 197
   K.2 VAMP ....................................................... 198

L Proof of Lemma 4 ................................................ 199
List of Figures

2.1 A a natural image (a) and its db4 wavelet decomposition (b); (c) represents the image formed out of the top 5% magnitude-wise wavelet coefficients of the original image. 11
2.2 QQ plot of $h_t$ for AMP (left) and for ISTA (right) at iteration $t = 5$. The size of the problem is $N = 2^{20}$ and the number of measurements $M = 2^{19}$. 29
2.3 NMSE of the estimates produced by AMP and FISTA against the iteration number $t$. 32
2.4 Diagram of the EP algorithm for CS. 40
3.1 NMSE versus outer-loop iteration number $t$ for different types of CG-VAMP algorithms. 65
3.2 The number of CG iterations $i|t|$ that follow only the stopping criterion (3.16) (red curve) and that follows both (3.16) and (3.17) (blue curve). 67
3.3 The NMSE and the computational time of CG-VAMP with ACG with three types of stopping criteria. 68
4.1 The NMSE of VAMP, CG-VAMP B and different versions of WS-CG-VAMP versus outer-loop iteration $t$ for different condition numbers $\kappa(A)$ of $A$. 91
4.2 The normalized error of the estimator based on (4.11) in WS-CG-VAMP with \( i = 1 \) and \( i = 5 \) respectively. The main line corresponds to the mean of the normalized error averaged over 10 realizations, while the error bars correspond to the standard deviation. The error is clipped to \( 10^{-4} \).

4.3 NMSE of WS-CG-VAMP B and WS-GD-VAMP B algorithms with different damping lengths.

4.4 NMSE over iteration number \( t \) for the LM MP algorithms (4.28)-(4.29) with different WS approximators.

4.5 NMSE over iteration number \( t \) for different Message Passing algorithms.

5.1 Divergence estimation error with the standard deviation error bars of the polynomial and the algebraic estimators.

5.2 Left: the NMSE of CG-VAMP algorithms with the two correction methods. Right: divergence estimation error with the standard deviation error bars of the polynomial and the algebraic estimators.

5.3 NMSE with the standard deviation error bars for two CG-VAMP algorithms and different condition numbers \( \kappa(\mathbf{A}) \) of \( \mathbf{A} \). The BB-MC divergence estimator corresponds to (5.2), while the polynomial divergence estimator is computed from (5.68).

5.4 Mean error with the standard deviation error bars of estimating the divergence \( \alpha_t \) within CG-VAMP.

5.5 NMSE of VAMP, CG-VAMP and WS-CG-VAMP algorithms using two different divergence estimation methods.

5.6 NMSE of AMP with two different divergence estimation methods.

5.7 The NMSE and the computational time of CG-VAMP with the polynomial divergence estimator and with ACG using three types of stopping criteria.
List of Tables

3.1 Software and hardware characteristics of the computational platform 64
3.2 Time (seconds) taken for CG-VAMP B with \( i = \{5, 10, 30, 200\} \) iterations to converge to their respective fixed points 65
4.1 Comparison of Long Memory MP algorithms 88
4.2 Computational and memory costs of the linear part of CG-VAMP and WS-CG-VAMP 89
4.3 A summary of four types of WS-CG-VAMP with different sets of Onsager corrections and methods of estimating those corrections 90
4.4 A summary of two pairs of WS-CG-VAMP and WS-GD-VAMP with different estimators for \( v_h \) and \( \{\hat{\gamma}_A^{\tau,j}\}_{\tau=0}^\tau \) 93
5.1 Time (in seconds) taken for SMP algorithms with two different divergence estimation methods to execute 15 iterations 129
## Acronyms and Abbreviations

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACG</td>
<td>Adapted Conjugate Gradient</td>
</tr>
<tr>
<td>AMP</td>
<td>Approximate Message Passing</td>
</tr>
<tr>
<td>BB-MC</td>
<td>Black-Box Monte Carlo</td>
</tr>
<tr>
<td>BP</td>
<td>Belief Propagation</td>
</tr>
<tr>
<td>BPDN</td>
<td>Basis Pursuit Denoising</td>
</tr>
<tr>
<td>CG</td>
<td>Conjugate Gradient</td>
</tr>
<tr>
<td>CG-VAMP</td>
<td>Conjugate Gradient VAMP</td>
</tr>
<tr>
<td>CoSaMP</td>
<td>Compressive Sampling Matching Pursuit</td>
</tr>
<tr>
<td>CS</td>
<td>Compressed Sensing</td>
</tr>
<tr>
<td>CT</td>
<td>Computed Tomography</td>
</tr>
<tr>
<td>DFT</td>
<td>Discrete Fourier transform</td>
</tr>
<tr>
<td>DnCNN</td>
<td>Denoising CNN</td>
</tr>
<tr>
<td>EP</td>
<td>Expectation Propagation</td>
</tr>
<tr>
<td>FIJLT</td>
<td>Fast ill-conditioned Johnson Lindenstrauss</td>
</tr>
<tr>
<td></td>
<td>Transform</td>
</tr>
<tr>
<td>FISTA</td>
<td>Fast ISTA</td>
</tr>
<tr>
<td>GD</td>
<td>Gradient Descent</td>
</tr>
</tbody>
</table>

xvi
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GLM</td>
<td>Generalized Linear Models</td>
</tr>
<tr>
<td>GOE</td>
<td>Gaussian Orthogonal Ensemble</td>
</tr>
<tr>
<td>GSC</td>
<td>Gram-Schmidt Conjugation</td>
</tr>
<tr>
<td>i.i.d.</td>
<td>independent and identically distributed</td>
</tr>
<tr>
<td>IHT</td>
<td>Iterative Hard Thresholding</td>
</tr>
<tr>
<td>ISTA</td>
<td>Iterative Soft Thresholding Algorithm</td>
</tr>
<tr>
<td>KL</td>
<td>Kullback–Leibler</td>
</tr>
<tr>
<td>LASSO</td>
<td>Least Absolute Shrinkage and Selection Operator</td>
</tr>
<tr>
<td>LIHT</td>
<td>Learned IHT</td>
</tr>
<tr>
<td>LISTA</td>
<td>Learned ISTA</td>
</tr>
<tr>
<td>LM</td>
<td>long-memory</td>
</tr>
<tr>
<td>LMMSE</td>
<td>Linear Minimum Mean Squared Error</td>
</tr>
<tr>
<td>LSL</td>
<td>Large System Limit</td>
</tr>
<tr>
<td>MAMP</td>
<td>Memory AMP</td>
</tr>
<tr>
<td>MF-OAMP</td>
<td>Matched Filter Orthogonal AMP</td>
</tr>
<tr>
<td>MMSE</td>
<td>Minimum Mean Squared Error</td>
</tr>
<tr>
<td>MP</td>
<td>Message Passing</td>
</tr>
<tr>
<td>MRI</td>
<td>Magnetic Resonance imaging</td>
</tr>
<tr>
<td>NLM</td>
<td>Non-Local Means</td>
</tr>
<tr>
<td>NMSE</td>
<td>Normalized Mean Squared Error</td>
</tr>
<tr>
<td>PnP</td>
<td>Plug-and-Play</td>
</tr>
<tr>
<td>RIP</td>
<td>Restricted Isometry Property</td>
</tr>
<tr>
<td>ROI</td>
<td>Right-Orthogonally Invariant</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>SE</td>
<td>State Evolution</td>
</tr>
<tr>
<td>SLE</td>
<td>system of linear equations</td>
</tr>
<tr>
<td>SMP</td>
<td>Scalable Message Passing</td>
</tr>
<tr>
<td>SURE</td>
<td>Stein’s Unbiased Risk Estimator</td>
</tr>
<tr>
<td>SVD</td>
<td>Singular Value Decomposition</td>
</tr>
<tr>
<td>VAMP</td>
<td>Vector AMP</td>
</tr>
<tr>
<td>WS-CG</td>
<td>Warm-Started Conjugate Gradient</td>
</tr>
<tr>
<td>WS-CG-VAMP</td>
<td>Warm-Started Conjugate Gradient VAMP</td>
</tr>
<tr>
<td>WS-GD</td>
<td>Warm-Started Gradient Descent</td>
</tr>
<tr>
<td>WS-GD-VAMP</td>
<td>Warm-Started Gradient Descent VAMP</td>
</tr>
</tbody>
</table>
Nomenclature

\((v_t)_i\) \(i\)-th element of a vector \(v_t\)

\((\cdot)\) operand

\((\cdot)^T\) Transpose

\((\cdot)^\dagger\) Left Moore-Penrose (pseudo) inverse

\(A\) Measurement operator

\(D_{(i,j)}\) \(j\)-th column of a matrix \(D\)

\(I\) Identity matrix

\(v_t\) a vector indexed by an integer \(t\)

\(w\) Measurement noise

\(x\) Signal vector

\(x \cdot y\) Inner product of vectors \(x\) and \(y\)

\(y\) Measurement vector

\(\chi_i\) \(i\)-th spectral moment of \(A^T A\)

\(\delta\) Compression rate

\(\delta_{(\cdot,\cdot)}\) Dirac delta function

\(E[\cdot]\) Expectation operator
NOMENCLATURE

\( \hat{\cdot} \) Estimated value

\( \kappa(A) \) Condition number of \( A \)

\( \ll \) Much less

\( \| \cdot \| \) \( l_2 \) norm

\( \| \cdot \|_0 \) \( l_0 \) semi-norm

\( \| \cdot \|_k \) \( l_k \) norm

\( \mathbb{C} \) Complex number

\( \mathbb{R} \) Real number

\( \nabla_x \) Gradient operator with respect to a vector \( x \)

\( \nabla_x \cdot f(x) \) Divergence of a function \( f(x) \) with respect to \( x \)

\( \mathcal{N}(m, \Sigma) \) Gaussian density with mean \( m \) and covariance \( \Sigma \)

\( a.s \equiv \) Almost surely equals

\( a.s \sim \) Almost surely distributed as

\( \sim \) Distributed as

\( \text{Var}[\cdot] \) Variance operator

\( | \cdot | \) Absolute value

\( \overset{(d)}{\Rightarrow} \) Converges in distribution

\( \{v_j\}_{j=j_{\text{min}}}^{j_{\text{max}}} \) The elements \( j_{\text{min}}, j_{\text{min}} + 1, \ldots, j_{\text{max}} \) of a vector \( v \)

\( \text{card}(S) \) cardinality of a set \( S \)

\( D_{i,j} \) \((i,j)\)-th element of a matrix \( D \)

\( f_{\setminus k}(x) \) Factorization as \( \prod_{i \neq k} f_i(x) \) without the \( k \)-th factor \( f_k(x) \)

\( M \) Number of measurements

\( \text{xx} \)
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N$</td>
<td>Signal length</td>
</tr>
<tr>
<td>$supp(v)$</td>
<td>Support of a vector $v$</td>
</tr>
<tr>
<td>$Tr{\cdot}$</td>
<td>Trace</td>
</tr>
<tr>
<td>$v_i$</td>
<td>$i$-th element of a vector $v$</td>
</tr>
<tr>
<td>$v_w$</td>
<td>Variance of the measurement noise $w$</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Introduction

In the modern world, the amount of data generated, transmitted, received and processed is monotonically increasing. Partly, this can be explained by the desire to perceive and work with more accurate data, which in some cases is related to the resolution of this data. As an example, we would like to consume higher resolution audio, image and video contents, while preserving the same transmission/reception and processing speeds. One way to approach this complexity challenge is to design more powerful hardware. The alternative approach and the focus of this thesis is to develop analytical ideas that would leverage the structure present in the original data to represent it in a more compact form with the ability to recover it with no or satisfactory distortion. This approach involves two stages: 1) designing a compression method that extracts the essential information from the data and 2) building a reconstruction algorithm that recovers the original data of high ambient dimension from much fewer samples. In the signal processing community, these tasks are sometimes referred to as Compressed Sensing (CS).

While there are many different sampling/representation approaches, in this thesis we focus on recovering a real data vector $\mathbf{x} \in \mathbb{R}^N$ of dimension $N$ from a set of $M$ linear measurements

$$y_m = \mathbf{a}_m^T \mathbf{x} \quad m \in (1, \ldots, M),$$

(1.1)
where \( a_m \in \mathbb{R}^N \). Stacking all the measurements (1.1) together results in the following underdetermined linear system

\[
\mathbf{y} = \mathbf{Ax},
\]

where \( \mathbf{y} = (y_1, ..., y_M)^T \) is the measurement vector and \( \mathbf{A} = (a_1, ..., a_M)^T \) is the measurement operator that we assume to be known. This kind of a system can be found in the applications like Computed Tomography (CT) imaging where \( \mathbf{A} \) is the Radon transform, or in the Magnetic Resonance imaging (MRI) where the measurement operator represents the Fourier matrix.

The above formalism allows us to consider the applications like CT, MRI and others from the mathematical perspective, since now our task boils down to solving the system of linear equations (1.2) for \( \mathbf{x} \). Unfortunately, we immediately face the fundamental problem of this task: if the number of measurements, i.e. the number of linear equations, \( M \) is smaller then the number of unknowns \( N \), then there are infinitely many solutions \( \mathbf{x} \) such that satisfy the observation \( \mathbf{y} = \mathbf{Ax} \). Therefore, in a general case, solving this inverse problem, i.e. recovering the signal \( \mathbf{x} \) from (1.2), is an ill-posed problem [2].

There have been developed multiple perspectives on approaching the above inverse problem and most of them, in one way or another, assume two properties of the system (1.2). The first one concerns the signal itself: we assume that \( \mathbf{x} \) possesses some strong structure so that its actual number of degrees of freedom \( k \) is much smaller than the ambient dimensionality \( N \). A classical example of such a model would be sparsity. A sparse vector, or rather \( k \)-sparse to emphasize the level of sparsity, is such a vector that has at most \( k \) non-zero entries and it is a natural opposite of a dense vector where the major part of the entries are non-zero. For example, natural images are usually dense in the ambient domain and recovering an image \( \mathbf{x} \) with \( N^2 \) pixels from \( M < N^2 \) measurements might seem infeasible at the first glance. However, it turns out that natural images tend to be approximately sparse or compressible in the Wavelet domain, i.e. the Wavelet transformed image \( \mathbf{c} = \Phi \mathbf{x} \in \mathbb{R}^N \) has only at most \( k \) entries that carry most of the information. In this case, one can rewrite the measurement system (1.2) in
the equivalent form

\[ y = A\Phi c \]  \hspace{1cm} (1.3)

and solve for \( c \) instead. When \( c \) is \( k \)-sparse, in certain cases the success in recovering \( c \) depends on only the number of measurements \( M \) and the sparsity level \( k \), while the ambient dimensionality \( N \) plays a significantly smaller role.

The second assumption about (1.2), or rather (1.3) if we consider the sparsity model or something similar, is related to the measurement matrix \( A \) that needs to be able to extract the essential information about \( x \). For example, if our signal is sparse in the Wavelet domain, then we could design \( A \) to be the Noiselet transform \[3\] and this choice of the measurement matrix can be shown to provide very high compression potential \[4\], i.e. with this combination of \( A \) and \( \Phi \), the number of measurements \( M \) required to restore \( x \) is closest to the minimum required. However, this approach requires the engineer to identify the sparsity domain \( \Phi \), which might not be possible for some practical data, to design \( A \) accordingly. A more universal result can be achieved if we refer to random matrices \( A \) that demonstrate high compression rate and stable reconstruction guarantees for arbitrary \( x \).

When \( x \) is sufficiently structured and \( A \) is designed appropriately, there are multiple frameworks for solving the considered inverse problem. A popular example would be the classical CS methods where, first, one designs a cost function that imposes both consistency with the measurements \( y \) and the existence of the assumed mathematical structure of \( x \), like sparsity in the chosen domain \( \Phi \). Then, one chooses a method for optimizing this cost function to produce an estimate of \( x \), like interior-point methods \[5, 6\] or the Iterative Soft Thresholding \[7, 8\] algorithms. Alternatively, one can refer to the probabilistic methodology and make assumptions about the distributions of the signal \( x \) and the matrix \( A \). These distributions can fuel the Bayes formula to construct estimates such as Minimum Mean Squared Error (MMSE) estimate or maximum a posteriori estimate. While these estimates possess some important features and are quite desirable, their computation in practical scenarios is either impossible or computationally infeasible. As a result, a wide range of methods has been produced for approximating those estimates in a practical manner, including the family of iterative methods called Message Passing (MP) that will be the primal focus of this thesis. No-
tably, in 2009, Donoho and co-authors gave the birth to the *Approximate Message Passing (AMP)* algorithm [9], which demonstrated spectacular reconstruction potential and opened a whole new area of research on developing MP methods for the provable, stable and informative recovery of the signals from the compressed measurements. One of the remarkable extensions of AMP was the Vector AMP (VAMP) algorithm [10] that was shown to require relatively little randomness\(^1\) from \(A\) [11, 12], is provably convergent [13], able to tackle signals with complex structures like natural images and was conjectured to be Bayes-optimal. However, VAMP has one crucial problem: at each iteration, it requires solving a *system of linear equations (SLE)* of dimension \(N\), which is an operation of complexity \(O(N^3)\) and is intractable in large-scale inverse problems. It was shown [10] that the computational cost of this operation can be substantially reduced by precomputing the *Singular Value Decomposition (SVD)* of \(A\), but this would require storing an \(N\) by \(N\) singular vector matrix, which makes this approach infeasible from the memory point of view. Taking into account the advantages of such MP algorithms, one might wonder whether there are ways of adapting those algorithms to efficiently operate in the context of large-scale inverse problems.

### 1.2 Main Contributions

In this thesis we focus on designing upscaled and optimal MP algorithms, and additional tools for these algorithms to make their practical implementation stable and efficient. The list of the contributions is the following

1. **Stable implementation of the approximated VAMP algorithm**

   One way to upscale VAMP is to approximate the solution to the SLE by \(i = O(1)\) iterations of the *Conjugate Gradient (CG)* method to obtain the *Conjugate Gradient VAMP (CG-VAMP)* algorithm. Unfortunately, there is no practical tools for making CG-VAMP stable for a wide range of regimes and first we develop rigorous theory that provides such tools.

2. **Adapted CG for CG-VAMP**

\(^1\)Only \(N\) random values are used to construct such a matrix \(A\) as opposed to \(MN\) random values required for, for example, a random Gaussian matrix.
Using CG to upscale VAMP leads to a natural question: how many of those CG iterations \(i\) to do at each iteration of CG-VAMP? We provide a strategy and practical tools to find such \(i\), and obtain a CG-VAMP variant that has a faster time-wise convergence while preserving the accuracy of the final estimate.

3. **Warm-Starting in Message Passing**

A natural consequence of approximating the exact step of VAMP by CG is that the fixed point of CG-VAMP might be substantially worse than that of VAMP. We rigorously solve this problem by introducing the *warm-starting* framework for CG and propose *Warm-Started Conjugate Gradient VAMP (WS-CG-VAMP)* whose fixed points are the fixed points of VAMP for any \(i > 0\).

4. **Divergence estimation in Message Passing**

Lastly, we consider a subproblem that arises in every AMP/VAMP-like algorithm – computing the divergence of a denoiser used within the algorithm. We propose rigorous methods that when used within such algorithms lead to almost halving the computational time of the algorithms.

### 1.3 Publications

This thesis includes the contributions from works that have previously been submitted or published in peer reviewed journals and conference proceedings. The list of the publications is the following


• N. Skuratovs and M. E. Davies, "Divergence Estimation in Message Passing algorithms," is under review for IEEE Transactions on Information Theory.

1.4 Organization of the thesis

The structure of the thesis is the following:

Chapter 2 is dedicated to an overview of the background for our work. It begins with the introduction of the classical CS approach for recovering a sparse $x$ from $y$ as in (1.2). Then the original AMP algorithm is presented, compared to the CS methods and its advantages and disadvantages are discussed. Next, we introduce the Expectation Propagation (EP) framework, discuss how it leads to the VAMP algorithm and consider its limitations. The limitations of VAMP naturally lead to the discussion about the approximated version of VAMP – CG-VAMP and what are the advantages and drawbacks of the current approach.

Chapter 3 is devoted to developing a set of tools enabling CG-VAMP to be efficient and be stable for the finite dimensional inverse problems and for a wide range of CG iterations $i$. We begin by developing rigorous and efficient methods for stabilizing CG within CG-VAMP and estimating its intrinsic variance after each inner-loop iteration $i$. Next, we use these tools in combination with the proposed stopping criteria to propose the Adapted Conjugate Gradient (ACG) method. We demonstrate that when CG-VAMP uses ACG instead of regular CG, the resulting algorithm has a faster time-wise convergence while preserving the accuracy of the final estimate.

Chapter 4 discusses the drawback of approximating the SLE by the regular CG algorithm in MP and the inherent sub-optimality of CG-VAMP. Next, we introduce the warm-starting scheme for CG to obtain the WS-CG method that allows to regain the optimality of the resulting WS-CG-VAMP while preserving scalability. Additionally, we develop the large system limit model of WS-CG-VAMP that
guarantees most of the main asymptotic properties of VAMP. Similarly to the zero-initialized CG, we propose rigorous methods for stabilizing and estimating the intrinsic variance of WS-CG. Then, we generalize the idea of WS-CG and present a unified approximating framework. Through this framework we prove the optimality of WS-CG-VAMP and show that the approximation method from the recently proposed Memory AMP (MAMP) algorithm [14] is a special case of this framework.

Chapter 5 focuses on the problem of estimating the divergence of black-box denoisers like BM3D, DnCNN etc., used within Scalable Message Passing (SMP) (AMP, VAMP and their derivatives). First, we introduce the only general method for this task — Black-Box Monte Carlo (BB-MC) — and discuss its drawbacks. Next, we exploit several key properties of SMP algorithms and propose two large system limit models of the divergence of a denoiser. Then, we discuss how to make practical estimators out of these models. As a result we obtain two estimators, and compare both of them against BB-MC in the numerical experiments.
Chapter 2

Background

Recall that in this thesis we focus on solving the compressed sensing (CS) problem of recovering a signal $x \in \mathbb{R}^N$ of dimension $N$ from the following under-determined linear system

$$y = Ax + w,$$  \hspace{1cm} (2.1)

where $y \in \mathbb{R}^M$ is of dimension $M < N$, $w \in \mathbb{R}^M$ is the measurement noise vector, and $A \in \mathbb{R}^{M \times N}$ or $A \in \mathbb{C}^{M \times N}$ (but we will mainly consider the real case) is a measurement matrix that is assumed to be available. As mentioned before, recovering a general signal of dimension $N$ from $M < N$ measurements is an ill-posed problem and to handle this fundamental limitation and make the reconstruction task feasible, we will assume that $x$ has fewer degrees of freedom than the ambient dimensionality $N$. We begin by introducing the classical CS methods for solving this kind of inverse problem and then set up the probabilistic framework for Message Passing algorithms.

2.1 Low dimensional Signals, Sampling and Algorithms

The overview of the classical CS methods concerns the following three questions:

1. What are the assumptions about $x$ for the inverse problem to be feasible?
2. How can one design a (possibly random) measurement matrix $A$ to ensure the recovery of $x$ with a high probability?

3. What are the algorithms that are able to solve the constructed feasible inverse problem?

The below discussion follows this order.

2.1.1 Sparsity and Compressibility

The main obstacle of recovering $x$ from $y$ as in (2.1) is that the number of unknowns $N$ is larger than the number of observations $M$. An intuitive way of approaching this problem is to limit the number of degrees of freedom of $x$ to reduce the search space for the estimate $\hat{x}$. One of the most popular models to impose on the signal is $k$-sparsity when we limit $x$ to have at most $k < N$ non-zero entries,

$$\|x\|_0 = \text{card}(\text{supp}(x)) \leq k, \quad (2.2)$$

where

$$\text{supp}(v) = \{ i \in [N] : v_i \neq 0 \} \quad (2.3)$$

is the support of a vector $v \in \mathbb{R}^N$ and card($S$) is the cardinality of a set $S$. For the moment, we assume that the measurements $y$ from (2.1) are noiseless so $w = 0$, but we will remove this assumption shortly. In this case, to make the reconstruction problem well-posed, we require the target vector $x$ to be the only vector from the set of all $k$-sparse vectors

$$\Omega_k = \{ z \in \mathbb{R}^N : \|z\|_0 \leq k \} \quad (2.4)$$

that satisfies $y = Ax$. Then, a natural way of recovering such a sparse signal is by solving the following optimization problem

$$\min_{z \in \mathbb{R}^N} \|z\|_0 \quad \text{subject to } Az = y, \quad (2.5)$$

i.e. find the sparsest vector $z$ that satisfies the measurements $Az = y$.

However, many examples of real-world data are not sparse in the ambient domain, but rather in the transform domain, like Fourier or Wavelet domains. Then, instead of recovering $x$ directly, we could focus on recovering its transformed
version $c$ such that

$$\mathbf{x} = \mathbf{T}c = \sum_{i=1}^{N} \mathbf{T}_{(:,i)}c_i, \quad (2.6)$$

where $\mathbf{T}$ is the *sparsifying transform* and $\mathbf{T}_{(:,i)} \in \mathbb{C}^N$ defines the $i$-th column vector of $\mathbf{T}$. For example, in Figure 2.1 (a) we display a natural image 'men' of dimension 1024 by 1024 that will be used throughout the thesis and its db4 wavelet transform $c$ is depicted in Figure 2.1 (b). As it is seen from the figure, the image is dense in the original domain, but its transformed domain is mostly represented by entries with nearly zero magnitude. In this example, the vector $c$ is not exactly sparse, but is rather *compressible*, i.e. there are some $k \ll N$ entries that carry most of the power of the signal, while the remaining entries are approximately zero. There are multiple ways of mathematically defining a compressible signal and one popular approach is via the best $k$-term approximation error $\sigma_k(c)$

$$\sigma_k(c) = \min\{\|c - c_k\|_1 : c_k \in \mathbb{R}^N, \|c\|_0 \leq k\}. \quad (2.7)$$

One can say that $c$ is compressible if $\sigma_k(c)$ rapidly decreases with $k$. For compressible signals, one could set those nearly zero values to zero and obtain a sparse approximation, i.e. approximate $\mathbf{x}$ by

$$\mathbf{x}^* = \sum_{i=1}^{k} \mathbf{T}_{(:,j_i)}c_{j_i}, \quad (2.8)$$

where $\{j_i : i \in \{1, ..., k\}\}$ are the indices that form the best $k$-term approximation of $c$. For example, in Figure 2.1 (c) we have displayed an image where 95% of the smallest magnitude-wise wavelet coefficients $c$ were set to zero and yet the approximated image $\mathbf{x}^*$ looks very similar to the original $\mathbf{x}$. This supports the belief that one might achieve a satisfactory reconstruction error by trying to recover a simpler vector $\mathbf{x}^*$ that has only $k \ll N$ degrees of freedom instead of the original signal $\mathbf{x}$. This $k$-sparse model has been generalized in multiple ways in many works, including [15, 16, 17, 18, 19, 20].
Figure 2.1: A natural image (a) and its db4 wavelet decomposition (b); (c) represents the image formed out of the top 5% magnitude-wise wavelet coefficients of the original image.

2.1.2 Sampling

The next component we need to take into account when we deal with the CS inverse problem is the measurement operator $A$. First, we consider the simpler case, where our signal $x$ is the unique solution to $y = Az$, where $z$ is $k$-sparse, and we would like to use the optimization problem (2.5) as the way to recover $x$. Then, what are the properties of $A$ that ensure the success of this approach? Here we can split the answer into two cases. In the first case, we consider the conditions that ensure the recovery of every $k$-sparse signal $x$ from $y = Ax$, while the second case concerns the conditions that ensure the recovery of a specific signal $x$. The two cases are therefore called uniform conditions and nonuniform conditions respectively. While uniform recovery is clearly the more desirable choice among the two, the respective conditions are sometimes either hard to verify or provide more pessimistic guarantees.

We briefly discuss both cases, starting with the uniform recovery. First, define an additional $k$-sparse vector $v \in \Omega_k$, where $\Omega_k$ is as in (2.4). Then, uniqueness of a sparse solution to (2.5) is ensured if $Ax \neq Av$ for all the combinations of pairs of vectors $(x, v) \in \Omega_k$. Note that this condition is equivalent to

$$A(x - v) \neq 0$$

implying that a vector $r = x - v$ must not be in the null-space of $A$. By definition,
we have that \( r \in \Omega_{2k} \), so the necessary and sufficient condition for the successful reconstruction of any \( k \)-sparse vector from its measurements \( y = Ax \) is that there are no \( 2k \)-sparse vectors in the null-space of \( A \). There are many examples of deterministic matrices that possess this property including the partial Fourier matrix, where \( A = P_k F \) with \( F \) being the Fourier matrix and \( P_k \in \mathbb{R}^{(2k) \times N} \) being the subsampling matrix that picks the first \( 2k \) rows of \( F \) [21].

The uniform reconstruction results can be further improved at the expense of the universality. It was shown [21] that the set of matrices \( A \) with \( k + 1 \) rows that would lead to an incorrect reconstruction of a particular \( x \) has a Lebesgue measure zero. Thus, given a \( k \)-sparse vector, it can be successfully recovered from measurements \( y = Ax \) for almost every \( A \in \mathbb{C}^{(k+1) \times N} \).

The above discussion was centered around the reconstruction of a signal \( x \) by solving the optimization problem (2.5), which is an NP-hard problem [21] and thus is not feasible for even mildly large inverse problems. Not only that, but also we have assumed that \( x \) is exactly sparse, which is rarely the case in practice, and the measurements \( y \) are noiseless. A more realistic example would be a measurement system

\[
\mathbf{y} = \mathbf{A} \mathbf{x} + \mathbf{w},
\]

where \( \mathbf{w} \in \mathbb{R}^M \) is a zero-mean independent and identically distributed (i.i.d.) Gaussian noise vector \( \mathbf{w} \sim \mathcal{N}(0, \sigma^2 \mathbf{I}_M) \) and \( \mathbf{x} \) is compressible. In this context, a more practical and robust alternative to (2.5) would be the so-called Basis Pursuit Denoising (BPDN) [6], which reads as

\[
\min_{\mathbf{z} \in \mathbb{R}^N} \|\mathbf{z}\|_1 \quad \text{subject to } \|\mathbf{y} - \mathbf{A} \mathbf{z}\| \leq \eta,
\]

(2.11)

where we “convexified” the \( l_0 \)-norm \( \|\cdot\|_0 \) and instead of \( \mathbf{A} \mathbf{z} = \mathbf{y} \) we used a relaxed constraint \( \|\mathbf{y} - \mathbf{A} \mathbf{z}\| \leq \eta \) with \( \eta \) controlling the magnitude of the noise in \( \mathbf{y} \). Alternatively, one can reformulate (2.11) as an unconstrained problem to obtain the Least Absolute Shrinkage and Selection Operator (LASSO) [22]

\[
\min_{\mathbf{z} \in \mathbb{R}^N} \|\mathbf{y} - \mathbf{A} \mathbf{z}\| + \lambda \|\mathbf{z}\|_1.
\]

(2.12)

One can show that the two optimization problems yield the same solution when the parameters \( \eta \) and \( \lambda \) are chosen in a certain way [21]. Importantly, there is
a wide range of practical methods that can approximately or even exactly solve the two optimization problems given $A$ satisfies certain different conditions for uniform and nonuniform reconstruction.

The uniform case is usually considered in the light of the Restricted Isometry Property (RIP). We say that a matrix $A \in \mathbb{R}^{M \times N}$ follows RIP of order $k$ if there is a constant $0 < \delta_k < 1$ such that \cite{4, 21}

$$\left(1 - \delta_k\right)\|z\| \leq \|Az\| \leq \left(1 + \delta_k\right)\|z\|$$

(2.13)

for all $k$-sparse vectors $z \in \mathbb{R}^N$. This property implies that if we take any $k$ columns of $A$, then the corresponding Gram matrix would be well-conditioned. With the RIP in hand, one can establish the following result

**Theorem 1.** \cite[Theorem 6.11]{21} Suppose that $A \in \mathbb{R}^{M \times N}$ follows the RIP with the constant $\delta_{2k} < 0.4931$. Then the magnitude of the error of the solution $\hat{x}$ to (2.11) is bounded as

$$\|x - \hat{x}\| \leq \frac{C}{\sqrt{k}}\sigma_k(x) + D\eta,$$

(2.14)

where $\sigma_k(x)$ is the error of the best $k$-term approximation from (2.7) and constants $C, D > 0$ depend only on $\delta_{2k}$.

We note that a similar error result to (2.14) can be established for a sharp bound $\delta_{2k} < 1/\sqrt{2}$, see Theorem 2.1 in \cite{23}. The results of Theorem 1 suggest that as long as $A$ satisfies the required RIP condition, the error of the solution to (2.11) is proportional to the error of the best $k$-term approximation, as if we had the oracle information about the support of the largest magnitude-wise entries of $x$, plus a term controlling the noise level $\eta$.

The next goal is to figure out what matrices and how many rows of this matrix, i.e. the number of measurements $M$, ensure the required RIP condition. The first examples would be the class of randomly sampled orthonormal matrices

$$A = P_M U,$$

(2.15)

where $P_M \in \mathbb{R}^{M \times N}$ is a random subsampling matrix that randomly picks $M$ rows of the matrix $U \in \mathbb{R}^{N \times N}$, which in turns is an orthonormal matrix that has a low
coherence value $\mu(U)$ that is defined as [4, 24, 25]

$$\mu(U) = \max_{i,j \leq N} |U_{i,j}|^2 \in [1/N, 1].$$

(2.16)

When $A$ is designed as in (2.15), it is sufficient to have

$$M > CN\mu(U)\delta^{-2}k\ln^4(N),$$

(2.17)

measurements, where $C > 0$ is a constant, to ensure the RIP of $A$ of order $k$ with a constant $\delta_k \leq \delta$ with an overwhelming probability [21]. There are multiple examples of orthonormal matrices $U$ that have (nearly) minimal coherence $\mu(U)$, including the transforms like Fourier, Wavelet, Hadamard and many other types [21, 26]. For example, when $U$ is a Fourier matrix, which has the minimal coherence $\mu(U) = 1/N$, (2.17) simplifies to $M > C\delta^{-2}k\ln^4(N)$, which scales linearly with $k$ and poly-logarithmically with $N$. However, as pointed out before, the signal $x$ is usually sparse or compressible in some other domain $T$. In this case, in order to successfully recover a sparse vector $c$ such that $x = Tc$, we need to incorporate $T$ into $A$

$$A = P_M U T = P_M \tilde{U},$$

(2.18)

where $\tilde{U} = UT$ is the new orthonormal matrix and has the coherence

$$\mu(\tilde{U}) = \mu(U, T) = \max_{i,j \leq N} |U_{i,j}^T T_{(i,j)}|.$$

(2.19)

Thus, when we have a sparsifying transform $T$, we need to design $U$ to have minimal coherence with $T$ to reduce the number of measurements $M$. Unfortunately, in many practical scenarios this is hard to accomplish. For example, in MRI, $U$ can be modeled as the Fourier matrix which has maximum coherence $\mu(U, T) = 1$ with the popular sparsifying Wavelet transform $T$ [16, 27], leading to no compression $M = N$. There are different possible ways to tackle this problem including the following two. First, instead of the uniform sampling, where $P_M$ picks a row from $\tilde{U}$ with an equal probability, to use nonuniform sampling strategy. In this case, we can study the interaction between the columns of $U$ and the columns of $T$ for each specific pair $(U, T)$ and value certain rows of $\tilde{U}$ more than others and, therefore, sample them with a higher chance. This idea was rigorously studied in [27], where the authors showed that nonuniform sampling can lead to nearly optimal number
of measurements even for combinations like Fourier $\mathbf{U}$ and Wavelet $\mathbf{T}$.

Alternatively, one can refer to randomness and choose $\mathbf{U}$ to be, for example, a Gaussian matrix, whose entries are i.i.d standard normal values, or a random binary matrix, whose entries are independent and take values either $+1$ or $-1$ with an equal probability. When $\mathbf{A}$ is designed as in (2.15) with $\mathbf{U}$ being a Gaussian matrix and $\mathbf{P}_M$ being a uniform subsampling matrix, $\mathbf{A}$ satisfies RIP with a constant $\delta_k \leq \delta$ with an overwhelming probability when the number of measurements $M$ follows

$$M \geq C\delta^{-2}k \ln(eN/k), \quad (2.20)$$

which is the optimal bound [21]. Comparing (2.20) to (2.17) we note that a Gaussian matrix provides a much better bound on the sufficient number of measurements. Another important advantage of a Gaussian measurement operator is the universality with respect to the sparsifying transform $\mathbf{T}$. By means of random matrix theory, one can show that the bound (2.20) is invariant to the right-multiplication of a Gaussian $\mathbf{U}$ by an orthonormal $\mathbf{T}$ [21]. As a result, by using a random Gaussian measurement matrix, the user can be flexible with the choice of the sparsifying domain.

Lastly, we mention the nonuniform recovery results. Similarly to the uniform case, one can design $\mathbf{A}$ as in (2.15) with $\mathbf{U}$ being some (deterministic) orthogonal matrix and choose only $\mu(\mathbf{U}) \delta^{-2}k \ln^2(N)$ (2.21) measurements to ensure an error bound for $\hat{\mathbf{x}}$ similar to (2.14) by solving (2.11) with an overwhelming probability. Note that uniform recovery built on RIP requires $M$ to be by a few log factors greater than required for the nonuniform case. Similarly, when we choose $\mathbf{U}$ to be a Gaussian matrix, the number of measurements leading to stable reconstruction of a fixed $\mathbf{x}$ reduces to [21]

$$M \geq Ck \ln(eN/k), \quad (2.22)$$

which is actually only by a constant factor better than in the uniform case.
### 2.1.3 Reconstruction algorithms for sparse signals

Once we have figured out examples of compressible signals $\mathbf{x}$ and measurements matrices that create the context for provable recovery of $\mathbf{x}$, we need to work out an algorithm that would successfully accomplish the recovery. While there are many examples of practical algorithms for this task, in this thesis we point out a few classical ones. The first family of algorithms, usually referred to as greedy methods, includes the Compressive Sampling Matching Pursuit (CoSaMP) [28], which involves two functions [21]: the hard thresholding operator $H_k(\mathbf{z})$ that chooses $k$ entries of $\mathbf{z}$ with the largest magnitude and sets the rest of the entries to zero, and a function $L_k(\mathbf{z})$ that returns the support of these $k$ largest entries. Then, CoSaMP takes the following form

$$ S^{t+1} = \text{supp}(\hat{\mathbf{x}}_t) \cup L_2k(A^T(y - A\hat{\mathbf{x}}_t)) $$

$$ \mathbf{u}_{t+1} = \arg\min_{\mathbf{z}} \{ \|y - A\mathbf{z}\|, \text{supp}(\mathbf{z}) \subseteq S^{t+1} \} $$

$$ \hat{\mathbf{x}}_{t+1} = H_k(\mathbf{u}_{t+1}). $$

Another popular example of a greedy algorithm is the Iterative Hard Thresholding (IHT) [29] that reads as

$$ \hat{\mathbf{x}}_{t+1} = H_k(\hat{\mathbf{x}}_t + A^T(y - A\hat{\mathbf{x}}_t)). $$

Both CoSaMP and IHT can be rigorously analyzed under the RIP assumption on $A$. In particular, the following error bound exists for CoSaMP given $\delta_{8k} < 0.4782$ [21]

$$ \|\mathbf{x} - \hat{\mathbf{x}}_t\|_1 \leq \sigma_k(\mathbf{x}) + D\sqrt{k}\|\mathbf{w}\| + 2(\rho)^t\sqrt{k}\|\mathbf{x}\|, $$

where $\sigma_k(\mathbf{x})$ is defined in (2.7), $\mathbf{w}$ is the noise vector in $\mathbf{y} = A\mathbf{x} + \mathbf{w}$, and $C, D$ and $0 < \rho < 1$ are positive constants. The same error bound (2.27) (although with a different set of constants $C, D, \rho$) can be shown for the IHT given $\delta_{6k} < 1/\sqrt{3}$ [21]. Note that if $\mathbf{x}$ is exactly $k$ sparse and $\|\mathbf{w}\| = 0$, then both of the algorithms are guaranteed to converge in the limit $t \to \infty$. Importantly, the two algorithms have a linear convergence rate and are guaranteed to converge under the RIP conditions mentioned [28, 29]. Lastly, we point out the computational complexity of the two methods. Clearly, IHT is cheaper compared to CoSaMP, since it involves only
matrix-vector multiplications and the hard-thresholding operation. The latter operation can be solved in $O(N \log N)$ operations using a sorting algorithm like QuickSort, while the former can be performed in $O(N \log N)$ if $A$ is a fast operator like the Fourier transform. In the CoSaMP case, one additionally needs to solve the optimization (2.24), which is equivalent to computing the pseudo-inverse $u_{t+1} = A_{S_{t+1}}^d y = (A_{S_{t+1}}^T A_{S_{t+1}})^{-1} A_{S_{t+1}}^T y,$ (2.28)

where $A_{S_{t+1}}$ is a matrix whose columns correspond to the columns of $A$ indexed by the set $S_{t+1}$. By referring to methods like Conjugate Gradient (discussed later in the thesis), the complexity of this operation is $O(\text{card}(S_{t+1}) N \log N)$ given $A$ is a fast operator.

Another way of recovering $x$ from $y$ from (2.37) is provided by convex relaxation methods that aim to minimize the $l_1$ cost function (2.11)

$$
\min_z \left\{ F(z) = \|y - Az\|^2 + \lambda \|z\|_1 \right\}. \quad (2.29)
$$

The classical example is the Iterative Soft Thresholding Algorithm (ISTA), which is identical to IHT, but with $H_k$ replaced by the soft thresholding function

$$
\eta_\lambda(x) = \begin{cases} 
    x - \lambda & \text{if } x > \lambda \\
    0 & \text{if } -\lambda \leq x \leq \lambda \\
    x + \lambda & \text{if } x < -\lambda,
\end{cases} \quad (2.30)
$$

which operates point-wise and where $\lambda$ is a thresholding value. ISTA reads as [7]

$$
\hat{x}_{t+1} = \eta_\lambda(\hat{x}_t + A^T(y - A\hat{x}_t)), \quad (2.31)
$$

which is initialized with $\hat{x}^0 = 0$. Due to being a thresholded variant of the Landweber iteration$^2$, the convergence of ISTA to a local minima of (2.29) is ensured when $\|A\| < 1$ [7], which can always be achieved by rescaling $y$ and $A$. An accelerated version of ISTA, called Fast ISTA (FISTA) [8], uses an additional step where a very specific combination of $\hat{x}_t$ and $\hat{x}_{t-1}$ is computed. FISTA can be formulated

---

$^1$For a fast operator of dimension $N$, the matrix-vector multiplication has the complexity $O(N \log N)$ or smaller.

$^2$Landweber iteration is an iterative process of the form $\hat{x}_{t+1} = \hat{x}_t + A^T(y - A\hat{x}_t)$ [21, 30].
as
\[
\begin{align*}
\mathbf{u}_{t+1} &= \hat{x}_t + \frac{\alpha_{t-1} - 1}{\alpha_t}(\hat{x}_t - \hat{x}_{t-1}) \\
\hat{x}_{t+1} &= \eta \lambda (\mathbf{u}_t + \mathbf{A}^T(\mathbf{y} - \mathbf{A}\mathbf{u}_t)) \\
\alpha_{t+1} &= 1 + \frac{\sqrt{1 + 4(\alpha_t)^2}}{2},
\end{align*}
\]
which is initialized with \(\alpha_0 = \alpha_1 = 1\) and \(\hat{x}^0 = 0\). The difference in the computational complexity between ISTA and FISTA is negligible, but the latter turns out to converge faster to a minimizer \(x^*\) of (2.29). In particular, we have
\[
F(\hat{x}_{t}^{ISTA}) - F(x^*) = O(1/t) \quad (2.35)
\]
\[
F(\hat{x}_{t}^{FISTA}) - F(x^*) = O(1/t^2), \quad (2.36)
\]
where \(\hat{x}_{t}^{ISTA}\) and \(\hat{x}_{t}^{FISTA}\) are the vectors produced at iteration \(t\) by ISTA and FISTA respectively. However, there are regimes where both ISTA and FISTA are close to the fixed point and ISTA has a rigorously better rate \([22]\) so one might want to switch the algorithms at some point. Lastly, the computational complexity of both ISTA and FISTA is dominated by matrix-vector products and applying the soft-thresholding function, which is of \(O(N)\) operation as it does not involve sorting. As a result, both of the algorithms are of complexity \(O(N \log N)\).

The above examples of algorithms trying to compute a sparse estimate \(\hat{x}\) of \(x\) from measurements \(\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{w}\) are classical in the CS literature and many extensions have been proposed to improve convergence speed and stability. In particular, multiple acceleration methods has been proposed for IST and IHT algorithms \([31, 32, 33]\). Additionally, more advanced methods for directly solving the optimization problem (2.29) have been developed, including Homotopy Methods \([34, 35]\) that have provable error bounds under the RIP condition \([36]\), Primal-Dual and interior point methods \([5, 37, 38, 39]\). Unfortunately, these methods do not always scale up to large inverse problems, which is the primal target of this thesis. As an alternative route, one could use algorithms like ISTA and IHT as the bases for a neural network and learn certain parameters of the algorithm from the data, to obtain the Learned ISTA (LISTA) \([40, 41]\) and Learned IHT (LIHT) \([42]\) respectively.

In this thesis we choose a different path for extending the classical CS reconstruction methods – the probabilistic path. In particular, we will focus on the
Bayesian reconstruction and the associated Message Passing algorithms that are discussed next.

2.2 Message Passing for CS

As an alternative to the classical CS methods discussed above, one can refer to the probabilistic approach of constructing an estimate of a signal $x$ observed through the noisy linear system (2.1)

$$\mathbf{y} = \mathbf{A}x + \mathbf{w}, \quad (2.37)$$

where $\mathbf{w} \in \mathbb{R}^M \sim \mathcal{N}(0, v_w \mathbf{I}_M)$. One of the standard probabilistic frameworks for doing such an inference is provided by the Bayes formula for constructing the \textit{a posteriori} density

$$p(x|\mathbf{y}) = \frac{1}{Z} p(\mathbf{y}|x)p(x), \quad (2.38)$$

where $Z = \int p(x, \mathbf{y})d\mathbf{x}$ is the normalization scalar. Here, the function $p(x)$ represents the prior density, the statistical knowledge about $x$, while $p(\mathbf{y}|x)$ is the likelihood of observing $\mathbf{y}$ given the signal $x$. Based on the model (2.37), the latter is equivalent to

$$p(\mathbf{y}|x) = \prod_{j=1}^{M} \frac{1}{\sqrt{2\pi v_w}} e^{-\frac{1}{2v_w} (y_j - \sum_k A_{j,k}x_k)^2}. \quad (2.39)$$

Provided the posterior density, one could construct, for example, the \textit{Bayes-optimal} estimate $\hat{x}$ that minimizes the Mean Squared Error (MSE) $\mathbb{E}\left[ \sum_i (x_i - \hat{x}_i)^2 | \mathbf{y} \right]$. Expanding the sum and rearranging the terms gives

$$\mathbb{E}\left[ \sum_i (x_i - \hat{x}_i)^2 | \mathbf{y} \right] = \sum_i \mathbb{E}[x_i^2 | \mathbf{y}] - 2 \sum_i \mathbb{E}[x_i | \mathbf{y}] \hat{x}_i + \sum_i \hat{x}_i^2. \quad (2.40)$$

Taking the derivative with respect to $\hat{x}_i$ and equating it to zero gives the MMSE estimator of $x_i$,

$$\hat{x}_i^{\text{MMSE}} = \int_{\mathbb{R}^N} x_i p_i(x_i | \mathbf{y}) dx_i, \quad (2.41)$$
Background

where $p_i(x_i)$ is the marginal posterior density with respect to the component $x_i$

$$p_i(x_i|y) = \int_{x\backslash i} p(x|y)d\underline{x},$$

(2.42)

where the notation $x\backslash i$ means all the entries of $x$ but $i$-th. Unfortunately, for a general $A$ that is not extremely sparse, doing such an integration over an $N$-dimensional space is computationally infeasible. Yet, obtaining an estimate that minimizes the MSE is quite desirable and next we discuss how Approximate Message Passing (AMP) can solve this computational problem at the cost of standard first-order methods like ISTA or IHT, provided $A$ is sufficiently random.

2.2.1 Belief propagation as an instance of Expectation Propagation

AMP is an iterative method that aims to construct an approximation of the Minimum MSE (MMSE) estimate (2.41) and was originally proposed by Donoho and co-authors in [9]. As the name suggests, AMP is an approximated version of a particular message passing algorithm called Belief Propagation (BP) or the Sum-Product algorithm [43, 44]. On a general level, the goal of BP is to marginalize a density, $f(x_1, x_2, ..., x_N) = f(x)$, with respect to a variable $x_i$, $i \in (1, ..., N)$, i.e. find $f_i(x_i)$ such that:

$$f_i(x_i) = \int_{x\backslash i} f(x)dx.$$  

(2.43)

To make this problem tractable, it is assumed that the density $f(x)$ is a product of $M$ factors $f_j(x_j)$, where $x_j$ is a subset of entries $x$,

$$f(x) = \prod_{j=1}^{M} f_j(x_j).$$

(2.44)

For example, the posterior density (2.38) in the original form involves just two (unnormalized) factors $f_1(x) = p(y|x)$ and $f_2(x) = p(x)$ so that the vectors $x_1 = x_2 = x$ corresponds to the whole signal vector $x$.

To unify the discussion in this thesis, we will use the Expectation Propagation (EP) framework [45] to define the BP algorithm, since EP will also be used in the following sections as the basis for other variants of Message Passing algorithms.
In every EP algorithm, the density \( \tilde{f}(\mathbf{x}) \) approximating (2.44) follows the same structure as the exact density \( f(\mathbf{x}) \),

\[
\tilde{f}(\mathbf{x}) = \prod_{j=1}^{M} \tilde{f}_j(\mathbf{x}_j),
\]

where \( \tilde{f}_j(\mathbf{x}_j) \) approximates \( f_j(\mathbf{x}_j) \) and is constrained to be from a chosen class of densities, like the Gaussian distribution. To produce the approximated density \( \tilde{f}(\mathbf{x}) \), the EP algorithm iteratively refines each \( \tilde{f}_j(\mathbf{x}_j) \) one by one and tries to achieve a local expectation consistency between the approximated factors \( \{\tilde{f}_j(\mathbf{x}_j)\}_{j=1}^{M} \) and the exact ones \( \{f_j(\mathbf{x}_j)\}_{j=1}^{M} \). This refinement process starts with picking a factor to update, \( \tilde{f}_k(\mathbf{x}_k) \), and removing it from the joint density (2.45) to obtain \( \tilde{f}_{\backslash k}(\mathbf{x}) = \tilde{f}(\mathbf{x})/\tilde{f}_k(\mathbf{x}_k) \). Next, EP generates a refined version of \( \tilde{f}_{\text{new}}(\mathbf{x}_k) \) that belongs to the chosen class of densities and ensures that the densities \( \tilde{f}_{\text{new}}(\mathbf{x}_k) \tilde{f}_k(\mathbf{x}) \) and \( f_k(\mathbf{x}_k) \tilde{f}_{\backslash k}(\mathbf{x}) \) match in terms of expectation of a chosen set of functions \( \{\phi_r(\mathbf{x})\}_{r=1}^{r_{\text{max}}} \),

\[
E[\{\phi_r(\mathbf{x})\}_{r=1}^{r_{\text{max}}} | \mathbf{x} \sim \tilde{f}_{\text{new}}(\mathbf{x}_k) \tilde{f}_k(\mathbf{x})] = E[\phi_r(\mathbf{x}) | \mathbf{x} \sim f_k(\mathbf{x}_k) \tilde{f}_{\backslash k}(\mathbf{x})].
\]

This is motivated by the fact that usually we are not interested in the exact density \( f(\mathbf{x}) \) but rather in some statistical information about it, like the mean, so that \( \phi_r(\mathbf{x}) = \mathbf{x} \), or the covariance, so that \( \phi_r(\mathbf{x}) = \mathbf{x}\mathbf{x}^T \). By refining each factor \( \tilde{f}_j(\mathbf{x}_j) \), \( j = 1, ..., M \) in this way a sufficient number of cycles, we hope to achieve the expectation consistent condition [46]

\[
E[\{\phi_r(\mathbf{x})\}_{r=1}^{r_{\text{max}}} | \mathbf{x} \sim \tilde{f}(\mathbf{x})] = E[\{\phi_r(\mathbf{x})\}_{r=1}^{r_{\text{max}}} | \mathbf{x} \sim f_1(\mathbf{x}_1) \tilde{f}_1(\mathbf{x})]
= E[\{\phi_r(\mathbf{x})\}_{r=1}^{r_{\text{max}}} | \mathbf{x} \sim f_2(\mathbf{x}_2) \tilde{f}_2(\mathbf{x})]
= ... 
= E[\{\phi_r(\mathbf{x})\}_{r=1}^{r_{\text{max}}} | \mathbf{x} \sim f_M(\mathbf{x}_M) \tilde{f}_{\backslash M}(\mathbf{x})].
\]

Then, the resulting expectation \( E[\{\phi_r(\mathbf{x})\}_{r=1}^{r_{\text{max}}} | \mathbf{x} \sim \tilde{f}(\mathbf{x})] \) is used as the approximation of the exact expectation \( E[\{\phi_r(\mathbf{x})\}_{r=1}^{r_{\text{max}}} | \mathbf{x} \sim f(\mathbf{x})] \). Here it is important to emphasize that while computing the expectation as in (2.46) might be impossible for the exact density \( f(\mathbf{x}) \) from (2.44), EP needs to evaluate those expectations for \( f_k(\mathbf{x}_k) \tilde{f}_{\backslash k}(\mathbf{x}) \), which involves only one exact factor and might be much more tractable.
To define BP through EP, for each \( v \in \mathbb{R} \) let the function \( \phi_{k,v}(x) \) be as
\[
\phi_{k,v}(x) = \delta(x_k - v),
\]
where \( \delta(\cdot) \) is the Dirac delta function. One can verify that when a density \( d(x) \) and a density \( h(x) \) follow
\[
\mathbb{E}[\phi_{k,v}(x)|x \sim d(x)] = \mathbb{E}[\phi_{k,v}(x)|x \sim h(x)]
\]
for \( k = 1, \ldots, N \) and \( \forall v \),
then it is equivalent to say that their marginal densities
\[
d_k(x_k) = \int_{x \backslash k} d(x)dx_{x \backslash k}
\]
\[
h_k(x_k) = \int_{x \backslash k} h(x)dx_{x \backslash k}
\]
are equivalent \[47\]
\[
d_k(x_k) = h_k(x_k).
\]

One can formalize the above formulation of BP in terms of EP. For this, first, every approximated factor \( \tilde{f}_j(x) = 1, j = 1, \ldots, M \) is initialized with 1. Then, at every iteration we pick a factor to refine, \( \tilde{f}_k(x_k) \), and compute the so-called \textit{cavity density} by removing \( \tilde{f}_k(x_k) \) from \( \tilde{f}(x) \),
\[
\tilde{f}_{\backslash k}(x) = \frac{\tilde{f}(x)}{\tilde{f}_k(x_k)}.
\]
Next, we compute the \textit{tilted density} by adding the true factor \( f_k(x_k) \) to the cavity density and normalizing,
\[
\tilde{q}(x) = \frac{\tilde{f}_{\backslash k}(x)f_k(x_k)}{\int \tilde{f}_{\backslash k}(x)f_k(x_k)dx}.
\]
After that, we compute the new approximated joint density \( \tilde{f}_{\text{new}}(x) \) by minimizing the \textit{Kullback-Leibler (KL) divergence} \[48\]
\[
D_{KL}(\tilde{q}(x)||\tilde{f}_{\text{new}}(x)) = \int x \tilde{q}(x) \log \left( \frac{\tilde{f}_{\text{new}}(x)}{\tilde{q}(x)} \right) dx
\]
under the constraint that \( \tilde{f}^{\text{new}}(\mathbf{x}) \) factorizes as

\[
\tilde{f}^{\text{new}}(\mathbf{x}) = \prod_{i=1}^{N} \tilde{q}_i(x_i). \tag{2.56}
\]

This optimization problem is equivalent to enforcing the matching condition (2.49) where \( d(\mathbf{x}) = \tilde{f}^{\text{new}}(\mathbf{x}) \) and \( h(\mathbf{x}) = \tilde{q}(\mathbf{x}) \) [47]. Therefore, we have that the update for every \( \tilde{q}_i(x_i) \) follows

\[
\tilde{q}_i(x_i) = \int_{\mathbf{x}\setminus i} \tilde{q}(\mathbf{x}) d\mathbf{x}\setminus i. \tag{2.57}
\]

Lastly, the refined factor \( \tilde{f}_k^{\text{new}}(\mathbf{x}_k) \) is formed as

\[
\tilde{f}_k^{\text{new}}(\mathbf{x}) = \frac{\tilde{f}^{\text{new}}(\mathbf{x})}{f_k(\mathbf{x})}. \tag{2.58}
\]

The above algorithm is iterated until the convergence to a fixed point (which always exists if every factor \( f_j(\mathbf{x}_j) \) in (2.44) is positive [49]) where the expectation consistency condition (2.47) is achieved.

### 2.2.2 Approximate Message Passing

The above BP algorithm can be applied to approximate the marginal posterior density (2.42) in the CS context. For this, we follow the route from [9] and assume that the prior \( p(\mathbf{x}) \) of the signal \( \mathbf{x} \) is separable,

\[
p(\mathbf{x}) = \prod_{i=1}^{N} p(x_i). \tag{2.59}
\]

In this case, the posterior density \( p(\mathbf{x}|\mathbf{y}) \) from (2.38) takes the following form

\[
p(\mathbf{x}|\mathbf{y}) = \frac{1}{N} p(\mathbf{y}|\mathbf{x}) p(\mathbf{x}) = \frac{1}{Z} \prod_{i=1}^{N} p(x_i) \prod_{j=1}^{M} \frac{1}{\sqrt{2\pi v_w}} e^{-\frac{1}{2v_w}(y_j - \sum_k A_{j,k} x_k)^2}, \tag{2.60}
\]

where we used the likelihood \( p(\mathbf{y}|\mathbf{x}) \) from (2.39). If we let \( f_j(\mathbf{x}) \) to be

\[
f_j(\mathbf{x}) = e^{-\frac{1}{2v_w}(y_j - \sum_k A_{j,k} x_k)^2}, \tag{2.61}
\]

then we can apply the above BP algorithm to marginalize (2.60). However, if we do so and follow the steps (2.53)-(2.58), we will face two major problems. First,
one can show [50, 51] that because the factor (2.61) is a function of every entry of $x$, the integration operation (2.57) cannot be simplified and every iteration of BP involves integration over $N-1$ dimensional space, which might be as hard as directly computing the marginal posterior density (2.42). Second, every iteration of BP requires a refinement of $M$ factors and each of these refinements requires performing $N$ marginalization operations. Even if one could approximate the marginalization by some cheap operation, it would still require performing $NM$ of such operations at each iteration of BP, which becomes intractable in the context of large scale inverse problems.

To upscale the above BP algorithm, the authors of AMP [9] constrained $A$ to be a random Gaussian matrix and considered the algorithm in the large system limit, where both $N$ and $M$ approach infinity at the same rate, $M = \delta N \to \infty$, $\delta = O(1)$. Under these conditions and performing a series of careful approximations, one can solve the above two problems and reformulate BP as an iterative first order method that updates a mean vector and a scalar variance (for a more detailed derivation of these steps, please refer to [50, 51]). This algorithm was called Approximate Message Passing and it is shown in Algorithm 1 [9, 52].

**Algorithm 1: Approximate Message Passing**

**Initialization:** $x_0 = 0$, $z_{-1} = 0$, $\alpha_{-1} = 0$, $t = 0$

1. while $t < t_{\text{max}}$ do
2. \[ z_t = y - A\hat{x}_t + \frac{N}{M} \alpha_{t-1} z_{t-1} \]
3. \[ r_t = \hat{x}_t + A^T z_t \]
4. \[ \hat{v}_{h_t} = \frac{1}{\sqrt{N}} ||r_t||^2 \]
5. \[ \hat{x}_{t+1} = g(r_t, \hat{v}_{h_t}) \]
6. \[ \alpha_{t+1} = \nabla_{r_t} \cdot g(r_t, \hat{v}_{h_t}) \]

**Output:** $\hat{x}_{t+1}$

Based on the derivation of AMP from BP, the function $g$ represents the conditional mean

$$g(r, v) = \mathbb{E}[x|r = x + \sqrt{v}h, h \sim \mathcal{N}(0, I), x \sim p(x)], \quad (2.62)$$

while the scalar $\alpha_t$ is the divergence of this function with respect to $r_t$

$$\alpha_{t+1} = \nabla_x \cdot g(x, \hat{v}_{h_t}) = \frac{1}{N} \sum_{i=1}^{N} \frac{\partial (g(x, \hat{v}_{h_t}))_{i}}{\partial x_i}, \quad (2.63)$$
As (2.62) suggests, we assume that at every iteration $t$, the input $r_t$ to the estimator $g$ acts as the true signal $x$ corrupted by independent zero-mean Gaussian noise $h_t$,

$$r_t = x + h_t \quad h_t \sim \mathcal{N}(0, v_{ht} I),$$

(2.64)

where $v_{ht}$ is the oracle variance of the error $h_t$. Thus, the scalar $\hat{v}_{ht}$ in Algorithm 1 is an estimate of $v_{ht}$. Finally, the term $\frac{N}{M} \alpha_{t-1} z_{t-1}$ in the computation of $z_{t+1}$ is called the Onsager correction term whose role is discussed next.

### 2.2.3 Rigorous dynamics of AMP

While the methods proposed in the original work [53] and an alternative approach from [50, 51] allowed to derive AMP from BP, these works did not provide a rigorous proof of the error dynamics in the algorithm. For example, for IHT, CoSaMP and many other CS algorithms, there are error bounds on estimates produced by these algorithms after each iteration (see Section 2.1.3). Such an analysis for AMP was carried out in the work by Bayati and Montanari in [54], where it was shown that one can rigorously define a 1D recursion called State Evolution (SE) that exactly predicts the MSE of the estimates produced by the algorithm, and confirmed the assumed model (2.64) under the following three assumptions.

**Assumption A1.** The dimensions of the signal model $N$ and $M$ approach infinity with a fixed ratio $\delta = \frac{M}{N} = O(1)$.

This assumption formalizes the fact that most of the analysis will be carried out in the Large System Limit (LSL) regime, where we expect a certain concentration phenomena to hold. Note that even though in this thesis we focus on the Compressed Sensing scenario $0 < \delta < 1$, the theory mentioned and proposed in the following holds for the case $\delta > 1$ as well [55, 56].

**Assumption A2.** The measurement matrix $A$ is orthogonally invariant, such that in the SVD of $A = USV^T$, the matrices $U$ and $V$ are independent of other random terms and are uniformly distributed on the set of orthogonal matrices. Additionally, the matrix $S^T S$ has the limiting eigenvalue distribution with the first
moments equal to the first $2t$ moments of the Marčenko-Pastur distribution \[57\], where $t$ is the maximum number of iterations of AMP.

The main candidate that follows this assumption is the class of matrices with i.i.d. entries $A_{i,j}$ with a zero mean $E[A_{i,j}] = 0$ and a finite variance $E[A_{i,j}^2] < \infty$ \[57\], which we will stick to in the context of AMP. The next assumption is a technical requirement that ensures certain inner-products converge in the LSL.

**Assumption A3.** The denoiser $g$ is uniformly Lipschitz so that the sequence of functions $g: \mathbb{R}^N \rightarrow \mathbb{R}^N$ indexed by $N$ are Lipschitz continuous with a Lipschitz constant $L_N < \infty$ as $N \rightarrow \infty$ \[58\], \[56\]. Additionally, we assume the following limits exist almost surely \[58\]

\[
\lim_{N \rightarrow \infty} \frac{1}{N} g(x + d_1)^T g(x + d_2), \quad \lim_{N \rightarrow \infty} \frac{1}{N} x^T g(x + d_1),
\]

\[
\lim_{N \rightarrow \infty} \frac{1}{N} d_1^T g(x + d_2), \quad \lim_{N \rightarrow \infty} \frac{1}{N} x^T d_1, \quad \lim_{N \rightarrow \infty} \frac{1}{N} \|x\|^2
\]

and

\[
\lim_{N \rightarrow \infty} \frac{1}{N} \nabla d_1 \cdot g(x + d_1) = \lim_{N \rightarrow \infty} \frac{1}{(C)_{1,2}} \frac{1}{N} d_1^T g(x + d_1).
\] (2.65)

Here $d_1, d_2 \in \mathbb{R}^N$ with $((d_1)_n, (d_2)_n) \sim \mathcal{N}(0, C)$ for some positive definite $C \in \mathbb{R}^2$ and $\frac{1}{N} \nabla d_1 \cdot g(x + d_1)$ defines the divergence of $g$ with respect to $d_1$,

\[
\frac{1}{N} \nabla d_1 \cdot g(x + d_1) = \frac{1}{N} \sum_{i=1}^{N} \frac{\partial(g(x + d_1))_i}{\partial(d_1)_i}
\] (2.66)

Note that the limit (2.65) is a standard result due to the Stein’s lemma \[59\] (See Lemma 6 in Appendix A). Additionally, we would like to mention that in the original work \[54\], which rigorously analysed the dynamics of AMP, the authors used a stronger assumption than A3: it was assumed that the denoiser $g$ is separable, $(g(x))_i = g_i(x_i)$, and the function $g_i$ is Lipschitz continuous. Later, this assumption was relaxed in \[56\] to non-separable uniformly Lipschitz continuous functions $g$ as in Assumption A3. Therefore, next we will use the more general Assumption A3. The same assumption will be used throughout the thesis, although with different notations that will be introduced shortly.

Lastly, Assumption A1 will be used throughout the thesis, while Assumption A2 is defined specifically for AMP. This assumption will be relaxed when we get
to analysing more general Message Passing algorithms.

**Onsager Correction in AMP**

In [54], the authors analyzed the error propagation in AMP under Assumptions A1-A3 and showed the model (2.64) holds and the MSE of the AMP’s estimates can be rigorously defined if the *Onsager correction* term \( \frac{N}{M} \alpha_{t-1} z_{t-1} \) is used to compute \( z_t \) at every iteration. To see its impact on AMP, consider Algorithm 1 without this term

\[
\begin{align*}
    r_t &= \hat{x}_t + A^T(y - A\hat{x}_t) \\
    \hat{x}_{t+1} &= g(r_t, v_{ht}).
\end{align*}
\]  

By comparing the above recursion to (2.30)-(2.31), we conclude that AMP without the Onsager term is identical to ISTA, if \( g \) is the thresholding function (2.30). As we will see in the simulation experiment coming next, adding this term to ISTA makes the algorithm converge by a few orders of magnitude faster to a fixed point with a much lower MSE.

To understand the error propagation in ISTA (2.67)-(2.68), consider the error vector \( h_t = r_t - x \), which is equivalent to

\[
    h_t = r_t - x = \hat{x}_t - x = A^T(y - A\hat{x}_t) = A^T w + (\hat{x}_t - x) - A^T A (\hat{x}_t - x)
\]

\[
= A^T w + B(\hat{x}_t - x) = A^T w + B q_t,
\]

where we used (2.37) and (2.64), defined a matrix \( B = I - A^T A \in \mathbb{R}^{N \times N} \) and the error vector

\[
q_t = \hat{x}_t - x.
\]  

In Appendix A we define the class of matrices called *Gaussian Orthogonal Ensemble (GOE)* (see Definition 1) and show that when \( A \in \mathbb{R}^{M \times N} \) is a zero-mean i.i.d. Gaussian matrix with variance \( \frac{1}{M} \), \( \delta B \) is a GOE matrix (see Lemma 7). Furthermore, in Appendix A we prove Lemma 8 stating that if we let \( v \in \mathbb{R}^N \) be some vector with \( \nu = \lim_{N \to \infty} \|v\|^2 < \infty \) and independent of a GOE matrix \( B \), then we have that the vector

\[
B v \xrightarrow{\text{(d)}} N(0, \nu I),
\]  

27
converges in distribution to a Gaussian density. This result in combination with (2.69) suggests that at iteration \( t = 0 \), the vector

\[
h_0 = A^T w + B q_0
\]  

(2.72)
is a zero-mean Gaussian vector, if we initialize AMP with \( x_0 = 0 \) and follow Assumption A2 so that \( q_0 = -x \) is independent of \( B \), and since \( w \) is assumed to be independent of \( A \) as well. Therefore, the model (2.64) is actually correct for ISTA at iteration \( t = 0 \). However, if we proceeded to the next iteration \( t = 1 \) and further, the same analysis would fail, since now \( q_1 \) is dependent on \( B \) through \( h_0 \). As a result, we would observe that \( h_t \) for \( t > 0 \) is non-Gaussian and the model (2.64) breaks down.

The rigorous method proposed in [54] for handling this dependency issue is very technical and beyond the scope of this thesis. Nevertheless, we would like to provide a short, although non-rigorous, intuition on what is the role of the Onsager correction and how AMP ensures the model (2.64), since those ideas will further appear in our analysis of the proposed MP algorithms. For this, first, assume that \( w = 0 \) and define the history matrices of the previously observed error vectors \( Q_t = (q_0, ..., q_{t-1}) \) and \( H_t = (h_0, ..., h_{t-1}) \). Then (2.69) suggests these matrices are related as

\[
H_t = B Q_t.
\]  

(2.73)

Therefore, when we consider a product \( B q_t \) at iteration \( t > 0 \), we have to take into account that \( B \) conditioned on \( q_t \) is dependent on \( Q_t \) and \( H_t \) through (2.73). To study this dependence, we consider an orthogonal decomposition of \( B \) into column spaces of \( Q_t \) and \( H_t \) and their complementary subspace

\[
B = Q_t \tilde{B}_t + H_t \bar{B}_t + \hat{B}_t.
\]  

(2.74)

Here, the two matrices \( \tilde{B}_t \) and \( \bar{B}_t \) represent the orthogonal projections of \( B \) on the column space \( Q_t \) and \( H_t \) respectively, while \( \hat{B}_t \) represents the matrix living the complementary subspace. Next, we consider each component of \( B \) separately. First, one can show that when \( B \) is a GOE matrix, the matrix \( \tilde{B}_t \) remains to be GOE and, because it is orthogonal to both \( Q_t \) and \( H_t \), is independent of both \( Q_t \) and \( H_t \) [54]. Therefore, when it is applied to a new error vector \( q_t \), the
Figure 2.2: QQ plot of \( h_t \) for AMP (left) and for ISTA (right) at iteration \( t = 5 \). The size of the problem is \( N = 2^{20} \) and the number of measurements \( M = 2^{19} \).

product \( \hat{B}q_t \) produces a zero-mean i.i.d. Gaussian vector as desired. Next, assume for the moment that the term \( Q_t\tilde{B}_t \) in (2.74) is a zero-matrix. Then, we could apply induction and work out that the product \( H_t\bar{B}_tq_t \) also produces an i.i.d. Gaussian vector, since each column of \( H_t = (h_0, ..., h_{t-1}) \) would be i.i.d. Gaussian (recall that for \( t = 0 \) we have shown that \( h_0 \) is i.i.d. Gaussian). Thus, the only component of \( B \) that makes the product \( Bq_t \) non-Gaussian is the matrix \( Q_t\tilde{B}_t \) in (2.74). One of the key results of [54] was the proof that the vector \( Q_t\tilde{B}_tq_t \) is cancelled out by the Onsager correction term \( \frac{N}{M}\alpha_{t-1}z_{t-1} \), which is not present in the ISTA algorithm (2.67)-(2.68). To numerically illustrate the impact of adding the Onsager correction, in Figure 2.2 we demonstrate the Quantile-Quantile (QQ) plot with respect to a Gaussian distribution of the error vector \( h_t \) for AMP and ISTA algorithms recovering a Bernoulli-Gaussian signal \( x \). In our context, the QQ plot demonstrates how close the distribution of the given samples is to a Gaussian distribution: the more samples are aligned in a straight line, the closer they are to a Gaussian distribution. Figure 2.2 confirms the fact that in AMP \( h_t \) is approximately i.i.d. Gaussian, while for ISTA it is not the case.

In this thesis we discuss and propose different extensions for AMP and all of them use a single or even multiple Onsager correction terms to ensure Gaussianity of certain error vectors propagated in the algorithm. While the specific form of
the Onsager correction is different for different Message Passing algorithms, the fundamental idea behind this term is the same: canceling out the part of \( B \) (or a different matrix formed out of \( A \)) that creates a non-Gaussian random vector. Deriving the closed form and stable ways of estimating this correction for the proposed MP algorithms is one of the main focuses of this thesis.

2.2.4 State Evolution and other properties of AMP

The fact that AMP uses the Onsager correction has multiple implications besides making \( h_t \) zero-mean i.i.d. Gaussian. In particular, one can use this fact to rigorously derive the State Evolution (SE) – a 1D model that predicts the asymptotic MSE of \( r_t \), i.e. the variance \( v_{h_t} \) of the error vector \( h_t \) in the LSL. The complete derivation of this result is very involved and an interested reader should refer to the original work [54]. However, here we will provide a very short but non-rigorous intuition for how SE is linked to AMP. Consider the variance \( v_{h_t} \) of the error vector \( h_t \) updated as in (2.69),

\[
\lim_{N \to \infty} \frac{1}{N} \| h_t \|^2 = \lim_{N \to \infty} \frac{1}{N} \| A^T w + B q_t \|^2.
\]

(2.75)

Once again, we refer to the decomposition (2.74) and assume that adding the Onsager term \( \frac{N}{M} \alpha_{t-1} z_t \) cancels out the term \( Q_t \tilde{B}_t q_t \) of the product \( B q_t \). As discussed before, if this assumption is correct, then one can show [54] that the product \( B q_t \) generates an i.i.d. Gaussian vector with the variance \( \delta^{-1} v_{q_t} \), where

\[
v_{q_t} = \lim_{N \to \infty} \frac{1}{N} \| q_t \|^2.
\]

(2.76)

Therefore, by expanding the norm in (2.75) we obtain

\[
v_{h_t} \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} \left( \| A^T w \|^2 + 2 w^T A B q_t + \| B q_t \|^2 \right)
\]

\[
\overset{a.s.}{=} v_w + \delta^{-1} v_{q_t} + 2 \lim_{N \to \infty} \frac{1}{N} w^T A B q_t,
\]

(2.77)

where we used Lemma 5, the fact that \( w \) is i.i.d. Gaussian with the variance \( v_w \) and the normalization \( \frac{1}{N} \text{Tr}\{A A^T\} = 1 \) of \( A \) in combination with Lemma 6 to obtain \( \lim_{N \to \infty} \frac{1}{N} \| A^T w \|^2 \overset{a.s.}{=} v_w \). Lastly, one can show that when the signal \( x \) and the measurement noise \( w \) are independent, the inner-product \( \frac{1}{N} w^T A B q_t \) almost surely converges to zero under Assumptions A1-A3 [60]. As a result, we
can recursively define the evolution of the variances $v_h$ and $v_q$ in the following way [54]

$$v_{h_{t+1}} = v_w + \delta^{-1}E(v_h)$$  \hspace{1cm} (2.78)

$$E(v) = \lim_{N \to \infty} \frac{1}{N} \mathbb{E} \left[ \|g(x + \sqrt{v}z, v) - x\|^2 \right] z \sim \mathcal{N}(0, I), x \sim p(x)$$  \hspace{1cm} (2.79)

which is initialized with $v_{h_0} = v_w + \delta^{-1}v_x$ as follows from the discussion from Section 2.2.3. Note that this initialization of SE is motivated by the fact that we initialize AMP with $x_0 = 0$, which is an effective initialization for the linear measurements $y = Ax + w$ and ensures that the SE will not be stuck at a trivial fixed point (see the analysis in Section 6 of [61] and the analysis in the works [62, 63]).

The SE recursion (2.78)-(2.79) is useful for multiple reasons. First, if one aims to minimize the SE, then this becomes equivalent to minimizing the MSE $E(v_h)$ from (2.79). This suggests to design $g$ to be a denoiser, since the input to the function $g$ acts as a Gaussian channel in the LSL. This characteristic indicates that AMP may be able to provably recover signals $x$ with a much more complex structure than sparsity. By comparison, ISTA as defined in (2.67)-(2.68) focuses on approximately sparse signals.

When $g$ represents the MMSE denoiser as in (2.62), it was shown in [64] that AMP is provably convergent in the LSL. Moreover, combining the results from [54] and [65], one can show that AMP is Bayes-optimal and converges to the desired conditional mean solution (2.41). The complete proof of this result can be found in [66].

To illustrate the power of AMP when the MMSE denoiser is available, consider recovering a Bernoulli-Gaussian signal, $x_i \sim \rho \delta(x_i) + (1 - \rho)\mathcal{N}(0, \sigma^2), i \in (1, ..., N)$, where $\rho = 0.9$ is the chance of the entry to take zero value. We choose $A$ to be a Gaussian matrix, $v_w$ that achieves 40dB SNR and compression rate $\delta = 0.5$.

We compare the normalized MSE (NMSE) $\frac{\|\hat{x} - x\|^2}{\|x\|^2}$ of AMP against FISTA$^3$ from Section 2.1.3. The NMSE averaged over 10 realizations is shown in Figure 2.3. As is seen, FISTA requires an order more iterations to converge and the fixed point that is worse compared to the fixed point of AMP, while the per-iteration

$^3$We used the MATLAB implementation from the library available at https://sourceforge.net/projects/gampmatlab/
computational cost of the two algorithms is approximately the same.

However, the MMSE denoisers might be not available for complex signals $x$ like natural images. In [54] it was shown that one can be flexible with the choice of the denoiser $g$ and use not only the MMSE denoisers (2.62), but any separable Lipschitz functions. Further, in [56] this result was generalized to any non-separable uniformly Lipschitz denoisers $g$ following Assumption A2. This suggests that one can recover signals like natural images by employing powerful Plug-and-Play (PnP) denoisers such as NLM [67], BM3D [68], DnCNN [58] and others. Although sparsity based denoisers, where we assume an image to be sparse in some transform domain like wavelets, are also capable of denoising natural images, these denoisers substantially underperform when compared to, for example, BM3D.

Moreover, if the denoiser $g$ is dependent on some free parameter $\theta_t$, like patch size, window size, distance between patches etc, then one can follow the analysis from [69] to show that the greedy method of choosing $\theta_t$, which minimizes $E(v_{h_t})$ from (2.79) at every iteration $t$, is the globally optimal strategy, assuming that $E(v)$ monotonically decreasing with $v$. When $\theta_t$ is chosen in this way, the combined set of parameters $\theta_t = (\theta_0, ..., \theta_t)$ used at every iteration $\tau \leq t$ produces the lowest possible variance $v_{h_t}$ [69]. In practice, this parameter selection can be carried out by constructing an estimate of the MSE $E(v_{h_t})$ by using the popular Stein’s
Unbiased Risk Estimator (SURE) \cite{60} that takes the following form \cite{1}

\[ \hat{\mathcal{E}}_t(\theta) = \frac{1}{N} \| \mathbf{r}_t - \mathbf{g}(\mathbf{r}_t, v_{ht}, \theta) \|^2 - v_{ht} + 2v_{ht} \alpha_t(\theta) \] (2.80)

which we explicitly defined as a function of \( \theta \). Here, the scalar \( \alpha_t(\theta) \) represents the divergence of \( \mathbf{g}(\mathbf{r}_t, v_{ht}, \theta) \) with respect to \( \mathbf{r}_t \) as in Algorithm 1. One can confirm that \( \hat{\mathcal{E}}_t(\theta) \) is a consistent estimator of \( \mathcal{E}(v_{ht}) \), \( \lim_{N \to \infty} \hat{\mathcal{E}}_t(\theta) \overset{a.s.}{=} \mathcal{E}(v_{ht}) \). Given the scalar \( \alpha_t(\theta) \), the complexity of evaluating the estimate \( \hat{\mathcal{E}}_t(\theta) \) is dominated by computing one inner-product of \( N \) dimensional vectors, which usually is negligible compared to evaluating the denoiser \( \mathbf{g} \) or computing the matrix-vector product with \( \mathbf{A} \) as in Algorithm 1. Then, provided the estimate \( \hat{\mathcal{E}}_t(v_{ht}) \), one can optimize \( \theta_t \) as

\[ \theta^*_t = \min_{\theta} \hat{\mathcal{E}}_t(\theta) \] (2.81)

This approach has been used within AMP in multiple works, including \cite{70, 71}. However, the task of estimating the divergence \( \alpha_t(\theta) \) is not trivial for PnP denoisers and one of the goals of this thesis is to develop fast and consistent estimators for this. This problem will be considered in Chapter 5. Note that access to the MSE of the produced estimates is helpful not only for optimizing the performance of AMP, but also as a tool to do an early stopping or have an uncertainty measure, which might be useful in applications like medical imaging.

### 2.2.5 Denoisers for image reconstruction

Before proceeding next, we would like to stop for a moment and give a brief introduction to several denoising methods that will be used throughout the thesis: Non-local Means (NLM), Block-matching and 3D filtering (BM3D) and Denoising Convolutional Neural Network (DnCNN). These examples are off-the-shelf denoisers that demonstrate (nearly) state-of-the-art performance in image denoising problems and they substantially differ from classical smoothing filters. Historically, filtering a signal \( \mathbf{x} \) from noisy measurements \( \mathbf{r} = \mathbf{x} + \mathbf{w} \), where \( \mathbf{w} \) is white noise, was done by performing some local averaging of pixels. That was motivated by the fact that natural images tend to be locally smooth while the values of noise in the neighbouring pixels are independent. Examples of such filters include Gaussian Smoothing filters \cite{72}, Anisotropic Filters \cite{73}, Total Variation filters \cite{74, 75}.

33
Since then, a new class of denoisers has arisen that use not only neighbouring pixels, but the whole image to filter each pixel. Such type of algorithms is usually referred to as non-local methods and they exploit the fact that patches located far from each other might be very similar and, therefore, can help to filter each other. One of the first successful examples leveraging this fact was the NLM filter [67, 76], which generates a filtered value $\hat{x}(k)$ of the pixel $k$ as

$$\hat{x}(k) = \frac{1}{Z(k)} \sum_{u=1}^{N} r(u) w(k, u),$$  \hspace{1cm} (2.82)

where $N$ is the size of the image, $r(u)$ is the $u$-th pixel of the noisy image $r$ and $w(k, u)$ is a weighting function, like a Gaussian function

$$w(k, u) = \frac{1}{Z(k)} e^{\frac{|r(\Psi_k) - r(\Psi_u)|^2}{h}}.$$  \hspace{1cm} (2.83)

Here, $h$ is a parameter defining the degree of filtering, $\Psi_k$ is a vector of pixels in the square neighbourhood of pixel $k$ and $Z(k) = \sum_{u=1}^{N} e^{\frac{|r(\Psi_k) - r(\Psi_u)|^2}{h}}$ is the normalization scalar. While local filters average over only neighbouring pixels, NLM finds similar patches and, effectively, increases the area to average over. As a result, NLM filter is able to substantially reduce the noise level while preserving more details of the original image [76]. To improve the quality of denoising, the user can optimize the size of the patch $\Psi_k$ and the parameter $h$ in (2.83) using, for example, the SURE method discussed above.

Another denoiser that leverages non-local similarities of natural images is the BM3D denoiser [68]. While NLM performs filtration in the original domain by averaging over similar patches, BM3D processes such patches in several steps. First, it picks $k$ reference patches in the image to form that number of clusters and groups the rest of the patches by matching them with the reference. If the distance (like $l^2$-norm) between the reference and some patch is smaller than a threshold, then the patch goes in the cluster associated with the reference. Next, all the patches in a cluster are stacked into a 3D array, which is further transformed by a linear operator (like Discrete Fourier transform (DFT)) and filtered using, for example, hard-thresholding or Wiener filtering. After that, the inverse transform is applied to each 3D array and the patches are returned to their original places. This process repeats twice (see additional information in [68]). To improve the
performance, one can optimize the parameters of the denoiser like the size of the patch, their number, the threshold value in the matching procedure etc.

Lastly, we will frequently use DnCNN in our simulations. This denoiser represents a deep convolutional neural network (NN) that is learnt to predict the residual of the signal’s estimate, i.e. instead of learning a mapping $\mathcal{H}(r) = \hat{x}$ the generates an estimate of the signal $x$ directly, DnCNN forms the estimate as $\hat{x} = r - \mathcal{R}(r)$ and learns the mapping $\mathcal{R}(r)$ [77]. This approach allows achieving higher training and testing accuracy for deeper NN (for more information on this, please refer to [77, 78]). Choosing the convolutional NN architecture empirically demonstrates better estimation accuracy, simplifies the learning process and allows using GPU to accelerate implementation of the denoiser. In this thesis we use a pretrained DnCNN with 20 layers from the D-AMP Toolbox$^4$.

2.2.6 Limitations of AMP

The AMP algorithm discussed above has many important advantages. In particular, AMP is a first-order method that does not require storing any high-dimensional data besides a few $N$ dimensional vectors, and is capable of recovering large-scale signals. It has a rigorously defined dynamics that was used to prove the convergence and Bayes-optimality of the algorithm, and that provides an inbuilt uncertainty measure. Besides that, AMP can benefit from PnP denoisers to recover complex data like natural images.

However, AMP is limited in the range of the inverse problems it can reliably solve. The main issue is that the measurement matrix $A$ is constrained to be from a class of i.i.d. matrices with a certain fixed singular value distribution, which is a fairly small class of random matrices. In particular, when $A$ is designed to be ill-conditioned or have a non-zero mean, AMP is evidenced to diverge [10, 79, 80, 81]. There are two possible directions of work to make AMP more stable for a broader range of $A$. The first one focuses on modifying the algorithm in order to stabilize AMP. In particular, one can apply damping [80, 81], mean-removal [81], sequential update of entries of the internal vectors in AMP instead of the parallel update [82] and other methods. Yet, these methods have significant disadvantages.

In the damping approach, the resulting algorithm might take orders more iterations to converge [10] and it is not entirely clear how to design the optimal damping parameters. The sequential update instead of the parallel one in AMP leads to an increase in the computational cost and inability to use fast measurement matrices A like Fourier operators [81].

In the next section we review the alternative path of designing Message Passing algorithms that would be able to operate with a wider range of measurement matrices while preserving most the main advantages of AMP.

2.3 Expectation Propagation and Vector AMP

In the previous section, we considered using BP to approximate the posterior density \( p(x|y) \) from (2.60) in an iterative fashion. Unfortunately, the exact application of BP led to an algorithm with an intractable computational cost so a series of additional approximations was required to upscale the algorithm. As a result, we arrived at the AMP algorithm. In this section we take an alternative route and use the Expectation Propagation (EP) framework described in Section 2.2.1 to construct an algorithm that is able to solve more general CS inverse problems while having a potential to operate as a first-order method, and which will be the basis for our work.

Recall that EP aims to approximate a joint density \( f(x) \) that is assumed to factorize as

\[
f(x) = \prod_j f_j(x_j)
\]

by a function

\[
\tilde{f}(x) = \prod_j \tilde{f}_j(x_j),
\]

where \( \tilde{f}_j(x_j) \) approximates the exact factor \( f_j(x_j) \). Depending on what constraints are put on every factor \( \tilde{f}_j(x_j) \), one obtains a different variant of EP. In Section 2.2.1 we used the general EP structure (2.53)-(2.58) with the expectation constraint as in (2.49), which implies the marginalization constraint (2.52), to define BP [47]. In this section, we use EP in a more classical way and assume that each factor \( \tilde{f}_j(x_j) \)
Background

belongs to a certain family of densities and impose a constraint on the moments of these factors. In particular, we will stick to the family of isotropic Gaussian densities, where

\[ \tilde{f}_j(x_j) = N(x_j; \mu_j, v_j I) \] (2.86)

for every \( j \), and will impose the expectation constraint (2.46) with two functions

\[ \phi_1(x) = x \]
\[ \phi_2(x) = xx^T. \] (2.87)

Note that since the family of (unnormalized) isotropic Gaussian densities is closed under multiplication [83], the model (2.86) implies that the approximated joint density (2.84) is also a Gaussian density,

\[ \tilde{f}(x) = \frac{1}{Z} N(x; \mu, v I), \] (2.88)

where \( Z \) is a normalization scalar.

The change of the constraint does not affect the main body of EP (2.53)-(2.58) apart from the step (2.57) where a new approximated joint density \( \tilde{f}^{\text{new}}(x) \) is computed. By definition, this step minimizes the KL divergence between \( \tilde{f}^{\text{new}}(x) \) and the tilted density \( \tilde{q}(x) \) under a constraint on the former density. Due to the new constraints (2.86)-(2.87), one can show [45] that this minimization process produces a density \( \tilde{f}^{\text{new}}(x) = \frac{1}{Z} N(\mu^{\text{new}}, v^{\text{new}} I) \), where

\[ \mu^{\text{new}} = \int x\tilde{q}(x)dx \] (2.89)
\[ v^{\text{new}} N + \|\mu^{\text{new}}\|^2 = \int \|x\|^2\tilde{q}(x)dx. \] (2.90)

The steps (2.89)-(2.90) are sometimes referred as moment matching. The reason for sticking to the family of Gaussian densities with the simplest type of covariance matrix is twofold. First, manipulating with Gaussian densities with more complex covariance matrices might result in a much higher computational cost of the resulting EP algorithm [84]. Second, as we will see latter, the isotropic Gaussian family turns out to be asymptotically sufficient for the CS problem considered. Lastly, since both \( \tilde{f}(x) \) and \( \tilde{f}_j(x_j) \) are Gaussian, the computation of the cavity density in (2.53) and of the updated factor in (2.58) substantially simplifies and
boils down to updating the mean and the variance.

We apply the above EP algorithm to approximate the posterior density $p(x|y)$ that is kept in the original form

$$p(x|y) = \frac{1}{Z}p(y|x)p(x)$$

(2.91)

with only two factors $f_1(x) = p(y|x)$ and $f_2(x) = p(x)$. Due to such a simple structure, the EP steps (2.53)-(2.58) with the step (2.57) replaced by (2.89)-(2.90) can be grouped into two symmetric blocks: Block A and Block B that are associated with the likelihood density $p(y|x)$ and with the prior density $p(x)$ respectively. In both blocks we form the approximated factors

$$q_{A\rightarrow B}(x) = \mathcal{N}(x; x_{A\rightarrow B}, v_{A\rightarrow B}I)$$

(2.92)

$$q_{B\rightarrow A}(x) = \mathcal{N}(x; x_{B\rightarrow A}, v_{B\rightarrow A}I),$$

(2.93)

where $q_{A\rightarrow B}(x)$ approximates $p(y|x)$ and $q_{B\rightarrow A}(x)$ approximates $p(x)$ respectively (in the previously defined notation, $q_{A\rightarrow B}(x)$ and $q_{B\rightarrow A}(x)$ are $\tilde{f}_1(x)$ and $\tilde{f}_2(x)$, respectively). To update these densities, in each block we construct the associated tilted densities

$$q_A(x) = q_{B\rightarrow A}(x)p(y|x)$$

(2.94)

$$q_B(x) = q_{A\rightarrow B}(x)p(x)$$

(2.95)

with the mean vectors

$$\mu_A = \int x q_{B\rightarrow A}(x)p(y|x)dx$$

(2.96)

$$\mu_B = \int x q_{A\rightarrow B}(x)p(x)dx$$

(2.97)

and the variances

$$v_A = \frac{1}{N} \int \|x\|^2 q_{B\rightarrow A}(x)p(y|x)dx - \frac{1}{N} \|\mu_A\|^2$$

(2.98)

$$v_B = \frac{1}{N} \int \|x\|^2 q_{A\rightarrow B}(x)p(x)dx - \frac{1}{N} \|\mu_B\|^2,$$

(2.99)

respectively. Note that we dropped the implicit dependence of the densities $q_{A\rightarrow B}(x)$, $q_{B\rightarrow A}(x)$, $q_A(x)$ and $q_B(x)$ on $y$ and $A$ to simplify the notation. Similarly, we dropped the normalization step that ensures that the tilted densities $q_A(x)$ and $q_B(x)$ are proper densities (see the exact steps in Section 2.2.1).
The steps (2.94)-(2.95) shed light on the chosen notation of densities $q_{A \rightarrow B}(x)$ and $q_{B \rightarrow A}(x)$: these densities not only approximate the associated factors in (2.91), but also represent the information sent from one block to another to form the corresponding tilted densities. Once the tilted densities are formed, they can be used to update the approximated factors

\[
q_{A \rightarrow B}(x) = \frac{\text{proj}[q_A(x)]}{q_{B \rightarrow A}(x)} = \frac{\mathcal{N}(\mu_A, v_A I)}{\mathcal{N}(x_{B \rightarrow A}, v_{B \rightarrow A} I)},
\]

(2.100)

\[
q_{B \rightarrow A}(x) = \frac{\text{proj}[q_B(x)]}{q_{B \rightarrow A}(x)} = \frac{\mathcal{N}(\mu_B, v_B I)}{\mathcal{N}(x_{A \rightarrow B}, v_{A \rightarrow B} I)},
\]

(2.101)

where the operator $\text{proj}[f(x)]$ performs the moment matching procedure (2.89)-(2.90) and returns an isotropic Gaussian density with the first two moments matching the same moments of the input density. The above EP algorithm is illustrated on a diagram in Figure 2.4. Note that in the above EP iterations we do not need to fully compute the tilted densities $q_A$ and $q_B$ as in (2.94) and (2.95), but only need the first two moments of those densities to form the updates (2.100)-(2.101).

In [10] the authors followed the same route and derived the updates of the means and variances for the densities (2.92)-(2.93) based on (2.94)-(2.101), and arrived at the EP algorithm summarized in Algorithm 2. Here, each block contains a function $g_s$ that computes the mean $\mu_s$ of the associated tilted distributions $q_s$. Due to the structure of $q_B$ from (2.95), the mean vector (2.97) is computed by

\[
g_B(x_{A \rightarrow B}, v_{A \rightarrow B}) = \mathbb{E}[x | x'_{A \rightarrow B} = x + \sqrt{v_{A \rightarrow B}} h, h \sim \mathcal{N}(0, I), x \sim p(x)]
\]

(2.102)

that represents the MMSE denoiser applied to the intrinsic Gaussian channel $x'_{A \rightarrow B}$. Next, since we assume the measurement noise $w$ to be zero-mean i.i.d. Gaussian and due to the definition of $\mu_A$ from (2.96), $g_A$ represents the Linear Minimum Mean Squared Error (LMMSE) estimator

\[
g_A(x'_{B \rightarrow A}, v_{B \rightarrow A}) = (v_w^{-1} A^T A + (v_{B \rightarrow A})^{-1} I)^{-1} (v_w^{-1} A^T y + (v_{B \rightarrow A})^{-1} x'_{B \rightarrow A})
\]

(2.103)

Similarly to AMP, EP uses the divergence $\gamma_s$ of the respective estimator $g_s$ to form an Onsager correction term. Its role is to ensure a certain asymptotic behaviour of EP, which will be discussed in more detail in the following section.

39
Figure 2.4: Diagram of the EP algorithm for CS.
Besides that, the divergence $\gamma^t(\cdot)$ is used to form an estimate of the respective tilted variance $v^t(\cdot)$. The latter are then utilized to construct an estimate of variances $v^t_{A\rightarrow B}$ and $v^t_{B\rightarrow A}$ of the approximated densities $q^t_{A\rightarrow B}(x)$ and $q^t_{B\rightarrow A}$, respectively.

### Algorithm 2: EP/VAMP for CS [10, 79, 85]

**Initialization:** $x^0_{B\rightarrow A} = 0$, $v^0_{B\rightarrow A} = \hat{v}_x$, $t = 0$

1. while $t < T_{\text{max}}$ and $\hat{v}^t_{B\rightarrow A} \geq \epsilon$ do
   2. Block A
      3. $\mu^t_A = g_A(x^t_{B\rightarrow A}, \hat{v}^t_{B\rightarrow A}, \hat{v}_w)$
      4. $\gamma^t_A = \frac{1}{N} \nabla(x^t_{A\rightarrow B}) \cdot g_A(x^t_{B\rightarrow A}, \hat{v}^t_{B\rightarrow A}, \hat{v}_w)$
      5. $\hat{v}^t_A = \gamma^t_A \hat{v}^t_{B\rightarrow A}$
      6. $x^{t+1}_{A\rightarrow B} = \frac{1}{1-\gamma^t_A} (\mu^t_A - \gamma^t_A x^t_{A\rightarrow B})$
      7. $\hat{v}^{t+1}_{A\rightarrow B} = \frac{\hat{v}^t_A - (\gamma^t_A)^2 \hat{v}^t_{B\rightarrow A}}{(1-\gamma^t_A)^2}$
   3. Block B
      8. $\mu^t_B = g_B(x^t_{A\rightarrow B}, \hat{v}^t_{A\rightarrow B})$
      9. $\gamma^t_B = \frac{1}{N} \nabla(x^t_{A\rightarrow B}) \cdot g_B(x^t_{A\rightarrow B}, \hat{v}^t_{A\rightarrow B})$
     10. $\hat{v}^t_B = \gamma^t_B \hat{v}^t_{A\rightarrow B}$
     11. $x^{t+1}_{B\rightarrow A} = \frac{1}{1-\gamma^t_B} (\mu^t_B - \gamma^t_B x^t_{A\rightarrow B})$
     12. $\hat{v}^{t+1}_{B\rightarrow A} = \frac{\hat{v}^t_B - (\gamma^t_B)^2 \hat{v}^t_{A\rightarrow B}}{(1-\gamma^t_B)^2}$
     13. $t = t + 1$

**Output:** $\mu^t_B$

Lastly, since in practice we usually do not have the access to the true values of the variances $v_x = \lim_{N \rightarrow \infty} \frac{1}{N} ||x||^2$ and $v_w = \lim_{N \rightarrow \infty} \frac{1}{N} ||w||^2$ of the true signal $x$ and of the noise measurement $w$ respectively, the algorithm uses their estimated versions $\hat{v}_x$ and $\hat{v}_w$. Similarly, we define $\hat{v}^t_A$, $\hat{v}^t_B$, $\hat{v}^t_{A\rightarrow B}$ and $\hat{v}^t_{B\rightarrow A}$ to be estimates of $v^t_A$, $v^t_B$, $v^t_{A\rightarrow B}$ and $v^t_{B\rightarrow A}$ respectively.

### 2.3.1 Error Model and Asymptotic Properties of EP

Although EP has been used for the signal recovery for many years, including the works [83, 84, 86, 87, 88] and many others, none of them provided the exact, rigorous description of how the error propagates in the algorithm. Ideally, we would like to confirm that the chosen form of estimators (2.102) and (2.103) are...
consistent and there is an asymptotic model that predicts the intrinsic uncertainty in the algorithm as in AMP. For EP, such an LSL analysis was first non-rigorously proposed in [79] and soon after, the works [10] and then [85] provided the rigorous analysis under the following three assumptions. The first one is the same as for AMP from Section 2.2.3 and the second assumption is the same as well, but with the notations adapted to EP from Algorithm 2.

**Assumption B1.** The dimensions of the signal model $N$ and $M$ approach infinity with a fixed ratio $\delta = \frac{M}{N} = O(1)$

**Assumption B2.** The function $g_B$ is uniformly Lipschitz so that the sequence of functions $g_B : \mathbb{R}^N \mapsto \mathbb{R}^N$ indexed by $N$ are Lipschitz continuous with a Lipschitz constant $L_N < \infty$ as $N \to \infty$ [58], [56]. Additionally, we assume the sequences of the following inner-products exist and are almost surely finite as $N \to \infty$ [58]

$$\lim_{N \to \infty} \frac{1}{N} g_B(x + d_1)^T g_B(x + d_2)$$

$$\lim_{N \to \infty} \frac{1}{N} d_1^T g_B(x + d_2)$$

$$\lim_{N \to \infty} \frac{1}{N} x^T d_1$$

$$\lim_{N \to \infty} \frac{1}{N} \|x\|^2$$

and

$$\lim_{N \to \infty} \frac{1}{N} \nabla d_1 \cdot g_B(x + d_1) = \lim_{N \to \infty} \frac{1}{(C)_{1,2}} \frac{1}{N} d_2^T g_B(x + d_1). \quad (2.104)$$

Here $d_1, d_2 \in \mathbb{R}^N$ with $((d_1)_n, (d_2)_n) \sim \mathcal{N}(0, C)$ for some positive definite $C \in \mathbb{R}^2$.

The third assumption will allow us to carry out the analysis of EP for a broader class of random matrices compared to the AMP case. Namely, we will allow $A$ to be from the class of right-orthogonally invariant matrices that is defined next.

**Assumption B3.** The measurement matrix $A$ is Right-Orthogonally Invariant (ROI), such that in the SVD of $A = USV^T$, the matrix $V$ is independent of other random terms and is Haar distributed, i.e. is uniformly distributed on the set of orthogonal matrices, while $U$ is allowed to be any orthogonal matrix and the matrix $S^T S$ is allowed to have any Limiting Eigenvalue Distribution with a compact support.

Note that the family of matrices in Assumptions A2 is a special case of ROI matrices, where both singular vector matrices $U$ and $V$ are Haar distributed and
the singular vector matrix $S$ is constrained such that its limiting spectrum follows the Marchenko-Pastur distribution.

Recall that in AMP we were concerned with how certain error vectors interact with a GOE matrix $B$ and how we can ensure Gaussianity of $h_t$ updated as in (2.69). In EP, we are interested in studying how the output error of each block

$$h_t = x_{A \rightarrow B} - x$$  \hspace{1cm} \text{(2.105)}
$$q_t = x_{B \rightarrow A} - x$$  \hspace{1cm} \text{(2.106)}

and their transformed versions

$$m_t = V^T h_t$$  \hspace{1cm} \text{(2.107)}
$$b_t = V^T q_t$$  \hspace{1cm} \text{(2.108)}

evolve with iteration $t$ and how they interact with a Haar matrix $V$. The authors of [10] showed that by applying the SVD of $A = USV^T$ to Algorithm 2, one can define an equivalent iterative process of the above error vectors

$$b_t = V^T q_t$$
$$m_t = \frac{1}{1 - \gamma_A}(\tilde{g}_A(b_t, \hat{v}_{B \rightarrow A}^t, \hat{v}_w) - \gamma_A b_t)$$  \hspace{1cm} \text{(2.109)}
$$h_t = V m_t$$
$$q_t = \frac{1}{1 - \gamma_B}(\tilde{g}_B(h_t, \hat{v}_{A \rightarrow B}^t) - \gamma_B h_t),$$  \hspace{1cm} \text{(2.110)}

where $\tilde{g}_A$ is the SVD version of (2.103)

$$\tilde{g}_A(b_t, \hat{v}_{B \rightarrow A}^t, \hat{v}_w) = \left((\hat{v}_w)^{-1} S^T S + (v_{B \rightarrow A}^t)^{-1}\right)^{-1}\left((\hat{v}_w)^{-1} S w + (v_{B \rightarrow A}^t)^{-1} b_t\right)$$  \hspace{1cm} \text{(2.111)}

and

$$\tilde{g}_B(h_t, \hat{v}_{A \rightarrow B}^t) = g_B(x + h_t, \hat{v}_{A \rightarrow B}^t) - x$$  \hspace{1cm} \text{(2.112)}

is the error of the MMSE denoiser $g_B$ at iteration $t$. Just like in the AMP case, we construct the history matrices $Q_{t+1} = (q_0, ..., q_t)$, $B_{t+1} = (b_0, ..., b_t)$, $M_{t+1} = (m_0, ..., m_t)$ and $H_{t+1} = (h_0, ..., h_t)$ that can be related through (2.107)-(2.108) as

$$(M_t, B_t) = V^T(H_t, Q_t).$$  \hspace{1cm} \text{(2.113)}

Next it was shown that one can decompose $V$ into the subspaces spanned by the above error vector matrices, similarly to how it was done in (2.74) for AMP. Further, this decomposition was used to prove that the Onsager correction terms in
EP cancel out the structured parts of this decomposition so that the matrix-vector products

$$
\lim_{N \to \infty} b_t = \lim_{N \to \infty} V^T q_t \overset{a.s.}{\sim} \mathcal{N}(0, v_{B \to A}^T A) \tag{2.114}
$$

$$
\lim_{N \to \infty} h_t = \lim_{N \to \infty} V m_t \overset{a.s.}{\sim} \mathcal{N}(0, v_{A \to B}^T B) \tag{2.115}
$$
generate zero-mean i.i.d Gaussian vectors at each iteration $t$ in the LSL [10, 85]. The Onsager correction terms that ensure this property are $\gamma_{A \to B}^t b_t$ and $\gamma_{B \to A}^t h_t$ [10, 85]. Similarly to the AMP case, (2.114)-(2.115) support the idea to design $g_B$ to be the MMSE denoiser as in (2.102) and $g_A$ to be the LMMSE estimator as in (2.103).

Besides the asymptotic Gaussianity of $h_t$ and $b_t$, EP possesses most of other advantages of AMP. In particular, one can define the State Evolution that predicts the intrinsic uncertainty in EP. For this, define two error functions [10]

$$
\mathcal{E}_A(v) = \int \frac{1}{v^{-1} + v^{-1}} p(\lambda) d\lambda \tag{2.116}
$$

$$
\mathcal{E}_B(v) = \lim_{N \to \infty} \frac{1}{N} \mathbb{E}[\|g_B(x + \sqrt{v} z, v) - x\|^2 | z \sim \mathcal{N}(0, I), x \sim p(x)], \tag{2.117}
$$

where $\rho(\lambda)$ is the asymptotic distribution of the eigenvalues of $\Lambda = S^T S$, and two sensitivity functions

$$
\gamma_A(v) = \int \frac{1}{v^{-1} + v^{-1}} p(\lambda) d\lambda \tag{2.118}
$$

$$
\gamma_B(v) = \frac{\mathcal{E}_B(v)}{v}. \tag{2.119}
$$

Then, one can show that the variances $v_{A \to B}^t = \lim_{N \to \infty} \frac{1}{N} \|h_t\|^2$ and $v_{B \to A}^t = \lim_{N \to \infty} \frac{1}{N} \|q_t\|^2$ evolve as [10]

$$
v_{A \to B}^t \overset{a.s.}{=} \frac{\mathcal{E}_A(v_{B \to A}^t) - \gamma_A(v_{B \to A}^t)^2 v_{B \to A}^t}{(1 - \gamma_A(v_{B \to A}^t))^2} \tag{2.120}
$$

$$
v_{B \to A}^{t+1} \overset{a.s.}{=} \frac{\mathcal{E}_B(v_{A \to B}^t) - \gamma_B(v_{A \to B}^t)^2 v_{A \to B}^t}{(1 - \gamma_B(v_{A \to B}^t))^2} \tag{2.121}
$$

The SE of EP can be used to prove similar asymptotic properties as in the AMP case. In particular, EP was recently shown to be provably convergent [13,
and its fixed points, when unique, were conjectured to be Bayes-optimal [10] under Assumptions B1-B3.

Next, we would like to quickly return to the question whether the family of isotropic Gaussian densities is the best choice for EP. An obvious advantage of using isotropic Gaussians is the reduced number of variables one needs to estimate throughout the algorithm. On the other hand, one could use the family of Gaussian densities with, for example, a diagonal covariance as proposed in the works [90, 91] and instead of a single variance $v_A^t$ (similarly with $v_B^t$), estimate a vector $v_A^t \in \mathbb{R}^N$. This operation is more computationally demanding and is less robust (see the conclusion section in Chapter 6), but one might wonder whether this change would give some advantage over the isotropic family. However, the authors of [46] proved that under Assumptions B1-B3, the fixed points of an EP algorithm with a diagonal covariance and the same EP but with an isotropic covariance are the same. Convergence-wise, the EP algorithm with isotropic covariance matrices has not demonstrated any convergence issues based on the numerical experiments demonstrated in this thesis and by other authors [10, 14, 58, 79, 92, 93] when the measurement matrix $A$ is ROI and the size of the inverse problem is sufficiently large ($N$ and $M$ are above $10^3$). As a result, under these conditions, the EP algorithm with isotropic covariance might be a more preferable choice.

Lastly, we discuss the range of inverse problems EP from Algorithm 2 can handle while preserving the asymptotic properties discussed before. Originally, the above analysis for EP was carried out in the context where $A$ is ROI from Assumption B3. While this class of matrices is more general than the class of i.i.d. random matrices suitable for the AMP algorithm, the two classes suffer from the same problem: they do not have a fast implementation that would make the matrix-vector product with $A$ cost of order $O(N \log N)$, while implementing this operation at the cost $O(N^2)$ is intractable in large-scale inverse problems. However, very recently, the works [11, 12] rigorously extended the class of allowed matrices for EP to include the so-called Fast ill-conditioned Johnson Lindenstrauss Transform (FIJLT)

$$A = \text{SPHD},$$

which is composed of the following matrices: the values of the diagonal matrix $D$
are either $-1$ or $1$ with an equal probability; the matrix $H$ is some fast Hadamard transform, like Discrete Cosine Transform (DCT); The matrix $P$ is a random permutation matrix and $S$ is the singular values matrix of dimension $M$ by $N$ with the desired condition number. The complexity of the matrix-vector product with this kind of matrices is dominated by implementing the chosen Hadamard transform, whose complexity is usually of order $O(N \log(N))$ or even $O(N)$. Moreover, these types of matrices are not required to be stored and instead one can efficiently compute the matrix-vector products on-the-fly.

### 2.3.2 Vector Approximate Message Passing

The EP algorithm shown in Algorithm 2 assumes that we have the access to the MMSE denoiser $g_B$ for the chosen signal $x$. However, as we discussed for the AMP case, this might be not the case for certain real signals like natural images. To make the algorithm more flexible and practical, the authors of [10] rigorously extended the class of allowed denoisers to any separable Lipschitz function $g_B$, and further generalized to any non-separable Lipschitz continuous denoiser following Assumption B2 [58]. The resulting algorithm was called *Vector Approximate Message Passing (VAMP)*, which was rigorously shown to preserve most of the properties of EP. In particular, it was shown that the asymptotic Gaussianity of $b_t$ and $h_t$ in (2.115) and (2.115) is preserved as long as the key ingredient – the Onsager correction $\gamma_{t_B}^{x} A \rightarrow B$ – is computed correctly, i.e. $\gamma_{t_B}^{x}$ is chosen to be a consistent estimate of the denoiser’s divergence. Moreover, VAMP preserves the SE that is almost identical to (2.120)-(2.121) with the only change in the sensitivity function $\gamma_B$, which becomes [58]

\[
\gamma_B(v) = \lim_{N \rightarrow \infty} \frac{1}{N} \nabla_{(x + \sqrt{v} \mathbf{h})} \cdot g_B(x + \sqrt{v} \mathbf{h}, v).
\] (2.123)

Additionally, one can leverage the SURE estimation technique discussed at the end of Section 2.2.4 to optimize the denoiser to obtain higher efficiency of VAMP. As a result, we obtain a flexible algorithm for recovering complex data like natural images measured by a wider class of random matrices compared to the AMP case. Since VAMP generalizes the EP algorithm, in the following we will stick to the name VAMP when we refer to Algorithm 2.
2.3.3 Limitation of VAMP

Despite many advantages of VAMP, its practical implementation for recovering large-scale data is limited due to the LMMSE estimator (2.103), which requires computing the inverse of an $N$ by $N$ matrix. Since this operation is of computational complexity $O(N^3)$, in [10] it was proposed to precompute the SVD of $A = USV^T$ and reformulate the LMMSE estimator as

$$g_A(x_{B\rightarrow A}^t, v_{B\rightarrow A}^t) = \left(\frac{1}{v_w}VS^T SV^T + \frac{1}{v_{B\rightarrow A}^t}I\right)^{-1}\left(\frac{1}{v_w}VS^T u_t y + \frac{1}{v_{B\rightarrow A}^t}x_{B\rightarrow A}^t\right)$$

$$= V\left(\frac{1}{v_w}S^T S + \frac{1}{v_{B\rightarrow A}^t}I\right)^{-1}\left(\frac{1}{v_w}S^T \tilde{y} + \frac{1}{v_{B\rightarrow A}^t}V^T x_{B\rightarrow A}^t\right), \quad (2.124)$$

where we defined $\tilde{y} = u_t y$, which can be precomputed and stored. In this form, the LMMSE estimator requires inverting a diagonal matrix and this operation has a complexity of order $O(N)$. However, now one would require storing the singular-vector matrix $V$ of dimension $M$ by $N$, which becomes intractable from the memory point of view when we consider large-dimensional inverse problems. Additionally, when we choose $A$ to be the FJLT to reduce the cost of matrix-vector products, the resulting singular-vector matrix $V$ does not possess such a fast implementation as pointed out in [94] and evaluation of (2.124) is of complexity $O(N^2)$. Therefore, to solve this computational/memory bottleneck, we need an alternative method of upscaling VAMP. This is one of the main focuses of this thesis and in the next section we review some initial ideas developed for solving this problem.

2.4 Upscaling VAMP and Conjugate Gradient VAMP

In this section we briefly cover one of the methods for upscaling VAMP and set up the direction of a major part of the research carried out in this thesis. The upscaling considered next consists of two steps. First, we reformulate Block A in VAMP using the Woodbury transformation as suggested in [85]. In this work it
was shown that one can equivalently update $x^t_{A\to B}$ as
\[
x^t_{A\to B} = x^t_{B\to A} - \frac{1}{\gamma^t_A} A^T \mu^t_A, \tag{2.125}
\]
where
\[
\mu^t_A = g_A(x^t_{B\to A}, \hat{v}^t_{B\to A}) = \left(v_w I_M + v^t_{B\to A} A A^T\right)^{-1}(y - A x^t_{B\to A}) \tag{2.126}
\]
is an alternative form of the LMMSE estimator (2.103). Additionally, in (2.125) the scalar $\gamma^t_A$ is an estimate of the divergence of the new function $g_A$,
\[
\gamma^t_A = \frac{1}{N} \nabla_{(x^t_{B\to A})} \cdot \left(A^T g_A(x^t_{B\to A}, \hat{v}^t_{B\to A})\right). \tag{2.127}
\]
By making this transformation, we move the most expensive computation of Block A – the matrix inverse – from the signal domain of dimension $N$ to the measurement domain of size $M = \delta N$, which is usually much smaller than $N$. Importantly, the two variants of Block A are statistically identical and produce the same vector $x^t_{A\to B}$. In particular, $h_t = x^t_{A\to B} - x$ remains to be zero-mean i.i.d. Gaussian with the variance $v_{h_t}$, which can be alternatively defined through [85]
\[
v^t_{A\to B} \overset{a.s.}{=} \gamma^t_A(v^t_{B\to A})^{-1} - v^t_{B\to A}, \tag{2.128}
\]
where
\[
\gamma^t_A(v) = \delta \int \frac{\lambda}{v_w + v\lambda} p(\lambda)d\lambda \tag{2.129}
\]
is the new sensitivity function. The SE of Block B remains the same as in VAMP.

Although $M$ might be substantially smaller than $N$, directly inverting an $M$ by $M$ matrix in (2.126) is still computationally challenging and we would like to further reduce the computational cost of the algorithm. Therefore, the second step in upscaling VAMP is to formulate the new LMMSE estimate $\mu^t_A$ as the solution to the system of linear equations (SLE),
\[
W_t \mu^t_A = z_t, \tag{2.130}
\]
where
\[
W_t = v_w I_M + v^t_{B\to A} A A^T, \tag{2.131}
\]
\[
z_t = y - A x^t_{B\to A} \tag{2.132}
\]
and use an iterative method for approximating the exact solution $\mu'_A$. While there are various iterative methods suitable for this task, for now we stick to the Conjugate Gradient (CG) algorithm used in the work [92] that originally considered this direction of upscaling VAMP.

On a general level, the CG algorithm iteratively minimizes the following quadratic cost function
\[
\mathcal{f}(\mu) = \frac{1}{2} \mu^T W \mu - z_1 \mu, \tag{2.133}
\]
where $W$ is assumed to be a positive-definite matrix of dimension $M$. By setting $\nabla \mathcal{f}_t(\mu) = W \mu - z = 0$, one can verify that $\mathcal{f}(\mu)$ is minimized by a vector $\mu = W^{-1} z$. The key elements of CG are the $W$-orthogonal vectors $p^i$ satisfying
\[
(p^i)^T W p^j = 0 \quad (2.134)
\]
for $i \neq j$. In CG these vectors serve as the search directions and are referred to as conjugate directions. At each iteration, CG generates a new conjugate direction $p^i$ and the associated step size $a^i$ to form a new approximation $\mu^i$ of $\mu$,
\[
\mu^i = \sum_{j=0}^{i} a^i p^j. \tag{2.135}
\]
Here, the step size $a^i$ is selected by minimizing the cost (2.133), while the new conjugate direction $p^i$ is obtained by applying the Gram-Schmidt Conjugation (GSC) process to the residual vector
\[
r^i = y - W \mu^{i-1}. \tag{2.136}
\]
The GSC process removes from $r^i$ all the components that are not $W$-orthogonal with all the previous conjugate direction $p^j$, $j < i$, i.e.
\[
p^i = r^i + \sum_{j=1}^{i-1} b^i_j p_j. \tag{2.137}
\]
In general, the conjugation process will be successful and a new conjugate direction $p^i$ will follow (2.134) if we choose $b^i_j = \frac{(r^i)^T W p_j}{(p_j)^T W p_j}$ [95]. However, one can verify [95] that for CG all the scalars $b^i_j$ are equal to zero except for $b^{i,i-1} = b^i$. Next, we briefly cover the main properties of CG and for that we define the
error vector

\[ \mathbf{e}^i = \boldsymbol{\mu}^i - \boldsymbol{\mu}. \]  

Then one can show [95, 96] that after each iteration \( i \), the CG algorithm minimizes the error \( E^i = (\mathbf{e}^i)^T W \mathbf{e}^i \) within the subspace \( \mathbf{e}^0 + \text{span}\{ \mathbf{p}^0, \mathbf{p}^1, \ldots, \mathbf{p}^i \} \), where \( \mathbf{e}^0 = \boldsymbol{\mu}^0 - \boldsymbol{\mu} \) is the initial error vector. Therefore, after \( M \) iterations the error \( E^i \) converges to zero since \( W \) is positive definite and has rank \( M \). However, the traditional way of applying CG is to use only a few iterations \( i \) of the algorithm to obtain a low-complexity approximation. In this thesis we follow this idea and define the Conjugate Gradient VAMP (CG-VAMP) that uses several iterations of CG from Algorithm 3 to approximate (2.130). Although applying such an approximation makes CG-VAMP have a different and worse set of fixed points compared to VAMP [13, 92], when \( i \) is small, the resulting Message Passing algorithm has a computational complexity close to AMP’s.

**Algorithm 3:** CG for approximating \( W\boldsymbol{\mu} = \mathbf{z} \)

1. **Initialization:** \( \boldsymbol{\mu} = 0_M, \mathbf{r}^0 = \mathbf{p}^0 = \mathbf{z}, i = 0 \)
2. while \( i < i_{\max} \) do
3. \( \mathbf{d}^i = W\mathbf{p}^i \)
4. \( a^i = \frac{\|r^i\|^2}{(\mathbf{p}^i)^T \mathbf{d}^i} \)
5. \( \boldsymbol{\mu}^{i+1} = \boldsymbol{\mu}^i + a^i \mathbf{p}^i \)
6. \( \mathbf{r}^{i+1} = \mathbf{r}^i - a^i \mathbf{d}^i \)
7. \( b^i = \frac{\|r^{i+1}\|^2}{\|r^i\|^2} \)
8. \( \mathbf{p}^{i+1} = \mathbf{r}^{i+1} + b^i \mathbf{p}^i = \mathbf{z} - W\boldsymbol{\mu}^{i+1} + b^i \mathbf{p}^i \)
9. \( i = i + 1 \)
10. **Output:** \( \boldsymbol{\mu}^{i+1}, \{a^i\}, \{b^i\}, \|r^i\|^2 \)

2.4.1 State Evolution and Onsager correction in CG-VAMP

Using a few iterations \( i \) of CG to approximate the LMMSE estimator (2.126) allows us to upscale the VAMP algorithm. However, such a change in the estimator \( g_A \) also requires adapting the other components of Block A in order to preserve the
main properties of VAMP, i.e. the existence of the SE and the Gaussianity of the error vector $h_t = x^t_{A\to B} - x$ at the input of the denoiser $g_B$. In [92], it was shown that these properties do hold in CG-VAMP under Assumptions B1-B3 as long as the key ingredient – the divergence $\gamma^i_A$ from (2.127) – is adapted accordingly. In that work, the authors showed that in the LSL, CG acts as a linear mapping, despite the traditional non-linear nature of this algorithm, and used this linear mapping to derive a closed-form solution for $\gamma^i_A$ in CG-VAMP. Specifically, it was shown that the CG output $\mu^{t,i}_A$ can be equivalently represented as

$$\mu^{t,i}_A = \sum_{j=0}^i \gamma^{t,i}_j \phi^j_z,$$

(2.139)

where $\phi = AA^T$, and in the LSL, $r^{t,i}_j[j]$ is a scalar function of only $v^i_{B\to A}$ and $v_w$ and its exact definition can be found in Appendix D, where we prove a more general result. Then the authors applied (2.139) to (2.127) to obtain the solution for $\gamma^i_A$

$$\lim_{N \to \infty} \gamma^{t,i}_A \overset{a.s.}{=} \lim_{N \to \infty} - \sum_{j=0}^i r^{t,i}_j[j] \chi_{j+1},$$

(2.140)

where we defined the $j$-th spectral moment

$$\chi_j = \frac{1}{N} \text{Tr}\{(S^TS)^j\}.$$

(2.141)

Similarly, the equivalent model (2.139) of CG was used to derive the new form of the SE of Block A that now reads as

$$v^{t,i}_{A\to B} \overset{a.s.}{=} (\gamma^i_A)^{-2} \sum_{j=0}^{i} \sum_{k=0}^{i} \left(r^{t,i}_j[k]\chi_{j+k+1}\right) - v^i_{B\to A}.$$ 

(2.142)

This expression together with the SE of Block B of VAMP defines the overall SE of CG-VAMP.

The resulting CG-VAMP algorithm with the correctly chosen Onsager correction for CG serves as a scalable alternative to VAMP and possesses most of the properties of the original algorithm. The resulting computational cost of Block A in CG-VAMP is of order $O(i\psi(N))$, where $\psi(N)$ defines the computational cost of a matrix-vector product with $A$, i.e. is proportional to that of AMP.
2.4.2 Drawbacks and stability issues of CG-VAMP

While the above implementation of CG-VAMP provides a scalable alternative to VAMP for solving large-scale CS inverse problems with ROI $A$, the current version of the algorithm has two problems. First, the divergence $\gamma_{t,i}^A$ of CG and the SE $v_{A\rightarrow B}^{t,i}$ assumes the access to the spectral moments $\chi_j$ of the measurement operator $A$. Unfortunately, in many practical cases, this assumptions cannot be ensured and one should substitute the exact moments $\chi_j$ in both (2.140) and (2.142) by their estimated versions $\hat{\chi}_j$. To construct such an estimate, one could refer to the results from [1, 97, 98], where it is suggested to generate a random vector $e \in \mathbb{R}^N$ whose entries are either $+1$ or $-1$ with an equal probability and evaluate the following sequence of matrix-vector products

$$e_j = \Phi e_{j-1}$$

(2.143)

with an initialization $e_0 = e$. Then, one can show that [1, 97, 98]

$$\hat{\chi}_j = \frac{1}{N}\|e_j\|^2$$

(2.144)

is an unbiased and consistent estimator of $\chi_j$,

$$\mathbb{E}\left[\frac{1}{N}\|e_j\|^2\right] = \chi_j.$$  

(2.145)

However, for that type of method, the generated estimate $\hat{\chi}_j$ has the variance proportional to $\chi_{2j}$ [97]. Since we assume the normalization $\frac{1}{N}\text{Tr}\{S^TS\} = \chi_1 = 1$, we have that

$$\chi_2 = \frac{1}{N}\text{Tr}\{(S^TS - \chi_1)^2\} + \chi_1 = \frac{1}{N}\text{Tr}\{(S^TS - \chi_1)^2\} + 1$$

is greater than 1 if $\kappa(A) > 1$. Then, from the Hölder’s inequality [99] we have

$$\chi_j \geq (\chi_2)^{\frac{j}{2}},$$

which implies that the variance $\chi_{2j}$ of the estimate (2.144) has exponential growth with $j$. As a result, the estimates $\hat{\chi}_j$ of the spectral moments might be highly inaccurate for a larger number of CG iterations $i$. Moreover, these large spectral moments scale directly and indirectly (through $v_{t,i}^{A\rightarrow B}$) the variances $v_w$ and $v_{B\rightarrow A}^{t,i}$. In practice we do not have the direct access to them and can only use their estimated versions $\hat{v}_w$ and $\hat{v}_{B\rightarrow A}^{t,i}$ that are corrupted by some estimation error. In the
estimators based on (2.140) and (2.142) this estimation error is further magnified by exponentially in $j$ growing moments $\chi_j$. Since the SE (2.142) uses the moments $\chi_j$ of order up to $i^2 + 2$, even a small number $i$ might potentially lead to instability of the estimator for $v_{A\rightarrow B}^{i\ell}$. In Chapter 3 we will demonstrate numerically the instability of CG-VAMP when (2.140) and (2.142) are used to estimate the parameters of Block A, and will develop alternative estimation tools that do not involve the moments $\chi_j$.

The second problem with CG-VAMP is that when one uses a few iterations of CG to approximate LMMSE, the resulting Message Passing algorithm might have a slower convergence rate and an inferior set of fixed points. This result follows from a few facts. First, it was recently shown in [13] that LMMSE is the optimal estimator for Block A. Next, we recall from Sections 2.3.1 and 2.4 that CG-VAMP has a 1D SE (the combination of (2.142) and (2.121)). Thus, one can follow the same reasoning as in [69] and show that using a suboptimal estimator $g_A$ instead of LMMSE leads to suboptimal convergence rate and set of fixed points of the resulting MP algorithm. In Chapter 3 we address the acceleration aspect of CG-VAMP and propose a combination of stopping criteria for CG that leads to faster time-wise convergence of the algorithm without sacrificing the estimation accuracy.

Then, in Chapter 4 we rigorously develop a piratical warm-starting scheme for CG that can be used within CG-VAMP to ensure that every fixed point of the resulting MP algorithm corresponds to a fixed point of VAMP. Importantly, this result is ensured with any non-zero number of CG iterations $i > 0$.

### 2.5 Conclusions

In this chapter we reviewed a series of methods used for solving CS inverse problems and set up the context for the remaining of the thesis. We begun with the classical CS methods that recover a signal from undersampled measurements under the assumption of sparsity or compressibility of the signal. We introduced the notion of RIP, discussed how it can be used to derive rigorous recovery guarantees and presented several algorithms for reconstructing sparse signals like IHT and ISTA. Next, we moved to the probabilistic approach for solving CS inverse problems and introduced the general EP framework for approximating the MMSE
estimator of the signal. Initially, we used this framework to present the first scalable version of EP – AMP, which is proved to recover the MMSE estimate of a signal, have a predictable dynamics, benefit from PnP denoisers and, importantly, is a first-order method. However, these properties only hold for sub-Gaussian matrices and to extend the range of inverse problems, we referred to the next variant of EP – VAMP. VAMP preserves most properties of AMP while being able to handle ill-conditioned operators $A$ with relatively little randomness. Yet, its practical applicability is limited to relatively small dimensional problems because each iteration of VAMP requires computing the expensive LMMSE estimator. As a result, we introduced the upscaled version of VAMP – CG-VAMP where the LMMSE estimator is approximated with a few iterations $i$ of CG. Theoretically, CG-VAMP preserves most properties of VAMP while having the computational complexity similar to AMP.

However, the current version of CG-VAMP has several problems. First, it might struggle from stability issues when the number of CG iterations $i$ is even moderately large, which limits the approximation potential of CG-VAMP. Second, there are no practical strategies for choosing $i$ at each outer-loop iteration to obtain the faster convergence and better accuracy at the same time. Lastly, CG-VAMP does not utilize the information generated at the previous outer-loop iterations to operate at its full potential. These three problems are considered in the next two chapters.
Chapter 3

Stable and efficient
implementation of CG-VAMP

In the previous chapter we introduced the Vector Approximate Message Passing algorithm that is conjectured to be Bayes optimal (when $g_B$ is an MMSE denoiser) and can operate with a wider class of random matrices compared to the original Approximate Message Passing method. However, the applicability of VAMP was limited to relatively small dimensional inverse-problems due to intractability of the LMMSE estimator (2.126). To upscale VAMP, we introduced the Conjugate Gradient algorithm that uses a few iterations $i$ to approximate LMMSE at every outer-loop iteration $t$. The resulting Conjugate Gradient VAMP algorithm can have the computational and memory complexities similar to those of AMP, but the available tools for implementing CG-VAMP in practice make the algorithm unstable when $i$ is larger than a few dozens and the oracle information about the singular spectrum of $A$ is not available. Besides that, there is still an open question on how to choose the number of CG iterations $i$ at each outer-loop iteration $t$ in order to generate an accurate estimate of $x$ and keep a low computational cost of the overall algorithm at the same time. In this chapter we present efficient methods for stabilizing CG-VAMP to allow CG to use by an order more iterations than it was possible before, and propose several strategies for adaptively choosing the number of CG iterations to obtain fast time-wise convergence without sacrificing the accuracy of estimation.
We begin with developing stable methods for estimating the CG’s divergence $\gamma_{t,i}^{A}$ used to form the Onsager correction, and an alternative LSL model for $v_{A\rightarrow B}^{t,i}$ in CG-VAMP. Recall that here we consider CG-VAMP that can be summarized in the following two steps

$$x_{t}^{A\rightarrow B} = x_{t}^{B\rightarrow A} - \frac{1}{\gamma_{t}^{A}} A_{t}^{T} g_{A}^{i}(x_{t}^{B\rightarrow A}, \hat{v}_{B\rightarrow A}^{t}) \quad (3.1)$$

$$x_{t+1}^{B\rightarrow A} = \frac{1}{1 - \gamma_{t+1}^{B}} \left( g_{B}(x_{t}^{A\rightarrow B}, \hat{v}_{A\rightarrow B}^{t}) - \gamma_{t+1}^{B} x_{t}^{A\rightarrow B} \right) \quad (3.2)$$

where $g_{A}^{i}$ is the CG algorithm with $i$ iterations approximating $2.130$, $g_{B}$ is a denoiser, and

$$\gamma_{t,i}^{A} = \frac{1}{N} \nabla(x_{t}^{B\rightarrow A}) \cdot \left( A_{t}^{T} g_{A}^{i}(x_{t}^{B\rightarrow A}, \hat{v}_{B\rightarrow A}^{t}) \right) \quad (3.3)$$

$$\gamma_{t}^{B} = \frac{1}{N} \nabla(x_{t}^{A\rightarrow B}) \cdot g_{B}(x_{t}^{A\rightarrow B}, \hat{v}_{A\rightarrow B}^{t}) \quad (3.4)$$

are the respective divergences.

### 3.1 Practical estimation of $\gamma_{t,i}^{A}$ in CG-VAMP

To the best of our knowledge, in all of the works on MP where a non-trivial approximating function $g_{A}$ is used, the divergence $\gamma_{A}^{t,i}$ was estimated in one of two ways. In the first case [10, 14, 55, 79, 85, 92, 93, 100, 101], one derives the closed-form solution for $\gamma_{A}^{t,i}$ based on the definition $2.127$ (or, equivalently, defined as in Algorithm 2) and under the LSL assumption. The resulting expression will be a function of spectral moments $\{\chi_{j}\}_{j=1}^{j_{\text{max}}}$ up to some order $j_{\text{max}}$, and of the statistical parameters like $v_{w}$ and $v_{B\rightarrow A}^{t}$. Then, the estimator is constructed by using this expression and replacing all the oracle parameters by the estimated quantities, including the estimates of the spectral moments $\hat{\chi}_{j}$. Unfortunately, when the maximum order $j_{\text{max}}$ is large, using such an estimator within an MP algorithm might lead to severe stability issues.

The second approach [94] is also based on treating $\gamma_{A}^{t,i}$ as the divergence of $g_{B}$, but in this case, it is estimated using the Black-Box Monte Carlo method [1]. The drawback of this method is that it requires an additional execution of the function $g_{A}$ and this, effectively, leads to doubling the computational cost of Block A.
In this thesis, we leverage the LSL properties of MP algorithms and take a different perspective on the scalar $\gamma^t_A$ that leads to estimators that neither explicitly use the spectral moments $\chi_j$ nor require additional executions of $g_A$. Our idea is built around the following LSL result.

**Lemma 1.** Consider the CG-VAMP (3.1)-(3.4). Under Assumptions B1-B3 from page 42, the scalar $\gamma^{t,i}_A$ is almost surely equal to

$$\lim_{N \to \infty} \gamma^{t,i}_A \text{ a.s.} = \lim_{N \to \infty} \frac{1}{N} q^T_i A^T g^i_A(x^t_{B \to A}, \hat{v}^t_{B \to A}),$$

(3.5)

where $q_i$ is the error vector $q_i = x^t_{B \to A} - x$.

**Proof.** See Appendix B. \hfill\qed

Although the asymptotic result (3.5) cannot be used directly because of the explicit reliance on the error vector $q_i$, this identity becomes handy when we consider it together with the following reformulation of $z_t$ from (2.132)

$$z_t = y - Ax^t_{B \to A} = Ax + w - Ax = w - Aq_t,$$

(3.6)

where we used the definitions (2.1) of $y$. Combining (3.5) together with (3.6) gives

$$\lim_{N \to \infty} \gamma^{t,i}_A \text{ a.s.} = \lim_{N \to \infty} \frac{1}{N} w^T \mu^{t,i}_A - \frac{1}{N} z^T_t \mu^{t,i}_A.$$  

(3.7)

This result still cannot be used directly, since it is explicitly stated in terms of the measurement noise $w$. However, now we can exploit the random properties of $w$ to define a recursive model that asymptotically predicts the behaviour of the inner-product $\frac{1}{N} w^T \mu^{t,i}_A$. This is formally stated in the following theorem.

**Theorem 2.** Consider the CG-VAMP (3.1)-(3.4) and let $\mu^{t,i}_A = g^i_A(x^t_{B \to A}, \hat{v}^t_{B \to A})$ be the output of the zero-initialized CG from Algorithm 3 after $i$ inner-loop iterations. For $j = 0, \ldots, i$, let $a^j_t$ and $b^j_t$ be as in Algorithm 3. Define a sequence of scalars $\eta^j_t$ as

$$\eta^{j+1}_t = v_w \left( \delta - \frac{1}{N} z^T_t \mu^{t,j+1}_A \right) + b^j_t \eta^j_t$$

(3.8)

with $\eta^0_t = \delta v_w$ and $\delta = \frac{M}{N}$. Then, under Assumptions B1-B3 from page 42, we
have that
\[
\lim_{N \to \infty} \frac{1}{N} w^T \mu_A^{t,i} \text{ a.s.} = \lim_{N \to \infty} \sum_{j=0}^{i-1} a_j^i \eta_j^i.
\] (3.9)

Proof. See Appendix B.

With this theorem, one can define an estimator for \( \gamma_{t,i}^A \) as
\[
\hat{\gamma}_{t,i}^A = \frac{\sum_{j=0}^{i-1} a_j^i \hat{\eta}_j^i - \frac{1}{N} z_t^T \mu_A^{t,i}}{v_{B \to A}^t},
\] (3.10)
where \( \hat{\eta}_j^i \) is computed from (3.8) with \( v_w \) replaced by its estimated value \( \hat{v}_w \). Importantly, (3.10) does not rely on the knowledge of \( \chi_j \) contrary to (2.140) and can be naturally implemented in the body of CG to update the estimate of the divergence \( \gamma_{t,i}^A \) after each iteration. Additionally, the computational cost of (3.10) is dominated by evaluating one inner-product of \( M \)-dimensional vectors and is of order \( O(M) \), which makes the cost of the estimator negligible compared to the cost of the CG routine. Similarly, the estimator requires storing only several scalar values and, therefore, the memory costs are negligible as well. We will show numerically that the range of \( i \) for which the estimator (3.10) demonstrates stable and accurate performance is by an order of magnitude larger than for the estimator based on (2.140).

### 3.2 Practical estimation of \( v_{t,i}^{A \to B} \) in CG-VAMP

Next, we propose an asymptotically consistent and stable estimator for the variance \( v_{t,i}^{A \to B} \) that can be naturally implemented within CG and has negligible computational and memory costs. For this, we refer to the asymptotic model (2.115) and use the Strong Law of Large Numbers, Lemma 5 in Appendix A, to show that
Stable and efficient implementation of CG-VAMP

$v_{A \rightarrow B}^{t,i}$ is asymptotically equivalent to

$$v_{A \rightarrow B}^{t,i} \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} \|h_t\| = \lim_{N \to \infty} \frac{1}{N} \|x_{A \rightarrow B}^{t,i} - x\|$$

$$= \lim_{N \to \infty} \frac{1}{N} \|q_t - (\gamma_{A}^{t,i})^{-1}A^{T} \mu_{A}^{t,i}\|^2$$

$$= \lim_{N \to \infty} \frac{1}{N} \left(\|q_t\|^2 - 2(\gamma_{A}^{t,i})^{-1}q_t^{T}A^{T} \mu_{A}^{t,i} + (\gamma_{A}^{t,i})^{-2} \frac{1}{N} \|A^{T} \mu_{A}^{t,i}\|^2\right)$$

$$\overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} \left(\gamma_{A}^{t,i}\right)^{-2} \|A^{T} \mu_{A}^{t,i}\|^2 - v_{B \rightarrow A}^{t,i}, \quad (3.11)$$

where in (a) we used (3.1) and in the last step we used Lemma 1. Although this result can be used directly by substituting in the estimated values $\hat{\gamma}_{A}^{t,i}$ and $\hat{v}_{B \rightarrow A}^{t,i}$ instead of their true versions, executing this estimator after each CG iteration $i$ would double the cost of computing the update for $x_{A \rightarrow B}^{t,i}$ due to the additional matrix-vector product $A^{T} \mu_{A}^{t,i}$. Therefore, next we reformulate (3.11) to obtain an estimator for $v_{A \rightarrow B}^{t,i}$ that utilizes the results computed in the CG routine to avoid additional matrix-vector products.

**Theorem 3.** Consider the CG-VAMP (3.1)-(3.4) and let $\mu_{A}^{t,i} = g_{A}(x_{B \rightarrow A}^{t,i}, \hat{v}_{B \rightarrow A}^{t,i})$ be the output of the CG algorithm from Algorithm 3 and $\gamma_{A}^{t,i}$ be as in (3.3). Additionally, let $\hat{v}_{B \rightarrow A}^{t,i}$ and $\hat{v}_{w}$ be the respective estimates of $v_{B \rightarrow A}^{t,i}$ and $v_{w}$ used to form the matrix $W_{t} = \hat{v}_{w}1 + \hat{v}_{B \rightarrow A}^{t,i}A^{T}A$ in the SLE (2.130). Then, under Assumptions B1-B3 from page 42, we have that

$$v_{A \rightarrow B}^{t,i} \overset{a.s.}{=} \lim_{N \to \infty} \frac{\zeta_{t} - \hat{v}_{w} \frac{1}{N} \|\mu_{A}^{t,i}\|^2}{\hat{v}_{B \rightarrow A}^{t,i} \left(\gamma_{A}^{t,i}\right)^{2}} - v_{B \rightarrow A}^{t,i}, \quad (3.12)$$

where the scalar $\zeta_{t}$ is iteratively defined as

$$\zeta_{t} = \zeta_{t-1}^{i} + a_{t}^{i} \frac{1}{N} \|r_{t}^{i}\|^2$$

with $\zeta_{0} = 0$ and $r_{t}^{i}$ and $a_{t}^{i}$ are as in Algorithm 3.

**Proof.** See Appendix C. $\blacksquare$

Thus we propose the following estimator of $v_{A \rightarrow B}^{t,i}$

$$\hat{v}_{A \rightarrow B}^{t,i} = \frac{\zeta_{t} - \hat{v}_{w} \frac{1}{N} \|\mu_{A}^{t,i}\|^2}{\hat{v}_{B \rightarrow A}^{t,i} \left(\gamma_{A}^{t,i}\right)^{2}} - \hat{v}_{B \rightarrow A}^{t,i}, \quad (3.14)$$
where instead of the true values of $\gamma_{t,i}^A$ and of $v_{B\rightarrow A}^t$ we used their estimated versions. The estimator (3.14) can be efficiently implemented in the body of CG and computed at each iteration at the additional cost of $O(M)$ operations and negligible memory costs provided an estimate of $\gamma_{t,i}^A$ is available.

### 3.3 Adaptive CG in CG-VAMP

The developed tools in the previous section allow one to efficiently estimate the scalar $\gamma_{t,i}^A$ used to form the Onsager correction and the SE for $v_{A\rightarrow B}^{t,i}$ after each CG iteration. Yet, there is still a question on how many of those CG iterations we should do at each iteration $t$ in order to obtain consistent progression and efficient performance of CG-VAMP. Clearly, if we did $i = M$ iterations of CG, we would be guaranteed to recover the exact LMMSE estimator, which is the optimal estimator $g_A$ used in (3.1) as was recently shown in [13]. Hence, a natural bound on the SE within CG-VAMP would be

$$v_{A\rightarrow B}^{t,i} \geq v_{A\rightarrow B}^{t,M}$$  \hspace{1cm} (3.15)

for $i \leq M$. The first consequence of this inequality is that CG-VAMP has a lower convergence rate (in terms of outer-loop iterations $t$) in comparison to VAMP and this is usually a reasonable compromise. However, this inequality also means that CG-VAMP may converge to a different and worse fixed point that is far from the VAMP’s fixed point. Moreover, if we choose insufficient number of iterations $i$ and CG generates a too rough approximation of LMMSE, then $v_{A\rightarrow B}^{t,i}$ might even increase with respect to $v_{A\rightarrow B}^{t-1,i}$ from the previous iteration. In the simulations demonstrated shortly, we will observe that the number of the CG iterations required to achieve a desired reduction of $v_{A\rightarrow B}^{t,i}$ per-outer-loop iteration significantly changes with $t$. This, therefore, motivates the benefit in the adaptive choice of $i[t]$ at each $t$.

To achieve a monotonic reduction of the variance $v_{A\rightarrow B}^{t,i}$ after each outer-loop iteration $t$, while keeping $i$ as small as possible, we can iterate the CG algorithm until the following inequality is met

$$\hat{v}_{A\rightarrow B}^{t,i} < c\hat{v}_{A\rightarrow B}^{t-1,i[t-1]}$$, \hspace{1cm} (3.16)
where the constant $c < 1$ defines the expected per-outer-loop iteration reduction of the variance $v^{t,i}_{A \rightarrow B}$. However, following the rule (3.16) ensures only the monotonicity of the SE and does not ensure the efficiency of CG-VAMP. To cover the later aspect of the algorithm, we need to consider not only the outer-loop dynamics, but also the inner-loop ones. From the numerical experiments we observe that the relative improvement of $v^{t,i}_{A \rightarrow B}$ after every next CG iteration $i$ decreases and eventually becomes negligible. Thus, to avoid inefficient operation of Block A, we might want to terminate CG earlier. On the other hand, the first CG iterations provide the most reduction in terms of the SE and terminating CG too early means that we miss the opportunity to cost-effectively lower the intrinsic variance. To ensure CG is operating efficiently, we suggest using the stopping criteria (3.16) in combination with the following rule: compare the relative expected improvement of $v^{t,i}_{A \rightarrow B}$ after each iteration $i$ and continue iterating CG if

$$\frac{\hat{v}^{t-1,i}_{A \rightarrow B} - \hat{v}^{t,i}_{A \rightarrow B}}{\hat{v}^{t,i}_{A \rightarrow B}} > \Delta,$$

(3.17)

where $\Delta$ is some positive scalar. This inequality suggests that if the last inner-loop iteration was efficient enough, which is defined by the hyperparameter $\Delta$, then we should continue iterating the CG algorithm even if the minimum criteria (3.16) is already met. The choice of the constant $\Delta$ depends on the parameters of the measurement system (2.37) and the chosen denoiser $g_B$, and should be adapted for different applications individually. Lastly, we suggest to bound $i$ by some $i_{max}$ in the case if we chose too ambitious a reduction level $c$, which is not achievable even for VAMP or within a reasonable number of CG iterations.

These stopping criteria together with the developed tools from the previous section are used to construct an Adapted Conjugate Gradient (ACG) shown in Algorithm 4. The first block of the algorithm implements the function $CG$ that represents the $i$-th iteration steps of CG from Algorithm 3 and outputs the vector $\mu_t^{t,i}$ and the data required for estimating $\gamma_t^{t,i}$ and $v_t^{t,i}_{A \rightarrow B}$. The second block uses the ideas from Section 3.1 to construct an estimate of $\gamma_t^{t,i}$ used to form the Onsager correction for CG and passed to the third block where an estimate $\hat{v}^{t}_{A \rightarrow B}$ is computed to implement the stopping criteria (3.16) and (3.17). The computational complexity per outer-loop iteration of ACG is dominated by that of the CG rou-
Stable and efficient implementation of CG-VAMP
time where we need to compute the matrix-vector product \( d^t_i = W^t p^t_i \), which is of complexity \( O(N \log N) \) or even smaller if \( A \) is a fast operator.

**Algorithm 4: Adaptive Conjugate Gradient**

<table>
<thead>
<tr>
<th>Initialization: ( \mu^{(0)}_A = 0, \mu^{(0)}_t = p^0_t = z^0_t, \nu^{(0)}<em>t = \delta \hat{v}^0</em>\nu, \zeta^{(0)}_t = 0, \hat{\gamma}^{(0)}_A = \infty, i = 0 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>while ( i &lt; i_{\text{max}} ) and ( \Delta(i) &gt; \Delta ) or ( \hat{\gamma}^{(i)}_{A \rightarrow B} ) do</td>
</tr>
<tr>
<td>1. Standard CG routine</td>
</tr>
<tr>
<td>2. Divergence of CG</td>
</tr>
<tr>
<td>3. SE of Block A</td>
</tr>
<tr>
<td>4. Output: ( \mu^{(i+1)}_A, \hat{\gamma}^{(i+1)}_A )</td>
</tr>
</tbody>
</table>

3.4 Simulation experiments

In this section we numerically compare the image reconstruction performance of CG-VAMP that utilizes the methods for estimating \( \gamma^{(l)}_A \) and \( v^{(l)}_{A \rightarrow B} \) from Section 2.4.1 against CG-VAMP with the proposed estimators (3.10) and (3.14). Additionally, we demonstrate the advantage of using ACG over regular CG with a fixed number of iterations within CG-VAMP. These algorithms are used to recover a natural image ‘man’ of dimension 1024 by 1024 shown in Figure 2.1 (a) measured by a Fast ill-conditioned Johnson Lindenstrauss Transform \( A = \text{SPHD} \) discussed at the end of Section 2.3.1. Besides the advantages of an FIJLT operator mentioned...
before, it also allows implementing VAMP for comparison, since
\[ AA^T = \text{SPHDD}^T \text{H}^T \text{P}^T \text{S}^T = \text{SS}^T \] (3.18)
and, therefore, the matrix inverse
\[ W_i^{-1} = (v_w \text{I}_M + v_{B \rightarrow A} AA^T)^{-1} = (v_w \text{I}_M + v_{B \rightarrow A} \text{SS}^T)^{-1} \]
used in the LMMSE (2.126) requires inverting only a diagonal matrix. However, here we emphasize that the CG-VAMP algorithms, and any other algorithms in this thesis besides VAMP, do not utilize the fact that \( W_i \) is diagonal and operate as if \( W \) is an arbitrary matrix. In all the experiments, we design the singular value matrix \( S \) to have singular values \( s_j \) following a geometric progression \( s_j / s_{j+1} = \rho \) with a constant \( \rho \) [10, 58] chosen to achieve the condition number \( \kappa(A) = 1000 \), unless stated otherwise. We choose the subsampling factor \( \delta = \frac{M}{N} = 0.05 \) and set the measurement noise variance \( v_w \) to achieve SNR \( \| x \|_2 / \| w \|_2 \) of 40dB. Lastly, in all the experiments, we estimate the divergence \( \gamma_B^t \) from (3.4) of the denoiser \( g_B \) using the Black-Box Monte Carlo (BB-MC). To construct an estimate, the BB-MC method executes the denoiser again \( g_B(x_{A \rightarrow B}^t + \epsilon e) \) with the original input \( x_{A \rightarrow B}^t \) perturbed by a random vector \( \epsilon e \), where each entry of \( e \) takes the values of \(-1 \) or \(+1 \) with equal probability, and we choose the scalar \( \epsilon \) similar to how it is chosen in the GAMP library\(^2\)
\[ \epsilon = \epsilon_0 \min \left( v_{A \rightarrow B}^t, \frac{1}{N} \| x_{A \rightarrow B}^t \|_1 \right) + \xi, \] (3.19)
where \( \xi \) is the the floating point precision in MATLAB. The difference between our implementation and how it is done in the GAMP library is in the chosen scalar \( \epsilon_0 \). In the GAMP library it is set to \( \epsilon_0 = 0.1 \), while we choose it depending on the denoiser and which scalar has led to a better performance of the MP algorithms. For example, for BM3D we set \( \epsilon_0 = 0.1 \), for DnCNN it is \( \epsilon_0 = 0.01 \), while for NLM we chose \( \epsilon = 0.001 \). We discuss this aspect in more details in Chapter 5.

All the experiments in this thesis are implemented on the platform described\(^1\)

\(^1\)One can verify [95] that CG converges to the exact solution in \( i = \min(M, k) \) iterations, where \( k \) is the number of distinct eigenvalues of \( W_i \). By choosing the singular values based on the geometric progression, we ensure that there are \( M \) distinct singular values, which is the hardest case.

\(^2\)The library is available at https://sourceforge.net/projects/gampmatlab/
in Table 3.1

Table 3.1: Software and hardware characteristics of the computational platform

<table>
<thead>
<tr>
<th>Software</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Operational system</td>
<td>Microsoft Windows 10</td>
</tr>
<tr>
<td>Computing Environment</td>
<td>MATLAB 2021a</td>
</tr>
<tr>
<td>Acceleration platform</td>
<td>Nvidia CUDA</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hardware</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>Intel i7-8086K CPU 4.00GHz</td>
</tr>
<tr>
<td>RAM</td>
<td>32 GB</td>
</tr>
<tr>
<td>GPU</td>
<td>Nvidia GeForce GTX 1080</td>
</tr>
</tbody>
</table>

3.4.1 Comparison of different CG-VAMP algorithms

In our first experiment, we numerically study the stability of CG-VAMP where $\gamma_{t,i}^A$ and $v^t_{A\rightarrow B}$ are estimated based on (2.140) and (2.142) respectively. We refer to this algorithm as CG-VAMP A. We compare CG-VAMP A against CG-VAMP with the proposed correction method (3.10) and the variance $v^t_{A\rightarrow B}$ estimator based on (3.14), which will be referred as CG-VAMP B. Here, both of the algorithms have the same update of $x^t_{B\rightarrow A}$ from (3.2), where we use the DnCNN denoiser from Section 2.2.5 and the BB-MC divergence estimator. In the experiment, the CG-VAMP algorithms are not provided with the oracle information about the singular spectrum $S$ of $A$ and for both (2.140) and (2.142) we estimate the moments $\chi_j$ by averaging over 1000 Monte-Carlo trials as discussed in Section 2.4.2. To compare the performance, we compute the oracle Normalized Mean Squared Error (NMSE) $\frac{\|g_B(x^t_{A\rightarrow B}) - x\|_2^2}{\|x\|_2^2}$ of the algorithms.

The NMSE of CG-VAMP A and CG-VAMP B with different numbers of CG iterations $i$ averaged over 10 realizations is shown in Figure 3.1, where we do not plot those algorithms that diverge immediately, as in the case of CG-VAMP A that becomes unstable for $i = 15$ and above. All the displayed curves experience negligible stochastic variations and stay almost identical between realizations. As seen from the plot, the practical version of CG-VAMP B remains stable for a wide range of inner-loop iterations and provides a progressively better estimate as $i$ increases. These algorithms converge to their respective fixed points at $t = 6$ and...
Stable and efficient implementation of CG-VAMP

Figure 3.1: NMSE versus outer-loop iteration number \( t \) for different types of CG-VAMP algorithms.

Table 3.2 demonstrates the average time taken by the algorithms to get to this iteration. We do not mention the time taken by VAMP since the algorithm is feasible only because of the specific structure of \( A \), while for a general \( A \), it is not feasible to run VAMP even for a single iteration for such a high-dimensional inverse problem. However, one can use the practical CG-VAMP B with a large \( i \) to reliably obtain almost arbitrary close performance, NMSE-wise, to that of VAMP.

Table 3.2: Time (seconds) taken for CG-VAMP B with \( i = \{5, 10, 30, 200\} \) iterations to converge to their respective fixed points

<table>
<thead>
<tr>
<th>Algorithm, ( i )</th>
<th>Time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CG-VAMP B, ( i = 5 )</td>
<td>10.01</td>
</tr>
<tr>
<td>CG-VAMP B, ( i = 10 )</td>
<td>11.76</td>
</tr>
<tr>
<td>CG-VAMP B, ( i = 30 )</td>
<td>18.6</td>
</tr>
<tr>
<td>CG-VAMP B, ( i = 200 )</td>
<td>82.61</td>
</tr>
</tbody>
</table>

3.4.2 CG-VAMP with Adaptive CG

In the next experiment we demonstrate the impact of different strategies of choosing the number of CG iterations \( i \) on the convergence properties of CG-VAMP. Consider the same CG-VAMP algorithm as in the previous experiment but with a NLM denoiser\(^3\) and set the condition number of \( A \) to \( 5 \times 10^4 \). To assess the

\(^3\)In this work we use the built-in implementation of NLM in MATLAB 2019
effect of including the stopping criteria (3.16) and (3.17), we replace regular CG by ACG from Algorithm 4 and choose the expected variance $v_{A \rightarrow B}^{t,i}$ reduction constant $c = 0.9$, the threshold for the normalized reduction of the variance $v_{A \rightarrow B}^{t,i}$ per inner-loop iteration $\Delta = 0.005$ and the maximum number of ACG iterations $i_{max} = 100$. In this setting, we aim to demonstrate that when the algorithm progresses and reduces its intrinsic variance, the number of CG iterations required to preserve the same level of per-iteration improvement changes. To do this, we plotted $i[t]$ versus $t$ in Figure 3.2, where the red curve represents ACG with only the stopping criterion (3.16), while (3.17) is effectively disabled by setting $\Delta = 10^{10}$. The blue curve on the same plot represents $i[t]$ that follows both stopping criteria (3.16) and (3.17) with $\Delta = 0.003$. Next, we considered the time-wise efficiency of three strategies: 1) we use both stopping criteria; 2) we use only the stopping criteria (3.16); 3) we do not use any stopping criteria apart from terminating when the number of ACG iterations reaches its maximum $i_{max} = 100$. In Figure 3.3 we plotted the NMSE of the three algorithms together with the computational time that algorithms needed to converge to iteration $t$. As expected, CG-VAMP with the strategy where we do maximum number of CG iterations at every outer-loop iteration (orange curve) converges the fastest iteration-wise. Yet, this approach loses time-wise to CG-VAMP with ACG with both stopping criteria by roughly 15% although the later takes an additional iteration to converge. This is explained by the fact that using the two stopping criteria allows ACG to stop when it becomes inefficient to run additional iterations and proceed to the denoising step instead. As seen from Figure (3.2), this variant of CG-VAMP uses only 70% of the allocated computational resources up until the convergence point. Lastly, the CG-VAMP algorithm with a single stopping criteria (3.16) converges slowest both iteration- and time-wise, as using only this stopping criteria leads to an early stop and puts most of the work during the early outer-loop iterations on the denoising step. Here we would like to emphasize that the time-wise improvement of adding the second stopping criterion (3.17) becomes more significant when the denoising part of CG-VAMP becomes cheaper. In Chapter 5 we develop an alternative to the BB-MC divergence estimation method that reduces the computational cost of the denoising block in most MP algorithms by almost a factor of 2, which makes adding the second stopping criterion even more advantageous (see the simulations.
Stable and efficient implementation of CG-VAMP

Figure 3.2: The number of CG iterations $i[t]$ that follow only the stopping criterion (3.16) (red curve) and that follows both (3.16) and (3.17) (blue curve).

3.5 Conclusions

In this chapter we proposed a combination of tools that substantially extended the reconstruction capabilities and efficiency of CG-VAMP, and created the theoretical basis for further results. First, we addressed one of the main issues with the available method for implementing CG-VAMP in practice. This is the stability of the algorithm when even a moderately large number of CG iterations $i$ is used to approximate the LMMSE. One of the sources of such instability is the available methods for estimating the SE of $v_{t,i}^{A \rightarrow B}$ and the divergence scalar $\gamma_{t,i}^{A}$ involved in the Onsager correction $\gamma_{A}^{t,i}x_{B \rightarrow A}$. To solve this problem, we referred to an alternative definition of the scalar $\gamma_{A}^{t,i}$ and rigorously developed an iterative method for constructing an estimate $\tilde{\gamma}_{A}^{t,i}$. The resulting method has a negligible computational and memory costs and can be naturally implemented within CG to update an estimate after each CG iteration. Similarly, we developed a robust iterative method for constructing an estimate of the intrinsic variance $v_{t,i}^{A \rightarrow B}$ with negligible computational and memory costs. We numerically demonstrated that when CG-VAMP utilizes the proposed estimators, the resulting algorithm can use an order of magnitude more CG iterations while remaining stable and obtain an
almost arbitrary close performance to that of VAMP. Next, we discussed different strategies for choosing the number of CG iterations at every outer-loop iteration of CG-VAMP in order to obtain the fastest time-wise convergence without sacrificing the accuracy of the final estimate. To implement those ideas in practice, we used the previously developed methods for estimating $\gamma_{t,i}^A$ and $\nu_{t,i}^{A \rightarrow B}$, and developed the Adaptive CG that can choose such a number of inner-loop iterations at each $t$ that provides efficient operation of CG-VAMP. To illustrate the impact of the proposed rules for adaptively choosing $i$, we conducted a series of experiments confirming the improved efficiency of CG-VAMP with ACG against CG-VAMP with regular CG. However, in the next chapter we will show that a CG-VAMP-like algorithm is capable of achieving the accuracy of VAMP while using any $i > 0$ inner-loop iterations and keeping the computational cost as low as that of AMP.
Chapter 4

Warm-starting in MP

In the previous chapter we conducted a numerical experiment where we showed that a practical implementation of CG-VAMP can get very close to the optimal performance of VAMP by using hundreds of CG iterations at each outer-loop iteration. Unfortunately, such an approach dramatically increased the algorithm’s computational cost compared to the case where we use only a few iterations of CG, which, however, leads to a much worse reconstruction performance. A natural question arises: is it possible to design an iterative method similar to CG that can be used within an MP algorithm to approximate the LMMSE estimator with only a few iterations and yet ensure that such an MP and VAMP would have the same set of fixed points? In this chapter we provably give a positive answer to this question by, first, extending regular CG from Algorithm 3 in Chapter 2 to the Warm-Started Conjugate Gradient (WS-CG) that utilizes a specific initialisation scheme based on the information generated previously in the MP algorithm. Additionally, we rigorously develop a new set of Onsager correction terms for WS-CG that ensure all the main asymptotic properties of the resulting MP algorithm, including the existence of a new SE that we rigorously derive. Next, we extend the idea of warm-starting and propose a unified iterative approximation framework that includes WS-CG as a special case, as well as Warm-Started Gradient Descent (WS-GD), besides other possible approximation methods. We prove that when an MP algorithm uses an instance of this approximation framework, the set of fixed points of such an MP is the same as that of VAMP and this result is invariant
to the number of iterations of this approximation framework. However, compared to other available first-order MP algorithms achieving VAMP’s fixed points [14, 55, 63, 93, 100], the proposed framework allows one to choose how much work (i.e. number of inner-loop iterations) to do at Block A in order to achieve a more reliable and faster time-wise convergence. Lastly, we rigorously show that the key element – the method for approximating LMMSE – of the recently proposed MP algorithm called Memory AMP (MAMP) corresponds to a particular instance of our unified framework. We conclude this chapter by numerical experiments demonstrating the improved dynamics of MP algorithms utilizing the proposed warm-started framework over MP algorithms that do not reuse the information generated at previous iterations.

4.1 Warm-starting the CG algorithm in CG-VAMP

In the previous chapter, we considered a CG-VAMP algorithm, where at each outer-loop iteration we implemented a zero-initialized CG to approximate the intractable LMMSE estimator (2.126). As pointed out in Section 3.3, the number of CG iterations plays a crucial role in the reconstruction potential of CG-VAMP and using too few of those iterations might even lead to an increase of the intrinsic variance in the algorithm. In other words, when we initialize CG with a zero-vector, first, we need to iterate CG a sufficient number of times to catch up with the accuracy from the previous outer-loop iteration \( t - 1 \), and only the remaining iterations of CG provide the resulting progress of the algorithm at iteration \( t \). Consequently, if we do not iterate a sufficient number of times, we will end up with a worse estimate than at the previous outer-loop iteration.

In this chapter we develop methods for overcoming this inefficient operation of CG-VAMP by referring to a concept that is common for certain optimization methods such as Sequential Quadratic Programming [102] and Interior-Point methods [103] – warm-starting. In the CG-VAMP algorithm, at each outer-loop iteration we have to approximate a solution to the updated SLE \( W_t \mu_t = z_t \) and a natural question arises: how different are these SLE at two consecutive iterations? If the
pair of the SLE are close, then due to asymptotic linearity of the CG algorithm from (2.139), the outputs $\mu^t_A$ and $\mu^{t-1}_A$ should be close as well. Then it would be natural to consider initializing the CG algorithm with "an approximation" of its expected output – the output from the previous outer-loop iteration. In particular, we propose to initialize the starting CG approximation $\mu_A^{t,0}$ as
\begin{equation}
\mu_A^{t,0} = \mu_A^{t-1,i}
\end{equation}
and the conjugate direction vector $p_t^i$ as
\begin{equation}
p_t^i = z_t - W_t \mu_A^{t-1,i} + b_{t-1}^{-1} p_{t-1}^{i-1}.
\end{equation}
Here, we do not use the initialization $p_t^0 = p_{t-1}^i$ because the residual term $r_t^i = z_t - W_t \mu_A^{t-1,i}$ is not the same as $r_{t-1}^i$ and using (4.2) accounts for the difference in the SLE at $t$ and $t - 1$. The CG method with the initializations (4.1)-(4.2) will be referred to as the Warm-Started CG (WS-CG) and a VAMP algorithm where the LMMSE is approximated by WS-CG will be called Warm-Started Conjugate Gradient VAMP (WS-CG-VAMP).

The idea of using a warm-starting strategy is supported by the similarity of the SLE (2.130) at two consecutive iterations given the algorithm has not progressed substantially. To see this, we use the SVD of $A = USV^T$ and define $\Lambda = SS^T$ to show that in the SLE (2.130), the matrix $W_t$
\begin{align*}
W_t &= v_w I_M + v_{tB\to A}^t AA^T = U(v_w I_M + v_{tB\to A}) U^T
\end{align*}
changes only in the eigenvalues, while the eigenvectors of $W_t$ remain the same for all $t$. Therefore, if $v_{tB\to A}$ and $v_{t-1B\to A}$ are close, then $W_t$ will not change much from the previous iteration.

Second, we observe that when the intrinsic variance $v_{tB\to A}$ does not change much at two consecutive iterations, the associated error vector $q_t$ stays approximately the same as well. As a result, the observed vector $z_t = w - Aq_t$ in the SLE remains close to its version from the previous iteration $z_{t-1}$. That, together with the similarity of $W_t$ at two consecutive iterations, provides the motivation for reusing the CG information generated at the previous outer-loop iteration.
4.1.1 Rigorous dynamics of Block A in WS-CG-VAMP

As presented in Section 2.4.1, when the CG algorithm is initialized with $\mu_A^{t,0} = 0$, the asymptotic dynamics of the resulting CG-VAMP algorithm can be defined through a 1D State Evolution. However, if we used the initializations (4.1)-(4.2) and kept the same structure of $x_{A \rightarrow B}^t$ from Algorithm 2 with a single Onsager correction $\gamma_{A}^{t,i} x_{B \rightarrow A}^t$, the resulting CG-VAMP algorithm would lose the SE property.

To see this, first, we can sequentially apply the initializations (4.1)-(4.2) to show that the output $\mu_A^{t,i}$ of WS-CG at iteration $t$ depends on all the previous outputs $\mu_A^{\tau,i}$ and the conjugate directions vectors $p_{\tau}^{t-1}$, i.e.

$$\mu_A^{t,i} = g_A^i (z_t, \mu_A^{\tau,i}{_{\tau=0}}, p_{\tau}^{t-1}{_{\tau=0}}).$$  

(4.3)

Next, we note that both $\mu_A^{\tau,i}$ and $p_{\tau}^{t-1}$ depend on $z_\tau$, which further depends on $w$ and $q_\tau$ as we know from (2.132). Therefore we have that the WS-CG output $\mu_A^{t,i} = g_A^i (Q_t, w)$, where $Q_t = (q_0, ..., q_t)$, depends on the whole history of the error vectors $q_\tau, \tau = 0, ..., t$ contrary to the zero-initialized CG, where $\mu_A^{t,i}$ is a function of only $q_t$ as follows from (2.139) and the fact that $z_t = w - A q_t$. In [60] it was shown that one needs to construct an Onsager correction term $\gamma_{A}^{t,\tau} x_{B \rightarrow A}^\tau$ for each error vector $q_\tau$ the function $g_A$ depends on. Thus, for WS-CG the new form of the update $x_{A \rightarrow B}^t$ must be changed to [60]

$$x_{A \rightarrow B}^t = \frac{1}{\sum_{\tau=0}^t \gamma_{A}^{t,\tau,i}} \left( \sum_{\tau=0}^t \gamma_{A}^{t,\tau,i} x_{B \rightarrow A}^\tau - A^T \mu_A^{t,i} \right)$$  

(4.4)

in order to preserve the asymptotic Gaussianity of the error vector $h_t = x_{A \rightarrow B}^t - x$ and ensure the existence of an SE. Here the scalars $\gamma_{A}^{t,\tau,i}$ are the generalized versions of the divergence (2.127) for each error vector $q_\tau$.

$$\gamma_{A}^{t,\tau,i} = \frac{1}{N} \nabla_{q_\tau} \cdot \left( A^T g_A^i (Q_t, w) \right) = \frac{1}{N} \sum_{k=1}^N \frac{\partial \left( A^T g_A^i (q_0, q_1, ..., q_t, w) \right)_k}{\partial (q_\tau)_k}.$$  

(4.5)

Besides this standard definition of $\gamma_{A}^{t,\tau,i}$, in Appendix F we show that it also follows the generalized form of the identity (3.5) from Lemma 1 on page 57,

$$\lim_{N \rightarrow \infty} \frac{1}{N} q_\tau A^T \mu_A^{t,i} \overset{a.s.}{=} \lim_{N \rightarrow \infty} \sum_{\tau=0}^t \psi_{t,\tau} \gamma_{A}^{t,\tau,i},$$  

(4.6)
where
\[ \psi_{\tau,\tau'} = \frac{1}{N} q_{\tau}^T q_{\tau'} = (\Psi_t)_{\tau,\tau'} \quad \text{for } \tau, \tau' \leq t. \quad (4.7) \]

As a result of changing the structure of \( x_t^A \to B \), the SE of Block A changes as well. To see this, we define \( C_A = \frac{1}{\sum_{\tau=0}^{\tau} \gamma_A^{\tau}_t} \) and follow the steps from (3.11) to obtain
\[
v_{t,i}^A \to B \overset{a.s.}{=} \lim_{N \to \infty} C_A^2 \frac{1}{N} \left( \sum_{\tau=0}^{t} \gamma_A^{\tau,i}_t q_{\tau} - A^T \mu_{t,i}^A \right)^2 + \sum_{\tau,\tau'=0}^{t} \psi_{\tau,\tau'} \gamma_A^{\tau,i}_t \gamma_A^{\tau',i}_t, \quad (4.8)\]

where in (a) we used (4.4) and the last step of (4.8) is due to (4.6). Now, in order to derive the closed-form solutions to (4.5) and (4.8), we follow a similar strategy as in Section 2.4.1 and, first, show that in the LSL, WS-CG corresponds to a linear mapping of all \( z_{\tau} \) for \( \tau = 0, \ldots, t \).

**Theorem 4.** Let \( \mu_{t,i}^A \) be the output of WS-CG with the initializations (4.1) and (4.2). Then, under Assumptions B1-B3 from page 42, we have that
\[
\mu_{t,i}^A = \sum_{\tau=0}^{t} \sum_{k=0}^{(t-\tau)} r_{t,i}^A[k] (\Phi)^k z_{\tau}, \quad (4.9)\]
where \( \Phi = AA^T \) and \( \lim_{N \to \infty} r_{t,i}^A[k] \) is a scalar function of \( \delta, v_w \) and \( \lim_{N \to \infty} \Psi_t \), and is defined in Appendix D.

**Proof.** See Appendix D.

As seen from the theorem, WS-CG has a similar asymptotic structure to (2.139) for zero-initialized CG and, in fact, as shown in Appendix D, the dependence of \( \mu_{t,i}^A \) from (4.9) on \( z_{\tau} \) is exactly the same as in (2.139). Next, by substituting (4.9) into (4.5) and (4.8), we can obtain the closed-form solutions for \( \gamma_{t,i}^A \) and \( v_{t,i}^A \to B \) respectively. The results are formulated in the following theorem.

**Theorem 5.** Let \( \mu_{t,i}^A \) be the output of the WS-CG algorithm with the initializations (4.1) and (4.2), and \( r_{t,i}^A[k] \) be as in (4.9). Then, under Assumptions B1-B3 from
Warm-starting in MP page 42, the scalar $\gamma_{A}^{i,\tau,i}$ from (4.5) almost surely converges to

$$\lim_{N \to \infty} \gamma_{A}^{i,\tau,i} = \lim_{N \to \infty} \frac{(t-\tau)i}{\sum_{k=0}^{(t-\tau)i} r_{\tau}^{t,i}[k] \chi_{k+1}}, \quad (4.10)$$

where $\chi_{j}$ is as in (4.49). Additionally, the variance $\upsilon_{A \to B}^{i}$ from (4.8) evolves as

$$\upsilon_{A \to B}^{i} = \lim_{N \to \infty} \frac{\Omega_{i}^{2} - \sum_{\tau,\tau'=0}^{t} \sum_{j=0}^{(t-\tau) i} \sum_{k=0}^{(t'-\tau') i} r_{\tau}^{t,i}[j] r_{\tau'}^{t,i}[k] \psi_{\tau,\tau'}^{i} \chi_{j+k+2} + \upsilon_{w} \chi_{j+k+1})}{2} \quad (4.11)$$

with

$$\Omega_{i}^{2} = \sum_{\tau,\tau'=0}^{t} \sum_{j=0}^{(t-\tau) i} \sum_{k=0}^{(t'-\tau') i} r_{\tau}^{t,i}[j] r_{\tau'}^{t,i}[k] \psi_{\tau,\tau'}^{i} \chi_{j+k+2} + \upsilon_{w} \chi_{j+k+1}). \quad (4.12)$$

Proof. See Appendix E.

Comparing Theorem 5 to (2.140) and (2.142), we can conclude that the asymptotic results for the divergence $\gamma_{A}^{i,\tau,i}$ and the variance $\upsilon_{A \to B}^{i}$ are structurally similar for both WS-CG and the zero-initialized CG. However, contrary to the zero-initialized CG case where the respective evolution model (2.142) for $\upsilon_{A \to B}^{i}$ depends only on the variance $\upsilon_{B \to A}^{i} = \lim_{N \to \infty} \frac{1}{N} \| q_{i} \|^{2}$ from the current iteration, the model (4.11) depends on the whole cross-correlation matrix $\lim_{N \to \infty} (\Psi_{t})_{\tau,\tau'}$ from (4.7). In other words, the model defining the evolution of the intrinsic variance in WS-CG-VAMP is multidimensional.

Besides the structural similarity of the asymptotic results for $\gamma_{A}^{i,\tau,i}$ and $\upsilon_{A \to B}^{i}$ for WS-CG and zero-initialized CG, the practical drawbacks of the two pairs of results are similar as well and, moreover, worse in the WS-CG case. In particular, (4.10) relies on access to the spectral moments $\chi_{j}$ of order at least $ti$ for constructing $\gamma_{A}^{i,\tau,i}$ contrary to the zero-initialized CG case, where the maximum moment order was $i$. An even worse situation with respect to the moment order is observed in (4.11), which also requires access to the cross-correlation values $\psi_{\tau,\tau'}$ from (4.7) and is harder to estimate than just $\upsilon_{B \to A}^{i}$ that is the sufficient information for (2.140). In this thesis we propose to estimate this cross-correlation value based on
the following asymptotic result
\[
\lim_{N \to \infty} \frac{1}{N} z^T \tau z_{\tau'} - \delta v_w = \lim_{N \to \infty} \frac{1}{N} (w - A q_{\tau})^T (w - A q_{\tau'}) - \delta v_w
\]
\[
\overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} q_{\tau} A^T A q_{\tau'}
\]
\[
\overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} q_{\tau}^T q_{\tau'} = \psi_{\tau,\tau'},
\]
(4.13)
where we used the following two asymptotic properties
\[
\lim_{N \to \infty} \frac{1}{N} w^T A q_{\tau} \overset{a.s.}{=} 0 \quad (4.14)
\]
\[
\lim_{N \to \infty} \frac{1}{N} q_{\tau}^T A^T A q_{\tau'} = \lim_{N \to \infty} \frac{1}{N} q_{\tau}^T q_{\tau'}
\]
proved in [60] to hold in MP algorithms under Assumptions B1-B3 from page 42. Then, one possible way to estimate \( \psi_{\tau,\tau'} \) is as
\[
\hat{\psi}_{\tau,\tau'} = \frac{1}{N} z^T \tau z_{\tau'} - \delta \hat{v}_w. \quad (4.16)
\]
In the simulation section, we confirm the validity of the SE (4.11) provided the oracle values of \( \chi_j, v_w \) and \( \psi_{\tau,\tau'} \) are known, but also demonstrate the instability of WS-CG-VAMP in the practical scenario where (4.10) and (4.11) are used to construct the corresponding estimators without the oracle information. Therefore, although Theorem 5 provides theoretical tools for studying the asymptotics of WS-CG-VAMP, we need an alternative robust method for estimating \( \gamma^{t,\tau,i}_{A} \) and \( v_{t,i}^{\tau,i}_{A\to B} \) in practical scenarios.

Lastly, we would like to mention another practical issue with the above approaches for estimating \( \gamma^{t,\tau,i}_{A} \) and \( v_{t,i}^{\tau,i}_{A\to B} \). As we mentioned before, the magnitude of the spectral moments \( \chi_j \) grow exponentially with \( j \) and the required precision of the variables storing these moments and, importantly, the intermediate results involving these moments, grows very quickly. Let \( p = \text{prec}(j) \) be a function defining the minimal number of significant decimal digits \( p \) used by variables in WS-CG-VAMP in order to remain stable when \( j \)-th spectral moment is used in the algorithm. Then, based on our simulation experiments in Section 4.3.2, for even mildly ill-conditioned \( A \), the required precision approximately grows as \( \text{prec}(j) = j/2 \). For example, if one uses \( i = 10 \) at every outer-loop iteration \( t \), then at \( t = 20 \) the maximal spectral moment order \( j_{\max} \) required to implement
(4.11) is $j_{\text{max}} = (2ti + 2)/2 = 201$. As a result, the variables with such a precision require a special treatment\(^1\) and the resulting computation of the estimator based on (4.11) might become extremely expensive. This problem might be avoided by a certain normalization\(^2\) of the scalars involved in (4.11), but that would require access to the closed-form solution to the moments $\chi_j$, which is usually unavailable in practice.

### 4.1.2 Robust estimation of Block A parameters in WS-CG-VAMP

We begin with deriving a more robust method for estimating $v_{t,i}^{A\rightarrow B}$ without explicitly referring to $\chi_j$ within WS-CG-VAMP. In Section 3.2 we introduced Theorem 3 that defines a robust and fast estimator of the variance $v_{A\rightarrow B}^{t,i}$ for CG-VAMP where the LMMSE estimator is approximated by zero-initialized CG. There, we started with an asymptotic identity (3.11) similar to (4.8) and used the conjugacy property of CG to construct an iterative update for the norm $\frac{1}{N} \|A\mu_{t,i}^{A}\|^2$. Unfortunately, the same approach does not apply to WS-CG because the warm-starting breaks down the conjugacy property. Instead, we refer to another fundamental asymptotic result of any MP algorithm where $x_{t}^{B\rightarrow A}$ and $x_{t}^{A\rightarrow B}$ are updated as in (3.2) and (4.4) respectively – the asymptotic orthogonality of the error vectors $q_{\tau}$ and $h_{\tau}$ for $\tau, \tau' \leq t$ \([60]\),

$$
\lim_{N \rightarrow \infty} \frac{1}{N} h_{\tau}^{T} q_{\tau} \overset{a.s.}{=} 0 \quad (4.17)
$$

that holds under Assumptions B1-B3 from page 42. One can show that the Onsager correction $\gamma_{B}^{t,i} x_{A\rightarrow B}^{t}$ in the update for $x_{B\rightarrow A}^{t}$ and $t$ Onsager correction terms $\{\gamma_{A}^{\tau,i} x_{B\rightarrow A}^{\tau}\}_{\tau=0}$ in the update for $x_{A\rightarrow B}^{t}$ can be viewed as the terms ensuring the orthogonality (4.17). This viewpoint will be used in the next chapter, where we get to designing methods for efficient estimation of the Onsager correction for the

---

\(^1\)The simulations experiments in this thesis have been implemented in MATLAB, and to execute the estimators based on (4.11) we used symbolic variables and the operations associated with this type of variable.

\(^2\)This approach was used in the implementation of the MAMP algorithm \([14]\), where an estimator with a similar structure to (4.11) was used to estimate $v_{A\rightarrow B}^{t,i}$. 

76
Warm-starting in MP

denoiser $g_B$. For now, we can use $(4.17)$ to obtain
\[
\lim_{N \to \infty} \frac{1}{N} \| x_{t \to B}^t - x_{B \to A}^t \|^2 - v_{B \to A}^t = \lim_{N \to \infty} \frac{1}{N} \| x + h_t - x - q_t \|^2 - v_{B \to A}^t \\
= v_{A \to B}^t + v_{B \to A}^t - v_{B \to A}^t.
\]
(4.18)

Then, substituting $\hat{v}_{B \to A}^t$ into the last result gives us an estimator for $v_{A \to B}^t$,
\[
\hat{v}_{A \to B}^t = \lim_{N \to \infty} \frac{1}{N} \| x_{A \to B}^t - x_{B \to A}^t \|^2 - \hat{v}_{B \to A}^t.
\]
(4.19)

Next, to derive a more stable method for estimating $\gamma_{A,t,i}^t$, we follow the same steps as in Section 3.1 where we derived an asymptotic iterative scheme for updating $\gamma_{A,t,i}^t$ for the zero-initialized CG. There, the key idea was to view $\gamma_{A,t,i}^t$ not as the divergence of $g_A$, but as a scalar following a particular identity, namely (3.5), which now takes a more general form (4.6). The latter can be used together with the fact that $z_\tau = w - Aq_\tau$, as was shown in (3.6), to obtain
\[
\lim_{N \to \infty} \frac{1}{N} w^T \mu_{A,t,i}^t - \frac{1}{N} z_\tau^T \mu_{A,t,i}^t \overset{a.s.}{=} \lim_{N \to \infty} \sum_{\tau=0}^t \psi_{t,\tau} \gamma_{A,t,i}^t. \tag{4.20}
\]

Then, by defining a matrix $Z_t = (z_0, z_1, ..., z_t)$, the set of scalars $(\gamma_{A,t,i}^t)_\tau = \gamma_{A,t,i}^t \tau$ for $\tau = 0, ..., t$ can be recovered from
\[
\gamma_{A,t,i}^t \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} w^T \mu_{A,t,i}^t - \frac{1}{N} Z_t^T \mu_{A,t,i}^t, \tag{4.21}
\]
where $\Psi_t$ is as in (4.7) and its asymptotic invertibility was confirmed in [92]. The following theorem completes this idea and presents an iterative process that defines the asymptotic behaviour of $\gamma_{A,t,i}^t$ within WS-CG-VAMP.

**Theorem 6.** Let $\mu_{A,t,i}^t$ be the output of the WS-CG algorithm with the initializations (4.1) and (4.2) after $i$ inner-loop iterations. For $j = 0, ..., i$, let $a_j^t$ and $b_j^t$ be the scalars computed as in the CG algorithm from Algorithm 3. Define a recursion
\[
\nu_i^t = \nu_{i-1}^t + a_{i-1}^t \eta_{i-1}^t \tag{4.22}
\]
\[
\sigma_i^t = \Psi_t^{-1}(\nu_i^t 1_t - \frac{1}{N} Z_t^T \mu_{A,t,i}^t) \tag{4.23}
\]
\[
\eta_i^t = w(\delta - \nu_i^t + v_{B \to A}^t \|\sigma_i^t\|_1) + b_{i-1}^t \eta_{i-1}^t \tag{4.24}
\]
with the initializations $\nu_0^t = 0$, $\eta_0^t = \delta v_w$ for $t = 0$, and $\nu_i^t = v_{i-1}^t$, $\eta_i^t = v_w(\delta -$
\[ \nu_{t-1} + v_{t}^{B \rightarrow A}||\gamma_{A}^{t-1, i}||_1 + b_{t-1}^{-1} \eta_{t-1}^{i} \text{ for } t > 0. \]

Then, under Assumptions B1-B3 from page 42, we have that

\[ \lim_{N \to \infty} \gamma_{A}^{t, i} \overset{\text{a.s.}}{=} \lim_{N \to \infty} \sigma_{t}^{i}. \quad (4.25) \]

**Proof.** See Appendix F. \[ \blacksquare \]

From the practical point of view, estimating the whole set \( \gamma_{A}^{t, \tau, i} \) requires an access to an estimate of \( \Psi_{t} \) that we can compute from (4.13) with \( v_{w} \) replaced by \( \hat{v}_{w} \). In the simulation section, we demonstrate that WS-CG-VAMP equipped with the variance \( v_{t}^{A \rightarrow B} \) estimator (4.19) and the estimator of \( \gamma_{A}^{t, \tau, i} \) based on Theorem 6 exhibits stable dynamics for a much larger range of \( i \) and \( t \) compared to the same algorithm but where \( \gamma_{A}^{t, \tau, i} \) is estimated based on (4.10). Yet, when the algorithm is progressing and the variance \( v_{t}^{B \rightarrow A} \) is decreasing, the matrix \( \Psi_{t} \) becomes ill-conditioned (since \( \lim_{N \to \infty} (\Psi_{t})_{\tau, \tau} = v_{\tau}^{B \rightarrow A} \) [100] and a small error in estimating \( \psi_{\tau, \tau'} \) leads to a large error in estimating the inverse \( \Psi_{t}^{-1} \). From the numerical study, we have observed that for a large number of outer-loop iterations \( t \), the WS-CG-VAMP with (4.13) used to estimate \( \psi_{\tau, \tau'} \) might eventually diverge. To the best of our knowledge, there is no method that demonstrates higher robustness in estimating \( \psi_{\tau, \tau'} \) and instead we seek a more basic structure of \( x_{A \rightarrow B}^{t} \) that involves a simpler correction of WS-CG.

Lastly, we can link the above theorem to Theorem 2 by making the following observations. First, note that by unrolling (4.22) we get \( \nu_{t}^{i} = \sum_{j=0}^{i-1} a_{t}^{j} \eta_{t}^{j} \). As we show in Appendix F, this scalar defines the asymptotic behaviour of the following inner-product

\[ \lim_{N \to \infty} \frac{1}{N} w^{T} \mu_{A}^{t, i} \overset{\text{a.s.}}{=} \lim_{N \to \infty} \sum_{j=0}^{i-1} a_{t}^{j} \eta_{t}^{j}. \quad (4.26) \]

and it fully matches the same inner-product for the zero-initialized CG in (3.9). Additionally, in Appendix B we showed for the zero-initialized CG that in (4.24) the term \(-\nu_{t}^{i} + v_{t}^{B \rightarrow A} ||\sigma_{t}^{i}||_1 \) can be simplified to \(-\frac{1}{N} z_{t}^{T} \mu_{t}^{i} \), which allowed to define \( \eta_{t}^{i} \) recursively as in (3.8) without the need for \( \nu_{t}^{i} \) and \( \sigma_{t}^{i} \). However, for the WS-CG it is not possible to make such a simplification and one needs to update all three components \( \nu_{t}^{i}, \sigma_{t}^{i} \) and \( \eta_{t}^{i} \) as in Theorem 6.
4.1.3 Approximate update $x^t_{A\rightarrow B}$ in WS-CG-VAMP

In the previous subsections we considered using the update (4.4) for $x^t_{A\rightarrow B}$ with the full set of Onsager corrections $\{\gamma^t_{A,\tau,i}x^\tau_{B\rightarrow A}\}_{\tau=0}$ in WS-CG-VAMP. However, from the numerical experiments we have found out that for small $t$ the magnitude of the Onsager correction terms $O^t = \gamma^t_{A,\tau,i}x^\tau_{B\rightarrow A}$ in (4.4) rapidly decays as $\tau$ decreases. This suggests that (4.4) might be well-approximated by the update

$$
\begin{align*}
x^t_{A\rightarrow B} &= (\gamma^t_{A,i})^{-1}(\gamma^t_{A,i}x^t_{B\rightarrow A} - A^T\mu^t_{A,i}) \\
&= x^t_{B\rightarrow A} - (\gamma^t_{A,i})^{-1}A^T\mu^t_{A,i},
\end{align*}
$$

(4.27)

which is exactly the same as the update for $x^t_{A\rightarrow B}$ in regular CG-VAMP since $\gamma^t_{A,i} = \gamma^t_{A,i}$. Therefore, an alternative, although non-rigorous, approach to implement WS-CG-VAMP is to keep the same update rule of $x^t_{A\rightarrow B}$ as in CG-VAMP and use (3.10) to estimate $\gamma^t_{A,i}$. While such an MP algorithm is no longer guaranteed to have most of the asymptotic properties discussed before and might suffer from stability issues, we will numerically demonstrate that WS-CG-VAMP with an update (4.27) is stable for many outer-loop iterations and can substantially outperform the standard CG-VAMP algorithm.

4.2 Generalized Warm-Start Framework

Previously, we considered the problem of upscaling VAMP by using a particular algorithm, Conjugate Gradient, to approximate the intractable LMMSE (2.126). To improve the convergence properties of an MP algorithm utilizing CG, we designed a certain warm-starting scheme utilizing information from the previous iterations. In this section, we generalize this idea and propose a unified warm-started framework for designing iterative methods that can be used to upscale VAMP. This framework includes WS-CG designed previously and a new Warm-Started Gradient Descent algorithm, besides other possible iterative methods. We rigorously prove that a fixed point of an MP algorithm utilizing an instance of this framework is a fixed point of VAMP. This implies that WS-CG-VAMP is conjectured to be Bayes optimal under the same assumptions as for VAMP. Besides that, we use the proposed framework to show that the upscaling mechanism of the recently
proposed Memory AMP algorithm [14, 104] is a special case of our warm-start framework.

4.2.1 Equivalent SLE in VAMP

Before proceeding next, we would like to introduce a pair of terms representing two types of MP algorithms: single-memory (SM) MP and long-memory (LM) MP. The former type of MP algorithms is shown in Algorithm 2 and represents an algorithm where the output of each block depends only on the last output of another block, i.e. \( x_{A \rightarrow B}^t \) depends only on \( x_{B \rightarrow A}^t \), and \( x_{B \rightarrow A}^{t+1} \) depends only on \( x_{A \rightarrow B}^t \). On the other hand, LM MP algorithms allow the output of a block to depend on all the outputs of the another block. For example, EP, VAMP and CG-VAMP use the function \( g_A^t(x_{B \rightarrow A}^t) \) that depend only on \( x_{B \rightarrow A}^t \) and, therefore, are SM MP, while LM MP algorithms such as WS-CG-VAMP use a linear function \( g_A^t(x_{B \rightarrow A}^t, x_{B \rightarrow A}^{t-1}, \ldots, x_{B \rightarrow A}^0) \) that depends on all \( x_{B \rightarrow A}^\tau, \tau = 0, ..., t \). Later in this chapter we will compare the benefits of both types of MP, while for now we focus on LM MP algorithms that take the following form

\[
x_{A \rightarrow B}^t = \frac{1}{\sum_{\tau=0}^{t} \gamma_{A}^{t,\tau}} \left( \sum_{\tau=0}^{t} \gamma_{A}^{t,\tau} x_{B \rightarrow A}^\tau - A^T g_A^t(X_{B \rightarrow A}^{t+1}, y) \right)
\]

\[
x_{B \rightarrow A}^{t+1} = \frac{1}{1 - \gamma_{B}^{t+1}} \left( g_B(x_{A \rightarrow B}^t, \hat{v}_{A \rightarrow B}^t) - \gamma_{B}^{t+1} x_{A \rightarrow B}^t \right),
\]

where \( X_{B \rightarrow A}^{t+1} = (x_{B \rightarrow A}^0, x_{B \rightarrow A}^1, \ldots, x_{B \rightarrow A}^t) \), \( g_B \) is a denoiser, and

\[
\gamma_{A}^{t,\tau,i} = \frac{1}{N} \nabla x_{B \rightarrow A}^\tau \cdot \left( A^T g_A^t(X_{B \rightarrow A}^t, y) \right)
\]

\[
\gamma_{B}^{t+1} = \frac{1}{N} \nabla (x_{A \rightarrow B}^t) \cdot g_B(x_{A \rightarrow B}^t, \hat{v}_{A \rightarrow B}^t)
\]

are the divergences of \( g_B \) and \( g_A \) respectively. In (4.28), the vector \( g_A^t(X_{B \rightarrow A}^{t+1}, y) \) is the output of some method for approximating the solution \( \mu_A^t \) to a rescaled version of the SLE from (2.130)

\[
W_t \mu_A^t = z_t,
\]

where \( z_t = y - Ax_{B \rightarrow A}^t \) remains the same as before, but \( W_t \) takes the new form

\[
W_t = \rho t I + AA^T,
\]
where \( \rho = \hat{\rho}_{B \rightarrow A} \). One can relate (4.33) to the original matrix \( W_t \) from (2.131) by multiplying the latter by \( (v_{B \rightarrow A}^t)^{-1} \). Importantly, if one implemented the MP algorithm (4.28)-(4.29) with

\[
\mathbf{g}_A^t(X_{B \rightarrow A}^{t+1}, y) = W_t^{-1}z_t
\]

being the exact solution to the SLE (4.32), then the update (4.28) would be invariant to rescaling of \( W_t \) due to the normalization by \( \gamma_{A}^{t,\tau} \). To see this, let \( \tilde{\mu}_A^t = \rho \mu_A^t \), where \( \mu_A^t = W_t^{-1}z_t \) is the solution to (4.32) and \( \rho \) is some non-zero scalar. Note that when \( \rho = v_{B \rightarrow A}^t \), \( \tilde{\mu}_A^t \) is the solution to (2.130). Additionally, let \( \tilde{\gamma}_{A}^{t,\tau} = \frac{1}{N} \nabla x_{B \rightarrow A}^\tau \cdot (A^T \tilde{\mu}_A^t) \) be the divergence of \( \tilde{\mu}_A^t \) with respect to \( x_{B \rightarrow A}^\tau \) and note that \( \tilde{\gamma}_{A}^{t,\tau} = \frac{1}{N} \nabla x_{B \rightarrow A}^\tau \cdot (A^T \mu_A^t) \). Then, if we constructed an equivalent update

\[
\tilde{x}_{A \rightarrow B}^t = \frac{\sum_{\tau=0}^{t-1} \gamma_{A}^{t,\tau} x_{B \rightarrow A}^\tau - A^T \mu_A^t}{\sum_{\tau=0}^{t-1} \gamma_{A}^{t,\tau}} = \frac{\sum_{\tau=0}^{t} \tilde{\gamma}_{A}^{t,\tau} x_{B \rightarrow A}^\tau - A^T \mu_A^t}{\sum_{\tau=0}^{t} \tilde{\gamma}_{A}^{t,\tau}},
\]

then it would be same as if \( \tilde{\mu}_A^t = \mu_A^t \) without the scaling by \( \rho \). This fact will be used shortly, when we analyze the MAMP algorithm from [14], while for now this result suggests that the VAMP algorithm with the original SLE (2.130) or with the rescaled SLE (4.32) are identical and, therefore, we can shift our focus to approximating the solution \( \mu_A^t \) of (4.32).

### 4.2.2 Long-memory LMMSE approximation

In the following, we consider approximating the LMMSE estimator (4.34) from the optimization point of view and use the same quadratic cost function as in Section 2.4 where we introduced the CG algorithm,

\[
f_i(\mu) = \frac{1}{2} \mu^T W_i \mu - z_i^T \mu.
\]

Recall that by setting \( \nabla \mu f_i(\mu) = W_i \mu - z_i = 0 \), we can verify that \( f_i(\mu) \) is minimized by the LMMSE estimator (4.34). Here, we consider minimizing this cost function by the following general iterative scheme

\[
\mu_{A}^{t+1} = \mu_A^t + a_i \mathbf{p}_i^t
\]

\[
\mathbf{p}_{i+1}^t = -\nabla \mu_{A}^{t+1} f_i(\mu_{A}^{t+1} + b_i \mathbf{p}_i^t),
\]

81
Warm-starting in MP

where $\mu_A^{t,i}$ acts as the approximation of (4.34) after $i$ iterations, $p_t^i$ is the new search direction and $a_t^i$ is the step size along $p_t^i$. One special case of (4.37) is the Gradient Descent (GD) algorithm for which we set $b_t^i = 0$ for all $i$ and choose some step size $a_t^i$. One way of choosing the step size is to refer to the optimal but fixed step size $a_t^i = a_t$ as \[95, 105\]

\[
2 = \frac{L_{\text{max}}(W_t) + L_{\text{min}}(W_t)}{\rho_t + \lambda_t},
\]

where $L_{\text{min}}(W_t)$ and $L_{\text{max}}(W_t)$ are the minimum and maximum eigenvalues of $W_t$ and $\lambda_t = L_{\text{max}}(S_\cdot S^T) + L_{\text{min}}(S_\cdot S^T)$. Similarly, by defining

\[
a_t^i = \frac{\|z_t - W_t \mu_A^{t,i}\|^2}{(p_t^i)^T W_t p_t^i}, \quad b_t^i = \frac{\|z_t - W_t \mu_A^{t,i+1}\|^2}{\|z_t - W_t \mu_A^{t,i}\|^2},
\]

one obtains the Conjugate Gradient method. One can show \[95, 105\] that CG is the optimal first-order method for minimizing (4.36) and has a substantially faster convergence rate compared to GD when the matrix $W_t$ is ill-conditioned.

Although both of the methods are guaranteed to converge to the exact solution (4.34) \[105\], the number of inner-loop iterations required for that is $i = M$ for CG and $i \to \infty$ for GD, which is as expensive as directly computing (4.34). To keep the number of inner-loop iterations of (4.37) of order $O(1)$ while preserving the ability to recover the exact LMMSE solution, we suggest combining a small number of inner loop iterations with the following warm-starting scheme at each outer loop iteration

\[
\mu_A^{0,0} = \mu_A^{t-1,i}, \quad p_0^0 = -\nabla_{\mu_A^{t-1,i}} f_t(\mu_A^{t-1,i}) + b_{t-1}^{-i} p_{t-1}^{-i},
\]

and $\mu_0^0 = 0$, $p_0^0 = z_t$ for $t = 0$. Note that by applying the initialization (4.40), one can follow the same reasoning as in Section 4.1.1 to show that $g_A^{t,i}(X_{B \to A}^{t+1}, y) = \mu_A^{t,i}$ is a function of all $x_{B \to A}^\tau$ for $\tau \leq t$. Therefore, we call such a function $g_A^{t,i}(X_{B \to A}^{t+1}, y)$ to be LM.

The following theorem establishes the asymptotic property of an LM MP algorithm utilizing (4.37) with the initialization (4.40).

**Theorem 7.** Consider an LM MP algorithm that fits into (4.28)-(4.29) with $g_A^{t,i}(X_{B \to A}^{t+1}, y) = \mu_A^{t,i}$ being the output of the iterations (4.37) with any $i > 0$
and the initialization (4.40). Assume that there is such \( i^* \) so that \( \mu_{t,i}^{i,i^*} \) minimizes \( f_t(\cdot) \) from (4.36) and that the chosen LM MP algorithm has a fixed point achieved at (possibly infinite) iteration \( t^* \). Then, under Assumptions B1-B3 from page 42, the vector \( \mu_{t,i}^{i,i^*} \) converges to the solution of (4.32).

**Proof.** Let an LM MP algorithm that fits into (4.28)-(4.29) converge to a unique fixed point at iteration \( t^* \). Consider the iteration \( t^* + 1 \) and use (4.40) to obtain

\[
\mu_{t,i}^{(t^*+1),0} = \mu_{t,i}^{t^*,i} \\
p_{t,i}^{0} = -\nabla_{\mu_{t,i}^{t^*,i}} f_{t,i}^{t^*+1}(\mu_{t,i}^{t^*,i}) + b_{t,i}^{i-1} p_{t,i}^{i-1}.
\]

Because we are at the fixed point, we have \( v_{B \to A}^{t^*} = v_{B \to A}^{t^*+1} \) so \( W_{t+1} = W_t \) and \( z_{t+1} = z_t \), which implies that \( \nabla_{(\cdot)} f_{t,i}^{t+1}(\cdot) = \nabla_{(\cdot)} f_{t,i}^{t}(\cdot) \), \( a_{t+1} = a_t \), and \( b_{t+1} = b_t \). Therefore, by running the iterative scheme (4.37) for \( i \) iterations, we produce \( \mu_{i}^{t^*+1} = \mu_{i}^{t^*} \) and \( p_{i}^{t^*+1} = p_{i}^{t^*} \). Then, after at most \( i \) outer-loop iterations, we obtain \( \mu_{i}^{t^*} \), which, by the assumption, minimizes \( f_t(\cdot) \) from (4.36) and, therefore, corresponds to the solution of the SLE (4.32). \( \blacksquare \)

The above theorem says that if one designs an iterative scheme for approximating the LMMSE estimator (4.34) that is proved to minimize (4.36), fits into the recursion (4.37) and uses the warm-starting (4.40), then a fixed point of an LM MP algorithm (4.28)-(4.29) utilizing such a scheme must be a fixed point of VAMP. Importantly, this is true for any number of iterations \( i > 0 \) of such an iterative approximation scheme. As a corollary, Theorem 7 proves that under Assumptions B1-B3 from page 42, a fixed point of WS-CG-VAMP must be a fixed point of VAMP. The same holds for *Warm-Started Gradient Descent VAMP (WS-GD-VAMP)*, where LMMSE (4.34) is approximated with a WS-GD algorithm defined above.

### 4.2.3 Memory AMP

The recently proposed MAMP algorithm [14] approximates (4.34) with an LM approximation method based on a Taylor series expansion of the matrix \( W_T^{-1} \) as a power series of \( W_T \) from (4.33). Using the ideas above, we can show that this approximation method corresponds to a particular WS-GD scheme with \( i = 1 \)
inner-loop iteration. In particular, in MAMP, at iteration $t$ the SLE (4.34) is approximated by the vector $\mu^t_A$ computed as [14]

$$\mu^t_A = (I - a_t W_t) \mu^{t-1}_A + \bar{a}_t z_t,$$

(4.41)

where $a_t$ is the same as in (4.38), while $\bar{a}_t$ is optimized to minimize the output variance $\sigma^2_{A \to B}$ of the step (4.28). To relate (4.41) to (4.37), we let $\bar{a}_t = c_t a_t$, where $c_t$ is some scalar, and add the corresponding superscript $i$. Then, after rearranging the terms in (4.41), one can obtain

$$\mu^{i,1}_A = \mu^{i-1,0}_A + a^0_i p^0_i$$

$$p^0_i = -(W_t \mu^{i-1,0}_A - c_t z_t).$$

(4.42)

By comparing (4.42) to (4.37), we conclude that this update corresponds to WS-GD with $i = 1$ minimizing the cost

$$f_t(\mu) = \frac{1}{2} \mu^T W_t \mu - c_t z_t^T \mu$$

(4.43)

that has a minimum at $c_t W_t^{-1} z_t$. Recall that a GD algorithm is guaranteed to converge to the minimum of (4.43) [105]. Then, by defining the initializations $\mu^{t,0}_A = \mu^{t-1,1}_A$ for $t > 0$ and $\mu^{0,0}_A = 0, b^0_t = 0$ for $t = 0$, setting $a^0_t = \frac{1}{\rho_t + \lambda}$, and applying Theorem 7 to (4.42) and (4.43), we confirm that at the fixed point $t^*$, the vector $\mu^{t\star,1}_A$ from (4.41) converges to $\mu^{t\star}_A = c_{t\star} W_{t\star}^{-1} z_{t\star}$. However, as pointed out in Section 4.2.1, the update (4.28) is invariant with respect to scaling of the LMMSE solution (4.34) and, therefore, MAMP has the same set of fixed points as VAMP.

### 4.2.4 Asymptotic behaviour of WS methods within Message Passing

Similarly to WS-CG-VAMP case, in the following we establish the asymptotic properties of an MP algorithm utilizing an instance of the iterative scheme (4.37) with the initialization (4.40) to approximate the LMMSE estimator (4.34). Due to the close similarity of this iterative scheme to WS-CG from the previous section, the derivation of the Onsager correction and the SE associated with this iterative scheme takes the same steps. First, we can show that the output $\mu^{t,1}_A$ of (4.37) after $i$ iterations is a linear mapping of all $z_\tau$ for $\tau \leq t$. 

84
Theorem 8. Consider the MP algorithm (4.28)-(4.29) and let \( \mu_{t,i}^{L,i} \) be the output of iterations (4.37) with the initializations (4.40) and approximating the solution to (4.32). Assume that a pair of sets of scalars \( \{a^i_j\}_{j=0}^i \) and \( \{b_j^i\}_{j=0}^i \) are almost surely convergent

\[
\lim_{N \to \infty} a^i_j = \bar{a}^i_j \quad (4.44)
\]

\[
\lim_{N \to \infty} b^i_j = \bar{b}^i_j \quad (4.45)
\]

for every \( 0 \leq j \leq i \) and for some finite \( \bar{a}^i_j \) and \( \bar{b}^i_j \). Then

\[
\mu_{t,i}^{L,i} = \sum_{\tau=0}^{t} \sum_{k=0}^{(t-\tau)i} v^{L,i}[k](\mathbf{A}\mathbf{A}^T)^k z_{t,tau}, \quad (4.46)
\]

where \( v^{L,i}[k] \) is defined in Appendix D and, in the limit \( \lim_{N \to \infty} \), is a function of only \( v_{w} \) and the cross-correlation matrix \( \Psi_t \) from (4.7).

Proof. See Appendix D

In Appendix E we derive the exact expression for \( \bar{a}^i_j \) and \( \bar{b}^i_j \) that satisfy (4.44)-(4.45) for WS-CG, while for WS-GD they are the same as their finite dimensional versions, as follows from Section 4.2.2.

While the models (4.9) and (4.46) are structurally similar, the scalar function \( r^{L,i}[k] \) in the former and \( v^{L,i}[k] \) in the latter are different due to the difference of the SLE being approximated as discussed in Section 4.2.1. Nevertheless, we can follow the same steps as in Section 4.1.1, plug the model (4.46) into (4.30) and into

\[
v^{L,i}_{A \to B} = \lim_{N \to \infty} \frac{1}{N} \|x^{L,i}_{A \to B} - \mathbf{x}\|^2 \quad (4.47)
\]

to obtain closed-form solutions for \( \gamma^{L,i,\tau,i}_{A,B} \) and \( v^{L,i}_{A \to B} \). These results are formally presented in the following theorem

Theorem 9. Consider the LM MP algorithm (4.28)-(4.29) and let \( \mu^{L,i}_{A} \) be the output of iterations (4.37) with the initializations (4.40) and approximating the solution to (4.32). Then, under Assumptions B1-B3 from page 42, the scalar \( \gamma^{L,i,\tau,i}_{A,B} \)
Warm-starting in MP

from (4.30) satisfies

\[
\lim_{N \to \infty} \gamma_A^{t,\tau,i} = \lim_{N \to \infty} \sum_{k=0}^{(t-\tau)i} v_{\tau}^{t,i}[k] \chi_{k+1},
\]

(4.48)

where

\[
\chi_j = \lim_{N \to \infty} \frac{1}{N} \text{Tr} \left\{ (SS^T)^j \right\}.
\]

(4.49)

Additionally, the variance \(v_{t,i}^{A \to B}\) from (4.47) evolves as

\[
v_{t,i}^{A \to B} \xrightarrow{a.s.} \lim_{N \to \infty} \frac{1}{N} \text{Tr} \left( \frac{\Omega_{t} - \sum_{\tau,\tau' = 0}^{t} \gamma_A^{t,\tau,i} \gamma_A^{t,\tau',i} \psi_{\tau,\tau'}}{N} \right).
\]

(4.50)

with

\[
\Omega_t = \sum_{\tau,\tau'=0}^{t} \sum_{j=0}^{(t-\tau)i} \sum_{k=0}^{(t-\tau')i} v_{\tau}^{t,i}[j] v_{\tau'}^{t,i}[k] (\psi_{\tau,\tau'} \chi_{j+k+2} + v_w \chi_{j+k+1}).
\]

(4.51)

Here, the scalar function \(v_{t,i}^{\tau,j}[j]\) is as in Theorem 8.

Proof. See Appendix E.

One can estimate both the moments \(\chi_j\) of the eigenvalue spectrum of \(AA^T\) and the cross-correlation matrix \(\Psi_t\) similar to the method at the end of Section 4.1.1. However, the practical drawbacks of these results are the same and, therefore, next we propose a variant of Theorem 6 that defines a rigorous iterative model of the vector \(\gamma_A^{t,i}\) where \((\gamma_A^{t,i})_r = \gamma_A^{t,\tau,i}\). For this, define the following recursion

\[
\nu_{t+1}^{i} = \nu_t^{i} + a_t^{i} \eta_t^{i},
\]

(4.52)

\[
\sigma_t^{i+1} = \Psi_t^{-1} \left( \frac{1}{N} Z_t^T \mu_t^{i+1} - \nu_t^{i+1} \mathbf{1}_i \right),
\]

(4.53)

\[
\eta_t^{i+1} = v_w \left( \delta - \frac{\nu_t^{i+1}}{v_q} - \| \sigma_t^{i+1} \|_1 \right) + b_t^{i} \eta_t^{i},
\]

(4.54)

with the initializations \(\nu_0^{0} = 0, \eta_0^{0} = \delta v_w\) for \(t = 0\), and \(\nu_t^{0} = \nu_{t-1}^{0}, \eta_t^{0} = v_w(\delta - \frac{\nu_{t-1}^{0}}{v_q} - \| \sigma_{t-1}^{0} \|_1) + b_{t-1}^{0} \eta_{t-1}^{0}\) for \(t > 0\). The recursion above is structurally similar to the one from Theorem 6 with subtle differences motivated by the change of the SLE being approximated.

With these definitions, we can establish the following asymptotic result for \(\gamma_A^{t,i}\).
Theorem 10. Consider the MP algorithm (4.28)-(4.29) and let $\mu^{t,i}_A$ be the output of iterations (4.37) with the initializations (4.40) and approximating the solution to (4.32). Let $\sigma^{i+1}_t$ be computed from (4.52)-(4.54) where $a^i_j$ and $b^i_j$ for every $0 \leq j \leq i$ follow (4.44)-(4.45). Then, under Assumptions B1-B3 from page 42, we have that

$$\lim_{N \to \infty} \gamma^{t,i}_A \overset{a.s.}{=} \lim_{N \to \infty} \sigma^i_t.$$  \hspace{1cm} (4.55)

In practice we suggest to use the empirical version of (4.55) where $v_w$, $v_q$, and $\Psi_i$ are substituted by their finite dimensional estimated versions. Meanwhile, the practical estimator of $v^{t,i}_{A \to B}$ remains the same as in (4.19) since it is built upon the fundamental property (4.17) that holds in every LM MP algorithm.

4.2.5 Additional computational costs associated with Long-Memory MP

In Section 3.3 we discussed the computational side of CG-VAMP algorithm and how adaptively choosing the right number of CG iterations at every outer-loop iteration affects the time required for CG-VAMP to converge. There, we mentioned and later numerically confirmed in Section 3.4.2 that choosing smaller $i$ leads to an increased $t$ required for CG-VAMP to converge. Since at each $t$ one needs to execute not only the linear function $g^{t}_A$, but also the denoiser $g^{t}_B$, which usually represents a substantial part of the computation complexity of each outer-loop iteration, choosing too few inner-loop iterations leads to a slower time-wise convergence compared to the adaptive choice of $i$. The same nuance should be taken into account when one uses a long-memory method for approximating the LMMSE. This immediately highlights an inherent drawback of MP algorithms like MAMP or the related LM algorithm like CAMP [93] and the rotationally invariant AMP [63, 100], where there is a fixed number of inner-loop iterations and, in fact, the smallest case: $i = 1$. As a result, this kind of MP algorithm might require to be run for hundreds or thousands of outer-loop iterations (see the simulation experiments below and from the MAMP work [14]) and this would require executing the denoiser that many times and might lead to slow time-wise convergence and potential instability of MP (see the conclusion chapter, Chapter 6).
Warm-starting in MP

However, this is not the only computational nuance of LM MP one should consider. The second one concerns the complexity of computing the long memory Onsager correction \( \sum_{\tau=0}^{t} \gamma_{A}^{T} x_{B \rightarrow A}^{\tau} \) in (4.28) at every \( t \). Even if implementing the denoiser had a negligible computation cost, updating the vector \( r_{t} \) still requires computing \( t \) summations of \( N \)-dimensional vectors. Since for large-scale inverse problems we assume that \( A \) has a fast matrix-vector multiplication with a cost \( O(N \log N) \), for large \( t \) computing this Onsager correction becomes as expensive as implementing multiple matrix-vector products with \( A \). Note that this Onsager correction is designed not to reduce the intrinsic variance but to keep the algorithm stable. As a result, one might benefit from reallocating these computational resources to the linear function \( g_{A}^{t} \) and do more inner-loop iterations, which would avoid doing too many outer-loop iterations. Again, this option is not possible in case of MAMP, CAMP or the rotationally invariant AMP, while the proposed WS-CG-VAMP and other general LM MP algorithms utilizing the unified approximation framework do provide such an opportunity. The following table summarizes the advantages of an LM MP algorithm utilizing the proposed unified approximation framework over other LM MP algorithms.

<table>
<thead>
<tr>
<th></th>
<th>LM MP with the unified WS approximation</th>
<th>MAMP, CAMP, rotationally invariant AMP</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Flexibility of the iterative approximation of LMMSE</strong></td>
<td>Can use many inner-loop iterations</td>
<td>Limited to 1 inner-loop iteration</td>
</tr>
<tr>
<td><strong>Capability to do adaptive number of the inner-loop iterations</strong></td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td><strong>Resource allocation</strong></td>
<td>To reduce SE</td>
<td>To correct SE</td>
</tr>
</tbody>
</table>

Lastly, we provide a quantitative comparison of the computational and memory costs of implementing WS-CG-VAMP and CG-VAMP. As we discussed in Sections 3.1 and 3.2, the total computational cost of estimating \( \gamma_{A}^{t,i} \) and \( v_{A \rightarrow B}^{t,i} \) for CG-VAMP is of order \( O(iM) \), while the memory cost is negligible. For WS-CG-VAMP, the cost of estimating \( v_{A \rightarrow B}^{t,i} \) through (4.19) is \( O(N) \), while estimating the
whole set of $t$ divergences $\{\gamma^{t,\tau,i}_A\}_{\tau=0}$ requires $O(tiM)$ operations ($t$ times more than for zero-initialized CG). Additionally, both CG-VAMP and WS-CG-VAMP require doing the Onsager correction, which for the former case requires storing and adding a single $N$ dimensional vector, while for the latter case it is storing and summing $t$ of such vectors. Lastly, WS-CG requires computing an additional matrix-vector multiplication $W_t \mu^{t-1,i}_A$ as a part of the initialization. At the same time, implementing the zero-initialized CG and WS-CG routine cost the same computationally- and memory-wise. The computational and memory costs of the linear part of CG-VAMP and WS-CG-VAMP are summarized in Table 4.2, where we used the fact that $M = \delta N$ to relate $O(M)$ and $O(N)$. The demonstrated results highlight the fact that the costs of every next iteration of CG-VAMP stays the same, while for WS-CG-VAMP every next iterations costs more.

Table 4.2: Computational and memory costs of the linear part of CG-VAMP and WS-CG-VAMP

<table>
<thead>
<tr>
<th></th>
<th>CG-VAMP</th>
<th>WS-CG-VAMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computational cost</td>
<td>$O((i \log N + i\delta)N)$</td>
<td>$O((i \log N + ti\delta + t)N)$</td>
</tr>
<tr>
<td>Memory cost</td>
<td>$O(N)$</td>
<td>$O(tN)$</td>
</tr>
</tbody>
</table>

4.3 Numerical Experiments

In this section we compare MP algorithms utilizing different warm-starting techniques and different forms of Onsager correction, verify the validity of the SE of WS-CG-VAMP and compare several proposed algorithms against VAMP, CG-VAMP and MAMP algorithms. We will compare all the mentioned algorithms in the context of an inverse problem as described in Section 3.4.1. For all the algorithms, unless stated otherwise, we use the same update of $x^t_{B\rightarrow A}$ as discussed in Section 3.4.1 and use the same computing platform as mentioned in Section 3.4

4.3.1 Comparison of different WS-CG-VAMP algorithms

First, we numerically compare several practical versions of WS-CG-VAMP that utilize different types of estimators for $\gamma^{t,\tau,i}_A$ that were discussed in Sections 2.4.1,
3.1, 4.1.1 and 4.1.3. To improve the stability of algorithms, all WS-CG-VAMP use the proposed variance $v_{A\rightarrow B}^{t,i}$ estimator based on (4.19). In this setup, we compare two pairs of WS-CG-VAMP. The first pair uses a single correction scalar $\hat{\gamma}_t^A$ and the full set of correction scalars $\{\hat{\gamma}_t^{\tau,i} \}_{\tau=0}^t$ estimated based on (2.140) and (4.10) respectively. For these two algorithms, we estimate the moments $\chi_j$ as in the experiment considered in Section 3.4.1 and we refer to these algorithms as WS-CG-VAMP A.single and WS-CG-VAMP A.full. In the same way, we have implemented a pair of WS-CG-VAMP with a single and the full set of correction scalars estimated based on (3.10) and (4.25) respectively. We refer to them as WS-CG-VAMP B.single and WS-CG-VAMP B.full, and for WS-CG-VAMP B.full we estimate the cross-correlation scalars $\psi_{\tau,\tau'}$ for $\tau, \tau' \leq t$ through (4.13). In Table 4.1 we list the four WS-CG-VAMP algorithms and their respective correction strategies. Additionally, we implement CG-VAMP B from Section 3.4.1 and VAMP for comparison. We evaluate the performance of all the algorithms for different condition numbers $\kappa(A) = \{10^2, 10^3, 10^4\}$.

Table 4.3: A summary of four types of WS-CG-VAMP with different sets of Onsager corrections and methods of estimating those corrections.

<table>
<thead>
<tr>
<th></th>
<th>WS-CG-VAMP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A.single</td>
</tr>
<tr>
<td>Set of divergences</td>
<td>single scalar $\hat{\gamma}_t^A$</td>
</tr>
<tr>
<td>Estimator of divergences</td>
<td>(2.140)</td>
</tr>
</tbody>
</table>

In the context described above, WS-CG-VAMP A.full diverges at iteration $t = 4$ for the three condition numbers, which we attribute to the fact that this estimator is sensitive to the error in estimating the moments $\chi_j$, the variances $v_w$ and the cross-correlation matrix $\Psi_t$. The other algorithms remain stable throughout the iterations and their NMSE averaged over 10 realizations is shown in Figure 4.1. As seen from the plot, WS-CG-VAMP B.single and WS-CG-VAMP B.full demonstrate almost identical performance and a consistent progression over $t = (1, ..., 25)$, although at some point we observe (not shown here) that WS-CG-VAMP B.single diverges, which we attribute to the lack of Onsager corrections.
Warm-starting in MP

Figure 4.1: The NMSE of VAMP, CG-VAMP B and different versions of WS-CG-VAMP versus outer-loop iteration $t$ for different condition numbers $\kappa(A)$ of $A$.

and violation of the SE as discussed in Section 4.1.3. The WS-CG-VAMP A.single is stable as well, although its fixed point is inferior to those of the previous two algorithms. The inferior reconstruction result might be partially explained by the fact that the estimator (2.140) represents a method that is blind to the actual data in the algorithm, contrary to (3.10) that iteratively extracts the information from the vectors $z_t$ and $\mu^{t,i}_{A}$ generated in the algorithm. Lastly, we see from the plot that provided the same amount of resources, the WS-CG-VAMP algorithm gets much closer to the fixed point of VAMP compared to CG-VAMP and in the limit $M = \delta N \to \infty$ can potentially converge to the fixed point of VAMP as $t \to \infty$ as suggested by Theorem 7.

4.3.2 The SE for WS-CG in WS-CG-VAMP

Next we would like to confirm the validity of the evolution model (4.11) of $v^{t,i}_{A \to B}$ in the WS-CG-VAMP algorithm. To measure the accuracy of the estimate $\hat{v}^{t,i}_{A \to B}$, we compute the oracle variance $v^{t,i}_{A \to B}$ and evaluate the corresponding normalized error $\frac{(v^{t,i}_{A \to B} - \hat{v}^{t,i}_{A \to B})^2}{(v^{t,i}_{A \to B})^2}$. We consider two cases: when $i = 1$ and when $i = 5$. For $i = 1$, the normalized error of estimating $v^{t,i}_{A \to B}$ averaged over 10 realizations is shown to the left in Figure 4.2. As seen from the plot, the estimator of $v^{t,i}_{A \to B}$ based on (4.11) accurately predicts the true magnitude of the error $\frac{1}{N}||x^{t,i}_{A \to B} - x||^2$. For the next experiment where we set $i = 5$, we increase the precision of MATLAB calculations to 64 digits instead of the standard 16, since the maximum order of
the spectral moments grows as \((2ti + 2)\) and those numbers quickly grow beyond the standard precision (see the discussion at the of Section 4.1.1). For WS-CG-VAMP with \(i = 5\), the accuracy of the SE’s prediction is shown to the right in Figure 4.2, where at iteration 13 the estimator diverges due to lack of precision (this can possibly be solved by using more than 64 digits).

4.3.3 WS-CG-VAMP and WS-GD-VAMP

Next we conduct a similar experiment as in Section 4.3.1 but now we compare different versions of WS-CG-VAMP and WS-GD-VAMP algorithms from Section 4.2.2. Here, in the update for \(x^t_{B \rightarrow A}\) we use BM3D\(^3\) denoiser and the polynomial divergence \(\gamma^t_B\) estimator from Chapter 5. In this context, we compare the stability of two pairs of WS-CG-VAMP and WS-GD-VAMP utilizing WS-CG and WS-GD from Section 4.2.2 with \(i = 1\) and \(i = 10\). The first pair uses (4.48) and (4.50) from Theorem 9 with the oracle values of \(\chi_j\), \(v_w\) and \(\psi_{\tau,\tau'}\) replaced by their estimated versions \(\hat{\chi}_j\), \(\hat{v}_w\) and \(\hat{\psi}_{\tau,\tau'}\). We refer to this pair as WS-CG-VAMP A and WS-GD-VAMP A respectively. The second pair, which will be referred as WS-CG-VAMP

\(^3\)The BM3D library used throughout the simulations can be downloaded from the website of the authors of the denoiser http://www.cs.tut.fi/ foi/GCF-BM3D/. For this particular implementation we used the ‘profile’ to be ‘np’. 

92
Warm-starting in MP

$B$ and $WS-GD-VAMP$ $B$, utilizes the alternative estimators based on (4.55) and (4.19). A summary of the two pairs of MP algorithms is shown in Table 4.4.

Table 4.4: A summary of two pairs of WS-CG-VAMP and WS-GD-VAMP with different estimators for $v_h$ and $\{\hat{\gamma}_A^{t,\tau,i}\}_{\tau=0}^t$.

<table>
<thead>
<tr>
<th></th>
<th>WS-CG-VAMP</th>
<th>WS-GD-VAMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Estimators of $v_h$</td>
<td>(4.50)</td>
<td>(4.19)</td>
</tr>
<tr>
<td>Estimators of ${\hat{\gamma}<em>A^{t,\tau,i}}</em>{\tau=0}^t$</td>
<td>(4.48)</td>
<td>(4.55)</td>
</tr>
</tbody>
</table>

From the numerical experiments we observed that both WS-GD-VAMP A and WS-GD-VAMP B with $i = 1$ diverge and to stabilize the algorithms we employ the same damping strategy of length $l = 2$ for $x_{B\rightarrow A}^{t+1}$ as has been proposed to use in MAMP [14]. Here, and further in MAMP, the damping of length $l$ computes the following linear combination of the current and the last $\tau_{max} = \min(t, l)$ updates [14]

$$x_{B\rightarrow A}^{t+1} = \zeta_t x_{B\rightarrow A}^{t+1} + \sum_{\tau=1}^{\tau_{max}} \zeta_{t,\tau} x_{B\rightarrow A}^{t-\tau_{max}+\tau},$$

(4.56)

where $x_{B\rightarrow A}^{t+1}$ is the undamped update as in (4.4), while the weights $\zeta_{t,\tau}$ for $\tau \in (1, ..., \tau_{max} + 1)$ are constructed to minimize the variance of $q_{t+1} = x_{B\rightarrow A}^{t+1} - x$. For this, one needs to construct the cross-correlation matrix $V_t \in \mathbb{R}^{(\tau_{max}+1, \tau_{max}+1)}$ of the error vectors $(q_t, \tilde{q}_{t+1}, ..., q_{t}, \tilde{q}_{t+1})$. In particular, we have

$$(V_t)(\tau, \tau') = \frac{1}{N} q_{(t-\tau_{max}+\tau)}^T q_{(t-\tau_{max}+\tau')}$$

(4.57)

for $1 < \tau, \tau' \leq \tau_{max}$, while the last row and column of $V_t$ is formed as

$$(V_t)(\tau_{max}+1, \tau') = (V_t)(\tau, \tau_{max}+1) = \frac{1}{N} \tilde{q}_{t+1}^T q_{(t-\tau_{max}+\tau)}$$

(4.58)

for $1 < \tau < \tau_{max}$ and

$$(V_t)(\tau_{max}+1, \tau_{max}+1) = \frac{1}{N} \|\tilde{q}_{t+1}\|^2$$

(4.59)

This matrix can be estimated in the same way as the matrix $\Psi_t$ in Section 4.1.1. Once $V_t$ is formed, we construct the damping coefficients $(\zeta_t)_\tau = \zeta_{t,\tau}$ as

$$\zeta_t = \frac{V_t^{-1}}{1^TV_t^{-1}1}$$

(4.60)

93
First, we illustrate the impact of damping and plot the NMSE of several algorithms of \( B \) variant with and without damping in Figure 4.3. As we see from the plot, the intrinsic variance of the WS-GD-VAMP algorithm without damping (this corresponds to \( l = 0 \)) consistently increases and the algorithm quickly diverges, even though the MP algorithm uses the correct set of Onsager correction terms at each outer-loop iteration. Adding the damping of even a minimal length \( l = 1 \) makes the algorithm stable although still slower then with the damping of length \( l = 2 \). Further increasing the damping length does not seem to improve the convergence properties of WS-GD-VAMP based on the numerical experiments. Similarly, we plotted the performance of WS-CG-VAMP algorithms with \( i = 1 \) inner-loop iterations and with different damping lengths. As it is seen, the impact of the damping on the dynamics of WS-CG-VAMP is almost negligible and this tendency has been observed in all the simulation experiments with different condition numbers, measurement noise level, types of denoiser and data (not shown here). It is still unclear why an MP algorithm with WS-CG with \( i = 1 \) and no damping performs almost as efficiently as with damping, and why it is not the case for WS-GD with \( i = 1 \). However, considering the approximation method from MAMP corresponds to a particular variant of WS-GD with \( i = 1 \) and MAMP is also unstable with no damping [14], the reasons for the instability of the two algorithms should be the same. Interestingly, if we increase the number of inner-loop iterations for WS-GD-VAMP to \( i = 10 \), then the damping makes almost no effect on the performance. Answering why this is the case is left for further work.

Next, we compare the NMSE performance of the \( A \) and \( B \) variants of the WS-CG-VAMP and WS-GD-VAMP algorithms in the context of the inverse problem considered in Section 4.3.1. The results are averaged over 10 realizations and shown in Figure 4.4. First, as seen from the plot, the \( A \) version of both the algorithms eventually becomes unstable, which we relate to high sensitivity of the estimators based on (4.48) and (4.50) to the estimation error of the parameters \( \nu_w \) and \( \psi_{\tau,\tau'} \) and of the spectral moments \( \chi_j \). Second, we see that the \( B \) variants of MP consistently reduce their intrinsic uncertainty after each iteration irrespective of the number of inner-loop iterations \( i \), which, theoretically, only affects the rate of convergence when \( N \) is sufficiently large. Lastly, we observe that using WS-CG leads to faster dynamics compared to using WS-GD, which can be partially
4.3.4 Comparison with MAMP

In the next experiment, we compare WS-CG-VAMP against the optimized version of MAMP that uses the damped update (4.56) with $l = 2$, which is required to ensure stability of the algorithm. Here, we implement two MAMP algorithms: one that uses the oracle moments $\chi_j$ computed through (4.49) assuming access to $SS^T$ and one with the estimated moments $\hat{\chi}_j$ as before. For comparison, we implemented CG-VAMP B from Section 3.4.1 and VAMP. The NMSE of the algorithms averaged over 10 realizations is shown in Figure 4.5.

As demonstrated on the plot, WS-CG-VAMP with $i = 15$ gets very close to the fixed point of VAMP within 20 outer-loop iterations and it takes approximately 41.3 seconds. The same NMSE is achieved by CG-VAMP with $i = 100$, but it requires 50 seconds for the same work. WS-CG-VAMP with $i = 1$ demonstrates consistent, but slower convergence and will take hundreds of iterations to achieve the same accuracy level. Time-wise, it took 38 seconds for the algorithm to get to iteration $i = 25$ and will take much more to converge to the same NMSE.
Figure 4.4: NMSE over iteration number $t$ for the LM MP algorithms (4.28)-(4.29) with different WS approximators.

Similar dynamics are exhibited by MAMP with the access to the oracle moments $\chi_j$, while the practical version of the same algorithm fails to follow the SE model and converges to an inferior fixed point. Both MAMP algorithms took 85 seconds to get to iteration $t = 25$. Such a big runtime discrepancy with WS-CG-VAMP with $i = 1$ is related to the fact that MAMP requires manipulating numbers of order $10^{50}$ and greater, and for this we used symbolic operations in MATLAB, which slows down the execution. Lastly, the difference in NMSE between MAMP and WS-CG-VAMP with $i = 1$ within the first 10 outer-loop iterations is due to the damping that MAMP is using, while WS-CG-VAMP is not. Adding the damping to WS-CG-VAMP removes such a gap.

4.4 Conclusions

In this chapter we proposed a series of theoretical ideas for upscaling VAMP while achieving the same accuracy of the final estimate, and methods for stably implementing those ideas in practice. First, we introduced the warm-starting extension of CG for the MP context and discussed why reusing information from the previous iterations might lead to improved dynamics of MP. Then, we discussed the theoretical aspect of such MP algorithms, introduced the long-memory version of
the update of $x^t_{A \rightarrow B}$, rigorously derived the set of divergences $\{\gamma^t_{A,\tau,i}\}_{\tau=0}^t$ of WS-CG used to form the Onsager corrections and derived the multi-dimensional SE associated with WS-CG. Then, we extended the ideas from Section 3.1 to the context of WS-CG and proposed a stable way of estimating the divergences $\gamma^t_{A,\tau,i}$ and derived a practical estimator of $v^t_{A \rightarrow B}$ that can be used within any MP algorithm considered in this work (apart from AMP). We finished the part related to WS-CG-VAMP by proposing a simplified version of the update $x^t_{A \rightarrow B}$ where we used WS-CG and only one Onsager correction. Even though theory shows that this simplification violates the rigorous structure of MP, the numerical experiments demonstrated that such an algorithm remains stable for many outer-loop iterations $t$.

The final theoretical part of this chapter was devoted to developing the generalized WS iterative framework for approximating LMMSE within MP algorithms. The proposed framework includes WS-CG as a special case, as well as the WS Gradient Descent (WS-GD), besides other possible algorithms. We prove that a fixed point of MP with an instance of this framework is a fixed point of VAMP and this result is invariant to the number of inner-loop iterations. As a result, we obtain a method for designing an MP algorithm that would be able to solve more
general inverse problems than AMP while preserving its computational complexity and (conjectured) optimality. Finally, this framework allowed us to show that the approximation method used in the recently proposed MAMP algorithm [14, 104] is a special case of our framework. To complete the unified approximation framework, we derived the closed-form solution to the set of Onsager correction terms and the SE for this framework as well as practical methods of implementing them within MP. We finished this chapter with a series of numerical experiments demonstrating the advantages of using the WS approximation tools within MP. However, not only the linear part of VAMP can be accelerated, but also the denoising part, which is the primal focus of the next chapter.
Chapter 5

Divergence estimation in Message Passing algorithms

In the previous chapters we discussed different methods for recovering a signal $x \in \mathbb{R}^N$ from a set of linear measurements

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

(5.1)

where $\mathbf{w} \in \mathbb{R}^M$ is Gaussian measurement noise vector and $\mathbf{A} \in \mathbb{R}^{M \times N}$ is a measurement matrix that is assumed to be available. In this thesis we mostly focused on the family of MP algorithms that includes such examples as AMP [9], OAMP [79], VAMP [10, 85], CG-VAMP [92, 106, 107], CAMP [93, 108], MAMP [14, 104], rotationally invariant AMP [63, 100] and the proposed WS-CG-VAMP algorithms, besides others. In the following, we refer to all of these algorithms as Scalable Message Passing (SMP).

On a general level, every SMP algorithm refines an intrinsic estimate $\hat{x}$ of the signal $x$ by iterating between the following two steps: 1) a “linear” step that extracts the information about $x$ from $y$ to make the intrinsic estimate $\hat{x}$ consistent with the measurements; 2) a denoising step that imposes some prior knowledge about the signal $x$ on the intrinsic estimates $\hat{x}$. The above mentioned SMP algorithms mostly differ in the linear step, while the denoising step is kept the same. In particular, in the previous chapters we discussed how VAMP constructs the linear step using the optimal yet not scalable LMMSE estimator and how MAMP, CAMP, rotationally invariant AMP and the proposed WS-CG-VAMP try
Divergence estimation in Message Passing algorithms

to approximate the LMMSE estimator in a scalable manner while preserving the
countered Bayes-optimality of VAMP under Assumptions B1-B3 from page 42.

In this chapter, we will focus on the second part of SMP – the denoising step.
In particular, we will focus on a specific sub-problem that arises in every SMP
algorithm: estimating the divergence of a denoiser to form the so-called Onsager
correction that guarantees the important asymptotic properties of SMP. In the
literature on SMP algorithms [58, 69, 106, 109, 110, 111] and others, the only sug-
gested method for estimating the divergence of a Plug-and-Play (PnP) denoiser,
such as Non-Local Means (NLM) [67], BM3D [68], Denoising CNN (DnCNN) etc,
is the Black-Box Monte Carlo (BB-MC) method [1] that we briefly discussed in
Section 3.4. In Chapters 3-4 we followed this tradition and used the BB-MC esti-
mator to implement different variants of CG-VAMP, WS-CG-VAMP and MAMP.
This method computes an estimate of the divergence of a function \( f(x) \) that admits
a well-defined second-order Taylor expansion by executing this function again at
the points \( x + \epsilon n \) with the scalar \( \epsilon \) approaching zero and where \( n \) is a zero-mean
i.i.d. random vector with a unit variance and finite higher order moments. Then
one can show that the divergence \( \frac{1}{N} \nabla_x \cdot f(x) \) of \( f \) is equivalent
to [1]

\[
\frac{1}{N} \nabla_x \cdot f(x) = \lim_{\epsilon \to 0} \frac{1}{N} \sum_{i=1}^{N} \frac{\partial f(x)}{\partial x_i} \tag{5.2}
\]

To approximate the expectation operator in (5.2), one can use MC trials and
implement the inner product inside of the expectation multiple times and average
the results. However, as pointed out in [1], if \( f \) affects \( x \) locally, i.e. the Jacobian
of \( f \) is “approximately” diagonal, the variance of the random term inside of the
expectation operator in (5.2)

\[
\hat{\alpha}(\epsilon) = \frac{1}{N} n^T \left( \frac{f(x + \epsilon n) - f(x)}{\epsilon} \right) \tag{5.3}
\]
scales as \( O\left(\frac{1}{N}\right) \). Then, one can use the law of large numbers to show that
\( \lim_{\epsilon \to 0} \hat{\alpha}(\epsilon) \) converges to (5.2) as \( N \to \infty \). This suggests that when \( N \) is suffi-
ciently large, one can obtain a satisfactory accuracy of divergence estimation with
only a single MC trial. These results were partly generalized in [1] to weakly-
differentiable functions \( f \).
Divergence estimation in Message Passing algorithms

While this approach provides a practical method for the divergence estimation of PnP denoisers and often leads to stable dynamics of SMP algorithms, it has two drawbacks. First, it assumes that the chosen denoiser $f$ admits a well-defined second-order Taylor expansion or, at least, is weakly-differentiable, which is not the case for denoisers like BM3D, which involves non-linear operations like hard-thresholding that does not meet either of the criteria [112]. This has several consequences. First, one can no longer use very small values of $\epsilon$ as in this case the estimator (5.2) becomes unstable [1]. As a result, often one needs to empirically tune the scalar $\epsilon$ for each denoiser individually to ensure the stability of the divergence estimator. In the numerical experiments in the previous chapters, we chose $\epsilon$ as in (3.19),

$$\epsilon = \epsilon_0 \min\left(v'_{A \to B}, \frac{1}{N} \|x'_{A \to B}\|_1\right) + \epsilon,$$

where $\epsilon_0$ is chosen based on the denoiser used. For example, for the BM3D denoiser we have found $\epsilon_0 = 0.1$ to produce a better estimation accuracy, while DnCNN tends to demonstrate better results with $\epsilon_0 = 0.01$. Importantly, the resulting scalar $\epsilon$ is much greater then the value dictated by the machine precision, which would be the ideal choice (given the denoiser is sufficiently differentiable), since we want $\epsilon$ to be as small as possible. Therefore, having a relatively large $\epsilon$ and the violation of the differentiability assumption of $f$ might lead to the reduced accuracy of estimation.

However, we would like to note that in the previous chapters, the analysis and derivation of several SMP algorithms was conducted under either Assumption A3 from page 26 or Assumption B2 from page 42 stating that the chosen denoiser $g$ is (uniformly) Lipschitz continuous. This assumptions does not hold for the denoisers such as BM3D, but as pointed out in [69] and has been numerically demonstrated in many works [58, 69, 94, 110, 111, 113, 114] and in this thesis, using such denoisers within MP algorithms leads to stable performance which is accurately predicted by the SE. Therefore, although non-rigorously, one could use such non-differentiable denoisers within SMP and wish to estimate their divergence efficiently.

The second problem with the BB-MC method is that it requires executing the denoiser an additional time or even multiple times, if one needs higher accuracy of
the estimate. When the dimension of the inverse problem is large, as in modern computational imaging tasks, executing powerful denoisers can be the dominant cost of the algorithm and it is desired to execute it as infrequently as possible.

In this chapter we study the dynamics of SMP algorithms and develop two rigorous asymptotic models for the divergence of a PnP denoiser used within the algorithm. These models lead to two divergence estimation techniques that neither require additional executions of the denoisers, nor empirical tuning. The first method works in a complete black-box fashion and has a minimal computational cost dominated by two inner-products of $N$-dimensional vectors, but empirically has inferior estimation accuracy compared to BB-MC and the second method. The second method also uses only the information generated within any SMP algorithm, has a computational complexity dominated by one matrix-vector product with $A$ and has a similar or superior accuracy of the divergence estimation compared to the hand-tuned BB-MC method. When an SMP algorithm incorporates a powerful denoiser such as BM3D, using the proposed methods for divergence estimation instead of BB-MC leads to almost halving the computational time of the algorithm. At the end of the chapter, we numerically compare the proposed methods against the hand-tuned BB-MC method in the context of AMP, VAMP, CG-VAMP and WS-CG-VAMP used for recovering natural images from compressed measurements.

5.1 SMP algorithms and their main asymptotic properties

Before introducing the main ideas behind the proposed divergence estimation methods, we would like to define the context for further analysis. The SMP algorithms mentioned at the beginning of the previous section can all be mapped into the following iterative scheme [60]

$$r_t = \frac{1}{C_r} \left( A^T f_t(S_{t+1}, y) - \sum_{\tau=0}^t \gamma_t \gamma\tau s_{\tau} \right)$$  \hspace{1cm} (5.5)$$

$$s_{t+1} = \frac{1}{C_s} \left( g_r(r_t) - r_t \alpha_t \right),$$  \hspace{1cm} (5.6)$$
Divergence estimation in Message Passing algorithms

which is initialized with \( s_0 = 0 \) and where \( S_{t+1} = (s_0, \ldots, s_t) \). Structurally, this MP algorithm is identical to the one considered in Section 4.2.1, but here we choose a general notation to emphasize that the further results hold for any SMP, not only the ones derived from EP. Here, the step \( r_t \) involves a, possibly long-memory, function \( f_t \) whose role depends on the chosen SMP algorithm and how it processes the residual vector

\[
z_t = y - A s_t. \tag{5.7}
\]

In the previous chapters we saw that the function \( f_t \) can be chosen to be the LMMSE estimator

\[
f_t(S_{t+1}, y) = W_t^{-1} z_t, \tag{5.8}
\]

where

\[
W_t = v_w I + v_q A A^T \tag{5.9}
\]

with \( v_q \) modeling the variance of the error vector

\[
q_t = s_t - x. \tag{5.10}
\]

Alternatively, \( f_t \) can represent some function approximating the LMMSE estimator, like the Conjugate Gradient algorithm, Warm-Started CG, Warm-Started Gradient Descent, and many others. In [60], the authors showed that AMP from Section 2.2.2 can also be fitted into the structure (5.5)-(5.6) by defining a certain LM function \( f_t \). In general, \( f_t \) is allowed to be any function with a well-defined Jacobian with respect to all of its input vectors. Besides \( f_t \), the update (5.5) involves up to \( \tau \) Onsager correction terms \( \sum_{\tau=0}^t \gamma^\tau_t s_{\tau} \) – one for each \( s_{\tau} \) the function \( f_t \) depends on. In general, the scalars \( \gamma^\tau_t \) represents the divergence of \( f_t \) with respect to \( s_{\tau} \),

\[
\gamma^\tau_t = \frac{1}{N} \nabla_{s_{\tau}} \cdot A^T f_t(S_{t+1}, y). \tag{5.11}
\]

For more details on how to estimate the set of these Onsager corrections for the linear function \( f_t \), refer to the previous chapters in the thesis. Lastly, the normal-
ization scalar $C^t_r$ in (5.5) is usually computed as
\[ C^t_r = -\sum_{\tau=0}^{t} \gamma^\tau_r, \] (5.12)
except for the AMP case, where it is $C^t_r = 1$.

In this chapter, we will mostly focus on the denoising step, $s_{t+1}$, that involves a denoiser $g_t(r_t)$ that acts on the intrinsic channel $r_t = x + h_t$, where
\[ h_t = r_t - x \] (5.13)
is modeled as a zero-mean i.d.d. Gaussian noise vector independent of $x$. Similarly to the linear step, the denoiser is corrected by the Onsager correction term $r_t \alpha_t$ where the scalar $\alpha_t$ is traditionally viewed as the divergence of the denoiser $g_t(r_t)$
\[ \alpha_t = \frac{1}{N} \nabla_{r_t} \cdot g_t(r_t). \] (5.14)
Additionally, this scalar is used to form the normalization scalar $C^t_s$, which is usually chosen to be $C^t_s = (1 - \alpha_t)$ [10], except for the AMP case, where it is set to $C^t_s = 1$.

When an SMP algorithm is designed as above, the error vectors $h_t$ and $q_t$ from (5.13) and (5.10) possess certain asymptotic properties that we will frequently use in the following analysis. The next theorem summarizes some of them.

**Theorem 11.** [58]: Consider an SMP algorithm (5.5)-(5.6), let Assumptions B1-B3 from page 42 hold and assume the divergence of the denoiser $g_t$ is bounded in the limit as $\lim_{N \to \infty} \alpha_t \in (0, 1)$. Then, for $\tau = 0, 1, \ldots$ and $\tau' = 0, 1, \ldots, \tau$ we have that

1. $h_\tau$ and $q_{\tau'}$ are asymptotically orthogonal
\[ \lim_{N \to \infty} \frac{1}{N} h_\tau^T q_{\tau'} \overset{a.s.}{=} 0 \] (5.15)

and $q_{\tau'}$ satisfies
\[ \lim_{N \to \infty} \frac{1}{N} w^T A q_{\tau'} \overset{a.s.}{=} 0. \] (5.16)
2. \( h_{\tau} \) and \( b_{\tau'} \) follow
\[
h_{\tau} = \tilde{h}_{\tau} + \beta(h_{\tau}) \tag{5.17}
\]
\[
b_{\tau'} = \tilde{b}_{\tau'} + \beta(b_{\tau'}) \tag{5.18}
\]
where \( \tilde{h}_{\tau} \) and \( \tilde{b}_{\tau'} \) are zero-mean i.i.d. Gaussian vectors satisfying
\[
\lim_{N \to \infty} \frac{1}{N} \| \tilde{h}_{\tau} \|^2 \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} \| h_{\tau} \|^2 = v_{h_{\tau}} < \infty \tag{5.19}
\]
\[
\lim_{N \to \infty} \frac{1}{N} \| \tilde{b}_{\tau'} \|^2 \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} \| b_{\tau'} \|^2 = v_{b_{\tau'}} < \infty \tag{5.20}
\]
and the vectors \( \beta(h_{\tau}) \) and \( \beta(b_{\tau'}) \) satisfy
\[
\lim_{N \to \infty} \frac{1}{N} \| \beta(h_{\tau}) \|^2 \overset{a.s.}{=} 0 \tag{5.21}
\]
\[
\lim_{N \to \infty} \frac{1}{N} \| \beta(b_{\tau'}) \|^2 \overset{a.s.}{=} 0. \tag{5.22}
\]

3. In the limit \( N \to \infty \), the matrices
\[
H_{t+1} = (h_0, ..., h_t) \tag{5.23}
\]
\[
Q_{t+1} = (q_0, ..., q_t) \tag{5.24}
\]
are full rank almost surely.

Besides the properties stated in the above theorem, for most MP algorithms, it was shown that there exists a State Evolution that defines the evolution of the intrinsic uncertainty in the algorithm. See the previous chapters for the SE of specific SMP algorithms.

### 5.2 Efficient estimation of the divergence in SMP algorithms

While for most SMP algorithms there are efficient and robust methods for estimating the set \( \{ \gamma_t \}_{t=0}^{T} \) of divergences of \( f_t \), there is only one available black-box method for estimating the divergence \( \alpha_t \) of a general denoiser – BB-MC method [1]. However, this method requires a hand-tuning and additional executions of the denoiser in order to estimate \( \alpha_t \). In this section we develop two theoretical models
for the divergence \( \alpha_t \) within SMP and propose associated estimators that can be computed using only the observed data in the algorithm and do not require additional executions of the denoiser. We begin with an intuition behind the methods and then move to the formal results.

### 5.2.1 Intuition

At the core of the proposed techniques is the following parametrized denoiser and its oracle error

\[
\hat{s}_{t+1}(\hat{\alpha}) = g_t(r_t) - \hat{\alpha}r_t \\
\hat{q}_{t+1}(\hat{\alpha}) = \hat{s}_{t+1}(\hat{\alpha}) - x,
\]

where \( \hat{\alpha} \) is a scalar parameter. Note that when \( \hat{\alpha} = \alpha_t \), is the actual divergence of \( g_t \), (5.25) is an instance of (5.6) with the normalization \( C_s' = 1 \). Additionally, by expanding the error \( q_{t+1} \) in (5.15), we find that

\[
\lim_{N \to \infty} \frac{1}{N} h_t^T q_{t+1} \overset{a.s.}{=} 0 \\
\Rightarrow \lim_{N \to \infty} \frac{1}{N} h_t^T s_{t+1} - \frac{1}{N} h_t^T x \overset{a.s.}{=} 0 \\
\Rightarrow \lim_{N \to \infty} \frac{1}{C_s'} \frac{1}{N} h_t^T \left( g_t(r_t) - \alpha_t r_t \right) \overset{a.s.}{=} 0,
\]

where in the last step we used

\[
\lim_{N \to \infty} \frac{1}{N} h_t^T x \overset{a.s.}{=} 0,
\]

which follows from (5.15) and the fact that \( x = -q_0 \). The result (5.27) implies that the orthogonality (5.15) holds for any finite \( C_s' \neq 0 \), including \( C_s' = 1 \). One can verify that when \( \hat{\alpha} = \alpha_t \), the error vector \( \hat{q}_{t+1}(\alpha_t) \) also follows the other main asymptotic properties of the original vector \( q_{t+1} \) in SMP algorithms, including the fact that \( V^T \hat{q}_{t+1}(\alpha_t) \) acts as a zero-mean i.i.d. Gaussian plus a perturbation whose magnitude almost surely converges to zero in the LSL.

The idea behind our method is to seek such a function \( E(\hat{\alpha}) \) that has a root at \( \alpha_t \) that we could solve for. As we just saw, when \( \hat{\alpha} = \alpha_t \), the error \( \hat{q}_{t+1} \) is asymptotically orthogonal to \( h_t \). Thus, a straightforward example of such a
function would be
\[ E(\hat{\alpha}) = \frac{1}{N} h_t^T \hat{q}_{t+1}(\hat{\alpha}). \] (5.29)

Then, one could recover \( \alpha_t \) by solving \( E(\hat{\alpha}) = 0 \). Unfortunately, this example of \( E(\hat{\alpha}) \) cannot be implemented in practice since it is explicitly formulated in terms of the error vectors that are not available. In this chapter, we use the observed quantities in the algorithm to construct two types of practical functions \( E(\hat{\alpha}) \) that equated to zero can produce an estimate \( \hat{\alpha}_t \) such that
\[ \lim_{N \to \infty} \hat{\alpha}_t \overset{a.s.}{=} \lim_{N \to \infty} \alpha_t. \] (5.30)

Next, we can adapt Corollary 2 from [56] to our form of SMP algorithms (5.5)-(5.6) to show that using such an estimate in SMP algorithms preserves the properties stated in Theorem 11.

**Lemma 2.** c.f. Corollary 2 [56]: Let the assumptions as in Theorem 11 hold. Consider an SMP algorithm (5.5)-(5.6) but where at every iteration \( t \), the divergence \( \alpha_t \) is replaced by a scalar \( \hat{\alpha}_t \) that satisfies (5.30). Define these iterations as
\[
\tilde{r}_t = \frac{1}{C^r_t} \left( A^T r_t(\tilde{S}_{t+1}, y) - \tilde{S}_{t+1} g_t \right) \quad (5.31)
\]
\[
\tilde{s}_{t+1} = \frac{1}{C^s_t} \left( g_s(\tilde{r}_t) - \tilde{r}_t \hat{\alpha}_t \right), \quad (5.32)
\]
where the rest of the components are the same as in (5.5)-(5.6). Then, the results (5.15)-(5.24) from Theorem 11 hold when \( h_r \) and \( q_r \) are replaced by the error vectors
\[
\tilde{h}_t = \tilde{r}_t - x
\]
\[
\tilde{q}_t = \tilde{s}_t - x,
\]
respectively.

**Proof.** See Appendix G. \(\blacksquare\)

This lemma suggests that the main asymptotic properties of an SMP algorithm (5.5)-(5.6) are preserved when \( \alpha_t \) is replaced by an estimate \( \hat{\alpha}_t \) that asymptotically converges to \( \alpha_t \) at every iteration \( t \). Thus, in the following we can focus only on
designing a divergence estimator and showing that it is asymptotically consistent under the assumption that Theorem 11 holds up to iteration $t$.

### 5.2.2 Algebraic divergence estimator

The first class of estimators we propose is a practical extension of the naive and unavailable estimator \( \hat{q}_{t+1} = \hat{s}_{t+1}(\hat{\alpha}) - \mathbf{x} \) and \( \mathbf{h}_t = \mathbf{r}_t - \mathbf{x} \) into (5.29) and noting that

$$
\lim_{N \to \infty} \frac{1}{N} \left( \mathbf{r}_t - \mathbf{x} \right)^T \left( \hat{s}_{t+1}(\hat{\alpha}) - \mathbf{x} \right) = \lim_{N \to \infty} \frac{1}{N} \left( \mathbf{r}_t - \mathbf{x} \right)^T \hat{s}_{t+1}(\hat{\alpha}),
$$

where we used (5.28) to obtain \( \lim_{N \to \infty} \frac{1}{N} \left( \mathbf{r}_t - \mathbf{x} \right)^T \mathbf{x} = 0 \). Still, the above equation involves \( \mathbf{x} \) explicitly, which can be resolved by considering the difference \( \mathbf{r}_t - \mathbf{r}_{t-1} \) instead of \( \mathbf{r}_t \) alone

$$
\lim_{N \to \infty} \frac{1}{N} \left( \mathbf{r}_t - \mathbf{r}_{t-1} \right)^T \left( \hat{s}_{t+1}(\hat{\alpha}) - \mathbf{x} \right) = \lim_{N \to \infty} \frac{1}{N} \left( \mathbf{h}_t - \mathbf{h}_{t-1} - \mathbf{x} \right)^T \left( \hat{s}_{t+1}(\hat{\alpha}) - \mathbf{x} \right)
$$

$$
= \lim_{N \to \infty} \frac{1}{N} \left( \mathbf{h}_t - \mathbf{h}_{t-1} \right)^T \hat{s}_{t+1}(\hat{\alpha}) - \frac{1}{N} \left( \mathbf{h}_t - \mathbf{h}_{t-1} \right)^T \mathbf{x}
$$

$$
\approx \lim_{N \to \infty} \frac{1}{N} \left( \mathbf{h}_t - \mathbf{h}_{t-1} \right)^T \hat{s}_{t+1}(\hat{\alpha}),
$$

(5.33)

where in (5.33) we used the fact that \( \mathbf{x} = -\mathbf{q}_0 \) together with (5.15) to obtain

$$
\lim_{N \to \infty} \frac{1}{N} \left( \mathbf{h}_t - \mathbf{h}_{t-1} \right)^T \mathbf{x} = -\lim_{N \to \infty} \frac{1}{N} \left( \mathbf{h}_t - \mathbf{h}_{t-1} \right)^T \mathbf{q}_0
$$

$$
= \lim_{N \to \infty} -\frac{1}{N} \mathbf{h}_t^T \mathbf{q}_0 + \frac{1}{N} \mathbf{h}_{t-1}^T \mathbf{q}_0 \approx 0
$$

Thus (5.34) result suggests that if we define a scalar function

$$
E_{t+1}(\hat{\alpha}) = \left( \mathbf{r}_t - \mathbf{r}_{t-1} \right)^T \hat{s}_{t+1}(\hat{\alpha})
$$

(5.35)

and equate it to zero, then we can recover \( \hat{\alpha} \) that ensures the orthogonality between \( \hat{q}_{t+1}(\hat{\alpha}) \) and \( \mathbf{h}_t \) and \( \mathbf{h}_{t-1} \). The following theorem summarizes and generalizes this idea.

**Theorem 12.** Consider an SMP algorithm following (5.5)-(5.6). Define a vector

$$
\mathbf{r}_t = \sum_{\tau=0}^{t-1} k^\tau \mathbf{r}_\tau
$$

(5.36)
with some scalar weights $\sum_{\tau=0}^{t-1} k_\tau^t = 1$. Then, under Assumptions B1-B3 from page 42, the inner-product $(r_t - \bar{r}_t)^T r_t$ is asymptotically non-zero and
\begin{equation}
\hat{\alpha}_t = \frac{(r_t - \bar{r}_t)^T g_t(r_t)}{(r_t - \bar{r}_t)^T r_t}
\end{equation}
almost surely converges to the divergence $\alpha_t$ of the denoiser $g_t(r_t)$,
\begin{equation}
\lim_{N \to \infty} \hat{\alpha}_t \overset{a.s.}{=} \lim_{N \to \infty} \alpha_t.
\end{equation}

**Proof.** First, due to the normalization of the weights $k_\tau^t$, we have that
\begin{equation}
r_t - x = \sum_{\tau=0}^{t-1} k_\tau^t r_\tau - \sum_{\tau=0}^{t-1} k_\tau^t x = \sum_{\tau=0}^{t-1} k_\tau^t h_\tau = \tilde{h}_t,
\end{equation}
where we defined a weighted error vector $\tilde{h}_t$. From (5.17) we have that this error vector
\begin{equation}
\tilde{h}_t = \sum_{\tau=0}^{t-1} k_\tau^t \tilde{h}_\tau + \sum_{\tau=0}^{t-1} k_\tau^t \beta(h_\tau)
\end{equation}
\begin{equation}
= \sum_{\tau=0}^{t-1} k_\tau^t \tilde{h}_\tau + \beta(\tilde{h}_t),
\end{equation}
where with a slight abuse of notation we defined a vector
\begin{equation}
\beta(\tilde{h}_t) = \sum_{\tau=0}^{t-1} k_\tau^t \beta(h_\tau).
\end{equation}
Note that because each $\tilde{h}_\tau$ is Gaussian, the first sum in (5.40) is Gaussian as well, while the second sum represents a vector whose magnitude almost surely converges to zero as follows from (5.21). Then, because of the assumption about $g_t$ being uniformly Lipschitz continuous, in the limit we have that
\begin{equation}
\lim_{N \to \infty} \frac{1}{N} \|g_t(x + h_t) - g_t(x + \tilde{h}_t)\|^2
= \lim_{N \to \infty} \frac{1}{N} \|g_t(x + \tilde{h}_t + \beta(h_\tau)) - g_t(x + \tilde{h}_t)\|^2
\leq \lim_{N \to \infty} L \frac{1}{N} \|\beta(h_t)\|^2 \overset{a.s.}{=} 0.
\end{equation}
Similarly, one can confirm that the same results holds for $g_t(x + \tilde{h}_t)$. Then, since
Divergence estimation in Message Passing algorithms

\[ \mathbf{r}_t - \mathbf{r}_t = \mathbf{h}_t - \overline{\mathbf{h}}_t \] as was seen in (5.34), we can use (2.104) to obtain

\[ \lim_{N \to \infty} \frac{1}{N} (\mathbf{r}_t - \mathbf{r}_t)^T g_t (\mathbf{r}_t) = \lim_{N \to \infty} \frac{1}{N} (\mathbf{h}_t - \overline{\mathbf{h}}_t)^T g_t (\mathbf{x} + \mathbf{h}_t) \]

\[ \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} \left( \mathbf{h}_t - \sum_{\tau=0}^{t-1} k^\tau_t \overline{\mathbf{h}}_t \right)^T g_t (\mathbf{x} + \overline{\mathbf{h}}_t) \]

\[ \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} \left( \mathbf{h}_t - \sum_{\tau=0}^{t-1} k^\tau_t \overline{\mathbf{h}}_t \right)^T \mathbf{h}_t \alpha_t \]

\[ \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} (\mathbf{h}_t - \overline{\mathbf{h}}_t)^T \mathbf{h}_t \alpha_t. \quad (5.44) \]

Next, we can use (5.28) to show

\[ \lim_{N \to \infty} \frac{1}{N} (\mathbf{r}_t - \mathbf{r}_t)^T \mathbf{r}_t = \lim_{N \to \infty} \frac{1}{N} (\mathbf{h}_t - \overline{\mathbf{h}}_t)^T (\mathbf{x} + \mathbf{h}_t) \]

\[ \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} (\mathbf{h}_t - \overline{\mathbf{h}}_t)^T \mathbf{h}_t. \quad (5.45) \]

Combining the results (5.44) and (5.45) gives

\[ \lim_{N \to \infty} \frac{1}{N} (\mathbf{r}_t - \mathbf{r}_t)^T g_t (\mathbf{r}_t) \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} (\mathbf{h}_t - \overline{\mathbf{h}}_t)^T \mathbf{h}_t \alpha_t = \alpha_t. \quad (5.46) \]

Lastly, from Theorem 11 we know that in the limit, the matrix \( \mathbf{H}_{t+1} = (\mathbf{h}_0, ..., \mathbf{h}_t) \) is full rank so the inner-product

\[ \frac{1}{N} (\mathbf{h}_t - \overline{\mathbf{h}}_t)^T \mathbf{h}_t = \frac{1}{N} \left( \mathbf{h}_t - \sum_{\tau=0}^{t-1} k^\tau_t \mathbf{h}_\tau \right)^T \mathbf{h}_t \]

is non zero almost surely. Noting that \( \lim_{N \to \infty} \frac{1}{N} (\mathbf{r}_t - \mathbf{r}_t)^T \mathbf{r}_t \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} (\mathbf{h}_t - \overline{\mathbf{h}}_t)^T \mathbf{h}_t \) completes the proof.

In the following, we refer to the estimator based on (5.37) as an algebraic estimator. By equating (5.35) to zero and solving for \( \hat{\alpha} \), one can show that the function \( E_{t+1} \) leads to the algebraic estimator with \( \mathbf{r}_t = \mathbf{r}_t - 1 \). While in the limit (5.37) holds for any set of weights \( k^\tau_t \) as long as the normalization is satisfied, in the finite dimensional case the asymptotic identities used to derive Theorem 12 are no longer exact and an additional stochastic error emerges. At the end of the previous chapter, we discussed the impact of this unmodeled stochastic error in regard to estimating the Onsager corrections for the linear estimator \( \mathbf{f}_t \). Unfortunately, a similar problem arises when we estimate the Onsager correction for the denoiser.
and this stochastic error might be significant if, for example, we use \( \mathbf{r}_t = \mathbf{r}_{t-1} \). In this case, the term \( \frac{1}{N}(\mathbf{h}_t - \mathbf{h}_t')^T \mathbf{x} \) (assumed to be equal in the limit to zero in (5.45)) might have considerable magnitude due to the fact that the magnitude of \( \mathbf{x} \) remains the same throughout the algorithm and might significantly exceed the magnitudes of \( \mathbf{h}_t \) and of \( \mathbf{h}_t' \) that should decay with \( t \). Due to a high difference in the magnitude, a small alignment of these error vectors with \( \mathbf{x} \) would result in a substantial quantity that is ignored in the step (5.45) and that affects the accuracy of the LSL approximation (5.45).

On the other hand, the finite dimensional model of \( \mathbf{h}_t \) deviates from the asymptotic one (5.17) and these deviations accumulate as the algorithm progresses. One of the effects of this error is that the asymptotic identity

\[
\lim_{N \to \infty} \frac{1}{N} \mathbf{h}_t^T \mathbf{g}_t(\mathbf{r}_t) \overset{a.s.}{=} \lim_{N \to \infty} \alpha_t \frac{1}{N} \mathbf{h}_t^T \mathbf{h}_t
\]

used in (5.44) to prove Theorem 12 becomes less accurate for finite \( N \) as the difference between \( t \) and \( \tau \) increases. For this reason we might observe poor quality of the divergence estimates if we use \( \mathbf{r}_t = \mathbf{r}_0 \). The detailed analysis of the optimal choice of weights \( k_t^\tau \) is left for further study, while in this thesis we consider the cases \( \mathbf{r}_t = \mathbf{r}_{t-1} \) and \( \mathbf{r}_t = \mathbf{r}_0 \). The important advantage of these two options is that the computational cost of the resulting algebraic estimators is dominated by the cost of two inner-products of \( N \)-dimensional vectors. Such a low cost allows one to efficiently tune a parametrized denoising using the SURE technique [59] to optimize the performance of the denoising block. The SURE framework was discussed in more details at the end of Section 2.2.4, while combining the algebraic divergence estimator with SURE to optimize the performance of MP algorithms is left for further work.

However, as will be demonstrated in the simulation section at the end of this chapter, these two special cases of the algebraic estimator are sensitive to finite values of \( N \) and \( M \), and might lead to stability issues of SMP algorithms when used as the tool for constructing the Onsager correction for the denoiser. However, if the task of forming the Onsager correction is delegated to some other accurate divergence estimator, then the polynomial estimator could be used to construct a fast estimate of the divergence to, for example, tune a parametrized denoiser using the SURE technique [59]. Recall from Section 2.2.4 that if \( \mathbf{g}_t(\mathbf{r}_t, \theta) \) is parametrized

111
by some $\theta$, then one can optimize the denoiser to minimize the MSE $\mathcal{E}_t(\theta) = \frac{1}{N}\|g_t(r_t, \theta) - x\|^2$ by minimizing the Stein’s Unbiased Risk Estimator $\hat{\mathcal{E}}_t(\theta)$ [1, 59]

$$\hat{\mathcal{E}}_t(\theta) = \frac{1}{N}\|r_t - g_t(r_t, \theta)\|^2 - v_{h_t} + 2v_{h_t}\alpha_t(\theta),$$

(5.49)

which is an unbiased estimator of the true MSE $\mathcal{E}_t(\theta)$ and where $\alpha_t(\theta)$ is the divergence of $g_t(r_t, \theta)$. Then, if one uses the polynomial estimator to construct an estimate of $\alpha_t(\theta)$, the computational cost of executing the SURE is dominated by implementing a few $N$-dimensional inner-products and can be efficiently evaluated multiple times per outer-loop iteration. Yet, for this strategy to be viable, we need a more robust estimator for the Onsager correction to ensure stable performance of SMP. In the next section we present such an estimator.

### 5.2.3 Polynomial divergence estimator

In this section we present another way of constructing a practical function $E(\hat{\alpha})$ that has a root almost surely converging to the divergence $\alpha_t$ of the denoiser $g_t(r_t)$. However, compared to the polynomial estimator that has a single root, the new estimator will have two roots and a special root identification step will be required. To execute this step, we will need a slightly more general formulation than (5.25) and in the following we will construct $E(\hat{\alpha})$ based on the following parametrized function

$$s_{t+1}(\hat{\alpha}, \tau) = g_t(r_t) - \hat{\alpha}(r_t - s_t),$$

(5.50)

which is associated with the corresponding error vector

$$\bar{q}_{t+1}(\hat{\alpha}, \tau) = s_{t+1}(\hat{\alpha}, \tau) - x.$$

(5.51)

Note that $\bar{s}_{t+1}(\hat{\alpha})$ is a special case of $\bar{s}_{t+1}(\hat{\alpha}, \tau)$ when the parameter $\tau$ is set to 0. Another important property of (5.50) is that the asymptotic orthogonality of $\bar{q}_{t+1}(\hat{\alpha}, \tau)$ and $h_t$ implies the orthogonality of $\hat{q}_{t+1}(\hat{\alpha})$ and $h_t$ and vice versa, since

$$\lim_{N \to \infty} \frac{1}{N} h_t \bar{q}_{t+1}(\hat{\alpha}, \tau) = \lim_{N \to \infty} \frac{1}{N} h_t (\hat{q}_{t+1}(\hat{\alpha}) + \hat{\alpha}s_t)$$

$$\xrightarrow{a.s.} \lim_{N \to \infty} \frac{1}{N} h_t \hat{q}_{t+1}(\hat{\alpha})$$

(5.52)
and this result is invariant with respect to $\tau$. Here we used (5.15) and the fact that $x = -q_0$ to show that
\[
\lim_{N \to \infty} \frac{1}{N} h_t^T s_\tau = \lim_{N \to \infty} \frac{1}{N} h_t^T (x + q_\tau) \overset{a.s.}{=} 0.
\] (5.53)

Thus, one can equivalently use (5.50) to derive estimators for $\alpha_t$.

To derive a new estimator of $\alpha_t$, consider the MSE of the parametrized denoiser $s_{t+1}(\hat{\alpha}, \tau)$ and recall that $r_t = x + h_t$ and $s_{t+1}(\hat{\alpha}, \tau) = x + \overline{q}_{t+1}(\hat{\alpha}, \tau)$. Then, after simple algebraic manipulations we obtain
\[
\frac{1}{N} \|s_{t+1}(\hat{\alpha}, \tau) - x\|^2 = \frac{1}{N} \|s_{t+1}(\hat{\alpha}, \tau) - x - h_t + h_t\|^2 \\
= \frac{1}{N} \|s_{t+1}(\hat{\alpha}, \tau) - r_t + h_t\|^2 \\
= \frac{1}{N} \left( \|s_{t+1}(\hat{\alpha}, \tau) - r_t\|^2 + 2h_t^T (s_{t+1}(\hat{\alpha}, \tau) - r_t) + \|h_t\|^2 \right) \\
= \frac{1}{N} \left( \|s_{t+1}(\hat{\alpha}, \tau) - r_t\|^2 + 2h_t^T (\overline{q}_{t+1}(\hat{\alpha}, \tau) - h_t) + \|h_t\|^2 \right) \\
= \frac{1}{N} \left( \|s_{t+1}(\hat{\alpha}, \tau) - r_t\|^2 - \|h_t\|^2 + 2h_t^T \overline{q}_{t+1}(\hat{\alpha}, \tau) \right).
\]

Next, if we define a function
\[
J_{t+1}^1(\hat{\alpha}, \tau) = \frac{1}{N} \left( \|s_{t+1}(\hat{\alpha}, \tau) - r_t\|^2 - \|h_t\|^2 \right),
\] (5.54)
this function would be equivalent to the MSE of the parametrized intrinsic measurement $s_{t+1}(\hat{\alpha}, \tau)$ plus an error,
\[
J_{t+1}^1(\hat{\alpha}, \tau) = \frac{1}{N} \|s_{t+1}(\hat{\alpha}, \tau) - x\|^2 + e_{t+1}^1(\hat{\alpha}),
\] (5.55)
where
\[
e_{t+1}^1(\hat{\alpha}) = -2h_t^T \overline{q}_{t+1}(\hat{\alpha}, \tau).
\] (5.56)

As was discussed before, when $\hat{\alpha} = \alpha_t$, the error vector $\overline{q}_{t+1}(\hat{\alpha}, \tau)$ is asymptotically orthogonal to $h_t$. This implies that in the limit, $J_{t+1}^1(\hat{\alpha}_{t+1}(\alpha_t))$ converges to the MSE of $s_{t+1}(\alpha_t, \tau)$ when $\hat{\alpha} = \alpha_t$, and is additionally corrupted by a non-zero error $e_{t+1}^1(\alpha_t)$ when $\hat{\alpha} \neq \alpha_t$.

At the same time, the same MSE can be observed in a different way. Using the definition of the vector $y$ and the fact that $s_{t+1}(\hat{\alpha}, \tau) = x + \overline{q}_{t+1}(\hat{\alpha}, \tau)$, we can
Divergence estimation in Message Passing algorithms

\[
\lim_{N \to \infty} \frac{1}{N} \|y - A\hat{s}_{t+1}(\hat{\alpha}, \tau)\|^2 = \lim_{N \to \infty} \frac{1}{N} \|w - A\hat{q}_{t+1}(\hat{\alpha}, \tau)\|^2
\]

\[
\overset{a.s.}{=} \delta v_w + \lim_{N \to \infty} \frac{1}{N} \|A\hat{q}_{t+1}(\hat{\alpha}, \tau)\|^2 - \frac{2}{N} w^T A \hat{q}_{t+1}(\hat{\alpha}, \tau).
\]

Next, we can use the conditioning technique [10, 54, 60] for the random matrix \( A \) to study the interaction between \( \hat{q}_{t+1}(\hat{\alpha}, \tau) \) and \( A \). In Appendix I we show that

\[
\lim_{N \to \infty} \frac{1}{N} \|A\hat{q}_{t+1}(\hat{\alpha}, \tau)\|^2 \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} \|\hat{q}_{t+1}(\hat{\alpha}, \tau)\|^2 + \zeta_{t+1}(\hat{\alpha}),
\]

where \( \zeta_{t+1}(\hat{\alpha}) \) depends on the whole history of vectors \( (q_0, ..., q_t) \) when \( \hat{\alpha} \neq \alpha_t \) and almost surely converges to zero for \( \hat{\alpha} = \alpha_t \). Similarly, in Appendix I we show that \( \frac{1}{N} w^T A \hat{q}_{t+1}(\hat{\alpha}, \tau) \) almost surely converges to \( v_w(\alpha_t - \hat{\alpha}) \), which becomes zero for \( \hat{\alpha} = \alpha_t \). Therefore one can define another MSE estimator

\[
J^2_{t+1}(\hat{\alpha}, \tau) = \frac{1}{N} \|y - A\hat{s}_{t+1}(\hat{\alpha}, \tau)\|^2 - \delta v_w,
\]

which is equivalent to

\[
J^2_{t+1}(\hat{\alpha}, \tau) = \frac{1}{N} \|\hat{q}_{t+1}(\hat{\alpha}, \tau)\|^2 + e^2_{t+1}(\hat{\alpha}),
\]

where \( e^2_{t+1}(\hat{\alpha}) \) almost surely converges to zero for \( \hat{\alpha} = \alpha_t \).

The important observation, which we theoretically confirm in Appendix I, about \( J^1_{t+1}(\hat{\alpha}, \tau) \) and \( J^2_{t+1}(\hat{\alpha}, \tau) \) is that their errors, \( e^1_{t+1}(\hat{\alpha}) \) and \( e^2_{t+1}(\hat{\alpha}) \), behave differently for \( \hat{\alpha} \neq \alpha_t \) and both almost surely converge to zero for \( \hat{\alpha} = \alpha_t \). Then, by defining a new function

\[
E_{t+1}(\hat{\alpha}, \tau) = J^1_{t+1}(\hat{\alpha}, \tau) - J^2_{t+1}(\hat{\alpha}, \tau),
\]

one could recover \( \alpha_t \) by finding the appropriate root to \( E_{t+1}(\hat{\alpha}, \tau) \). The following theorem shows that (5.59) corresponds to a particular quadratic polynomial.

**Lemma 3.** Consider an SMP algorithm following (5.5)-(5.6) and let \( v_{h_t} = \frac{1}{N} \|h_t\|^2 \). Then \( E_{t+1}(\hat{\alpha}, \tau) \) from (5.59) is the following quadratic polynomial

\[
E_{t+1}(\hat{\alpha}, \tau) = u_0 + u_1(\tau)\hat{\alpha} + u_2(\tau)\hat{\alpha}^2,
\]
where the scalar coefficients are defined as

\[
\begin{align*}
    u_0 &= \frac{1}{N} \left( \| r_t - g_t \|^2 - v_h - \| y - Ag_t \|^2 + \delta v_w \right) \\
    u_1(\tau) &= \frac{2}{N} (r_t - s_\tau)^T (r_t - g_t - A^T (y - Ag_t)) \\
    u_2(\tau) &= \frac{1}{N} (\| r_t - s_\tau \|^2 - \| A(r_t - s_\tau) \|^2)
\end{align*}
\]

with \( g_t \) used as a shorthand for \( g_t(r_t) \).

\textbf{Proof.} See Appendix H. \hfill \square

Note that the coefficients of the equation (5.60) are formed only from the data that is naturally circulated in any SMP algorithm. The computational cost of evaluating \( u_0, u_1 \) and \( u_2 \) is dominated by implementing two matrix-vector products \( Ag_t(r_t) \) and \( Ar_t \). However, this cost can be reduced by reusing the calculations to form the next update. All the algorithms mentioned at the beginning of this chapter compute the vector

\[
z_{t+1} = y - A s_{t+1}
\]

as part of the function \( f_t \) in (5.5). Using the definition of \( s_{t+1} \), this vector can be equivalently represented as

\[
z_{t+1} = y - A s_{t+1} = y - A g_t(r_t) - \alpha_t A r_t.
\]

Thus, one can reuse the results of the matrix-vector products \( A g_t(r_t) \) and \( A r_t \) to update \( z_{t+1} \) and, consequently, reduce the number of additional matrix-vector products down to 1. Additionally, one can store the \( m \)-dimensional vector \( A s_\tau \) to reuse this result in the implementation of \( u_2 \) and \( u_3 \).

Lemma 3 becomes useful in the light of the following theorem that establishes the asymptotic behaviour of the roots of \( E_{t+1}(\hat{\alpha}, \tau) \)

\textbf{Theorem 13.} Consider an SMP algorithm following (5.5)-(5.6) with \( f_t(s_{t+1}, y) \) having a well-defined Jacobian with respect to every input vector. Let \( \bar{s}_{t+1}(\hat{\alpha}, \tau) \) be defined as in (5.50) with \( \tau \leq t \). Additionally, let \( Q_{t+1} = (q_0, \ldots, q_t) \) and define a vector

\[
\beta_{t+1} = Q_{t+1}^\dagger (g_t(r_t) - x).
\]

\[115\]
Then, under Assumptions B1-B3 from page 42, the function $E_{t+1}(\hat{\alpha}, \tau)$ from (5.59) has two roots $\hat{\alpha}_1$ and $\hat{\alpha}_2$ that follow

$$\lim_{N \to \infty} \hat{\alpha}_1 \overset{a.s.}{=} \alpha_t$$

$(5.63)$

$$\lim_{N \to \infty} \hat{\alpha}_2(\tau) \overset{a.s.}{=} \lim_{N \to \infty} \frac{c_0^t + c_2^t \alpha_t}{c_1^t(\tau)},$$

$(5.64)$

where

$$c_0^t = -2(v_{ht} - v_w + \frac{1}{N}\beta_{t+1}^T Q_{t+1}^T A^T Ah_t)$$

$(5.65)$

$$c_1^t(\tau) = -2\frac{1}{N}q_{\tau}^T A^T Ah_t$$

$(5.66)$

$$c_2^t = v_{ht} - \frac{1}{N}\|Ah_t\|^2.$$  

$(5.67)$

Proof. See Appendix I. 

Then, one way to estimate $\alpha_t$ is by computing the roots to (5.60) and identifying which of the two roots is the correct one. In the following, we refer to this estimator as the polynomial estimator. As will be demonstrated in the simulation section, SMP algorithms with the polynomial estimator demonstrate stable dynamics similar to the BB-MC estimator even for $N$ and $M$ of order $10^4$. To combine the advantages of the algebraic and the polynomial estimators, one could potentially use the algebraic estimator to tune the denoiser via SURE and use the polynomial estimator to compute the final estimate of the divergence $\alpha_t$. This approach would combine the advantages of both methods and result in a fast and efficient way of updating $s_{t+1}$. We will explore this idea in future work.

### 5.2.4 Root identification for the polynomial estimator

While Theorem 13 relates the divergence $\alpha_t$ to one of the roots $\hat{\alpha}_1$ and $\hat{\alpha}_2$ of (5.60), it is still required to identify which of the two roots is the right one. In this subsection, we propose a method for assigning $\alpha_t$ to either $\hat{\alpha}_1$ or $\hat{\alpha}_2$. The idea is based on forming a pair of polynomials, $P_1(\hat{\alpha})$ and $P_2(\hat{\alpha})$, that share one root at $\alpha_t$ and the other two roots would be different by a substantial amount. Let the pair $\hat{\alpha}_1$ and $\hat{\alpha}_2$ and the pair $\hat{\alpha}_3$ and $\hat{\alpha}_4$ be the roots of $P_1(\hat{\alpha})$ and $P_2(\hat{\alpha})$ respectively. Additionally, assume that two roots from different pairs are the same. Then, one
way to form an estimate $\hat{\alpha}_t$ of the common root $\alpha_t$ would be

$$
\hat{\alpha}_t = \frac{\hat{\alpha}_{k^*} + \hat{\alpha}_{s^*}}{2},
$$

(5.68)

i.e. take the average of two roots that are from two different polynomials and that are the closest ones.

In our context, the polynomial $E_{t+1}(\hat{\alpha}, \tau)$ from (5.59) is a function of $\tau$. From (5.63) we know that the first root, $\hat{\alpha}_1$, is invariant with respect to $\tau$, but its not the case for the second root $\hat{\alpha}_2$. Given that there is a pair of indices $\tau \neq \tau'$ such that $\tau, \tau' \leq t$ and satisfies

$$
\lim_{N \to \infty} \frac{1}{N} q^T \tau A^T A_h = \lim_{N \to \infty} \frac{1}{N} q^T \tau' A^T A_h,
$$

(5.69)

we can generate a pair of polynomials $E_{t+1}(\hat{\alpha}, \tau)$ and $E_{t+1}(\hat{\alpha}, \tau')$ that share one root at $\alpha_t$ and have distinct second roots. While identifying when the condition (5.69) holds for a general SMP framework (5.5)-(5.6) is a challenging theoretical task and an interesting open problem, we can test this condition online. In Appendix J we show that finding a pair of indices $(\tau, \tau')$ that follows (5.69) is asymptotically equivalent to finding a pair that follows

$$
\lim_{N \to \infty} \frac{1}{N} z^T \tau (y - A \tau) = \lim_{N \to \infty} \frac{1}{N} z^T \tau' (y - A \tau').
$$

(5.70)

Note that all the elements involved in this condition are available and are computed as a part the main SMP routine so one can test the condition at a negligible computational cost. At the same time, from thorough numerical experiments with different MP algorithms, denoisers $g_t$ and condition numbers $\kappa(A)$, we found out that the condition (5.69) was satisfied by any pair $(\tau, \tau')$ and all of them led to a very similar accuracy of estimating $\alpha_t$ using the polynomial estimator. Motivated by this, in the simulation section coming next we will use the simplest pair $(t, t-1)$, which demonstrated a slightly better estimation accuracy, although this advantage was statistically insignificant.

Based by the above ideas, we make the following heuristic proposal.

**Proposal 1.** Consider an SMP algorithm following (5.5)-(5.6) with $f_t(S_{t+1}, y)$ having a well-defined Jacobian with respect to every input vector. Let $\tau, \tau' \leq t$ be
a pair of indices that follow the condition (5.69). Generate a pair of polynomials $E_{t+1}(\hat{\alpha}, \tau)$ and $E_{t+1}(\hat{\alpha}, \tau')$ associated with the pair $(\tau, \tau')$ as in Lemma 3. Let $\hat{\alpha}_1$ and $\hat{\alpha}_2$ define the roots of the first polynomials and $\hat{\alpha}_3$ and $\hat{\alpha}_4$ define the roots of the second polynomial, respectively. Then, choose an estimate $\hat{\alpha}_t$ of the divergence $\alpha_t$ of the denoiser $g_t(r_t)$ as in (5.68).

5.2.5 Theoretical identification of the root

In the previous section we developed a method for constructing an online test that identifies which of the two roots of the polynomial $E_{t+1}(\hat{\alpha}, \tau)$ from Lemma 3 corresponds to the divergence $\alpha_t$ of the denoiser $g_t$. However, from a series of extensive numerical experiments we have found out that the correct root always corresponded to the one with the minimal magnitude. These results have been observed within AMP, OAMP, VAMP, CG-VAMP and WS-CG-VAMP solving an inverse problem where a measurement matrix $A$ had the condition number ranging from $10$ to $10^9$, the measurement noise variance led to SNR $\frac{\|x\|^2}{\|w\|^2}$ ranging from $10$ to $40$ dB, and when BM3D, DnCNN or NLM denoisers were used. Such an extreme consistency made us wonder whether it is possible to identify the right root theoretically and in this section we partially answer this question.

Before proceeding next, note that the original structure of the polynomial $E_{t+1}(\hat{\alpha}, \tau)$ as a function of $\tau$ was designed to allow to form a pair of polynomials for the identification method discussed in the previous section. However, since now we are interested in the theoretical guarantees, the dependence of $E_{t+1}(\hat{\alpha}, \tau)$ on $\tau$ is redundant and in the following we let $\tau = 0$ and define $E_{t+1}(\hat{\alpha}) = E_{t+1}(\hat{\alpha}, 0)$.

To establish the theoretical connection between the divergence $\alpha_t$ and the two roots $\hat{\alpha}_1$ and $\hat{\alpha}_2$ of the polynomial $E_{t+1}(\hat{\alpha})$, we refer to the following two properties of a generic quadratic polynomial. We know that a quadratic polynomial corresponds to a parabola, which is oriented either up or down and this orientation is uniquely identified based on the sign of $u_3$. Second, if we know the orientation of the parabola, we can uniquely identify the root if we know the sign of the derivative of the polynomial at the desired root. Here, the derivative of the polynomial
Divergence estimation in Message Passing algorithms

from (5.60) corresponds to

\[
\frac{\partial}{\partial \hat{\alpha}} \left( u_1 + u_2\hat{\alpha} + u_3\hat{\alpha}^2 \right) = u_2 + 2u_3\hat{\alpha}.
\] (5.71)

Then, if we worked out the sign of (5.71) at \( \hat{\alpha} = \alpha_t \) and found out that it is, for example, always positive, then \( \alpha_t \) would correspond to the smallest root of (5.60) if \( u_3 \) is positive and to the largest root if \( u_3 \) is negative. The opposite would hold if it turns out that (5.71) at \( \hat{\alpha} = \alpha_t \) is always negative. The following theorem presents the LSL identities for (5.71) at \( \hat{\alpha} = \alpha_t \) for two examples of SMP algorithm (5.5)-(5.6). The first example is VAMP, where the linear \( f_t \) is the LMMSE estimator, \( f_t(S_t, y) = W_t^{-1}z_t \) with \( W_t \) and \( z_t \) defined as in Section 5.1. The second example is the Matched Filter Orthogonal AMP (MF-OAMP) [79], where the linear \( f_t(S_t, y) = z_t \) implements a naive approximation of the LMMSE estimator, i.e. approximates \( W_t^{-1} \) by the identity.

**Theorem 14.** Consider the MF-OAMP and VAMP algorithms equipped with the denoiser \( g_t(r_t) \) and let \( \alpha_t \) be the divergence of \( g_t(r_t) \). Define an unnormalized corrected denoiser

\[
\tilde{g}_{t+1}(r_t) = g_t(r_t) - \alpha_t r_t
\] (5.72)

and its error

\[
\tilde{q}_{t+1} = \tilde{g}_{t+1}(r_t) - x.
\]

Additionally, define an inner-product

\[
\bar{\psi}_t = \frac{1}{N} q_t^T q_{t+1},
\] (5.73)

where \( q_t = s_t - x \) is from iteration \( t \). Then, under Assumptions B1-B3 from page 42, the derivative (5.71) at \( \hat{\alpha} = \alpha_t \) almost surely converges to

- For MF-OAMP:

\[
\lim_{N \to \infty} \frac{1}{2} (u_2 + 2\alpha_t u_3)^{a.s.} = \lim_{N \to \infty} (1 - \chi_2) \left( \bar{\psi}_t - q_t \right),
\] (5.74)

where \( \chi_2 = \frac{1}{N} \text{Tr}\{(AA^T)^2\} \).
Divergence estimation in Message Passing algorithms

- For VAMP:
  \[
  \lim_{N \to \infty} \frac{1}{2} \left( w_2 + 2\alpha t w_3 \right) \overset{\text{a.s.}}{=} \lim_{N \to \infty} \frac{(v_w - v_{h_t})}{v_{q_t}} (\psi_t - v_{q_t}).
  \]
  (5.75)

Proof. See Appendix K. ■

We would like to emphasize that the unnormalized corrected denoiser \( g_{t+1}(r_t) \) uses the exact divergence \( \alpha_t \) in the contrast to the parametrized update \( \hat{s}_{t+1} \) from (5.25), which uses an arbitrary parameter \( \hat{\alpha} \). At the same time, \( g_{t+1}(r_t) \) naturally emerges in the proof of the theorem, while in the actual algorithm we use the update \( s_t \) from (5.6), which assumes an additional normalization by \( C_t^s = \frac{1}{1 - \alpha_t} \).

Theorem 14 implies that deducing the right root of \( E_{t+1}(\hat{\alpha}) \) boils down to analysing the sign of (5.74) for MF-OAMP and (5.75) for VAMP. Unfortunately, analysing these derivatives for a general denoiser \( g_t \) is very challenging and in the following we simplify the problem and consider the case where \( g_t \) is the MMSE denoiser. While for most real data we cannot ensure such an assumption exactly, certain denoisers like DnCNN or BM3D demonstrate nearly optimal performance and behave approximately as the MMSE method.

Since both (5.74) and (5.75) are formed out of two factors, \( \psi_t - v_{q_t} \) and the second one, depending on the MP algorithm, we split the analysis into two parts. We begin with analysing the term \( \psi_t - v_{q_t} \) and, by comparing \( g_{t+1}(r_t) \) from (5.72) and \( s_{t+1} \) from (5.6), we can conclude that \( s_{t+1} = \frac{1}{C_t^s} g_{t+1}(r_t) \), which implies \( q_{t+1} = \frac{1}{C_t^s} q_{t+1} \). Next, recall from Section 5.1 that \( C_t^s = \frac{1}{1 - \alpha_t} \), so the scalar \( \psi_t \) from (5.73) follows

\[
\psi_t = \frac{1}{N} q_t^T q_{t+1} = C_t^s \frac{1}{N} q_t^T q_{t+1} = (1 - \alpha_t) \frac{1}{N} q_t^T q_{t+1}.
\]
(5.76)

With this, we can arrive at

\[
\psi_t - v_{q_t} = \frac{1}{N} \left( (1 - \alpha_t) q_t^T q_{t+1} - q_t^T q_{t+1} \right).
\]
(5.77)

Next, we consider the factor \( \psi_t - v_{q_t} \) when the MP algorithms are at the fixed point and when they are still progressing. We start with the former case. Then, at the fixed point \( t^* \), assuming it exists, we have \( q_{t^*+1} = q_{t^*} \) so that (5.77) implies

\[
\psi_{t^*} - v_{q_{t^*}} = \frac{1}{N} \left( (1 - \alpha_{t^*}) q_{t^*}^T q_{t^*} - q_{t^*}^T q_{t^*} \right) = -\alpha_{t^*} \frac{1}{N} \| q_{t^*} \|^2.
\]
(5.78)
Then, we follow the assumption of Theorem 11 [58] that in the limit, the divergence of the denoiser is bounded as

$$\lim_{N \to \infty} \alpha_t \in (0, 1),$$

which implies that (5.77) is negative at the fixed point in the LSL.

Next, we consider the second case, where the algorithm is still progressing, and additionally assume that the intrinsic variance in the constructed SMP reduces monotonically. For the denoising step (5.6) it is ensured automatically when $g_t$ is MMSE, i.e. we have $v_{q_t+1} < v_q$ if $v_{h_t} < v_{h_{t-1}}$ [110]. For the linear step (5.5), there are multiple tools that ensure such a performance. For more details, check the previous chapters, the work [13] and the conclusion chapter, Chapter 6. Under this assumption, we can use the Cauchy–Schwarz inequality to bound (5.77) as

$$\tilde{\psi}_t - v_{q_t} < \frac{1}{N} \left( (1 - \alpha_t) \|q_{t+1}\| \|q_t\| - \|\tilde{q}_t\|^2 \right)$$

$$= \frac{1}{N} \|q_t\| \left( (1 - \alpha_t) \|q_{t+1}\| - \|q_t\| \right)$$

(a) $$\leq \frac{1}{N} \|q_t\| \left( \|q_{t+1}\| - \|q_t\| \right),$$

(b) $$< 0$$

(5.80)

where (a) follows from (5.79) and (b) is due to the assumption that the algorithm is not at the fixed point and monotonically reduces its intrinsic variance. Thus, the component $\tilde{\psi}_t - v_{q_t}$ in both (5.74) and (5.75) is negative under the assumption that $g_t$ is an MMSE denoiser.

Next, we consider the second component in both (5.74) and (5.75). For MF-OAMP, we can use the analysis from Section 2.4.2 where we showed that $\chi_2 > 1$ if the spectrum of $A$ is non-trivial, so that

$$1 - \chi_2 < 0,$$

which implies that (5.74) is always positive under the assumption that the denoiser $g_t$ is MMSE.

For VAMP, we can establish the following lemma.

**Lemma 4.** Consider the VAMP algorithm. Under Assumptions B1-B3 from page
the difference $v_{ht} - v_w$ is positive if the following inequality holds
\[ v_q \geq \frac{v_w}{\delta^{-1} - 1}. \]  
(5.82)

**Proof.** See Appendix L. ■

By definition, we have that $\mathcal{E}(v_{ht-1}) < v_q$ and, therefore, (5.82) is ensured as long as
\[ \mathcal{E}(v_{ht-1}) > \frac{v_w}{\delta^{-1} - 1}. \]  
(5.83)

In the following, we will assume for the VAMP case that the algorithm is in the regime where (5.83) holds, which represents a substantial part of the execution. Indeed, in practice, the subsampling factor $\delta$ is much smaller than 1 so the scaling $\delta^{-1} - 1$ in (5.83) is much greater than 1. In all the experiments in the previous sections, the MSE of VAMP was by a few orders of magnitude greater than the right hand side of (5.83). Given this assumptions holds, we have that $v_w - v_{ht}$ is negative and, together with the analysis of $\bar{\psi}_t - v_q$, we have that (5.75) is positive when $g_t$ is an MMSE denoiser.

Recall that given the sign of the derivative (5.71) at $\hat{\alpha} = \alpha_t$ and the sign of the scalar $u_3$ of the polynomial $E_{t+1}(\hat{\alpha})$ from Lemma 3, we can uniquely identify $\alpha_t$ among the two roots $\hat{\alpha}_1$ and $\hat{\alpha}_2$. Based on the above analysis on the derivative (5.71) at $\hat{\alpha} = \alpha_t$, in the case when $g_t$ is an MMSE denoiser, we can identify the right root based on the following proposal

**Proposal 2.** Consider the MF-OAMP and VAMP algorithms equipped with the MMSE denoiser $g_t(\cdot)$. Form a quadratic polynomial $E_{t+1}(\hat{\alpha}) = E_{t+1}(\hat{\alpha}, \tau)$ with $\tau = 0$ as in Lemma 3 and define $\hat{\alpha}_1$ and $\hat{\alpha}_2$ to be the two roots of $E_{t+1}(\hat{\alpha})$. Then, the divergence $\alpha_t$ of $g_t$ at $r_t$ can be estimated as
\[ \hat{\alpha}_t = \begin{cases} \max(\hat{\alpha}_1, \hat{\alpha}_2), & \text{if } u_3 < 0 \\ \min(\hat{\alpha}_1, \hat{\alpha}_2), & \text{if } u_3 > 0, \end{cases} \]  
(5.84)

where the scalar $u_3 = u_3(0)$ is as in Lemma 3.

The rule suggested by the above proposal is consistent with all the simulation experiments.
Lastly, we would like to mention the applicability of Theorem 14. While we constrained ourselves to two specific examples of SMP, i.e. MF-OAMP and VAMP, these two algorithms represent the opposite kinds of SMP algorithms in the following sense. Recall that VAMP implements the optimal function $f_t$ – the LMMSE estimator $f_t(S_{t+1}, y) = W_t^{-1}z_t$, while MF-OAMP constructs the most naive approximation of LMMSE by approximating the matrix inverse with the identity matrix, $f_t(S_{t+1}, y) = z_t$. Another example of SMP is CG-VAMP, which uses the Conjugate Gradient algorithm to approximate the LMMSE solution in an iterative fashion. At iteration $i = 0$, CG produces the same approximation as in MF-OAMP, while after $i = M$ iterations, CG is guaranteed to produce the exact LMMSE solution. For the remaining iterations $0 < i < M$, CG represents a somewhat intermediate result between the exact approximation and the naive approximation. At the moment we do not have a rigorous proof that one can extend the above analysis to CG-VAMP case, but the numerical experiments suggest that the rule from Proposal 2 is always consistent for CG-VAMP. The same tendency has been observed with respect to WS-CG-VAMP and AMP.

5.2.6 Implementation details of the polynomial estimator

As discussed in Section 5.2.2, when we implement SMP algorithms in practice, the finite dimensional model deviates from the asymptotic one, which results in the emergence of additional stochastic components in the algorithm. From our experiments, we have observed that after some iterations (close to the convergence point), the polynomial constructed from (5.60) might end up having complex roots. However, since we assume $g_t$ is a real-valued function, the divergence $\alpha_t$ must be a real value. Therefore, when the roots $\hat{\alpha}_1$ and $\hat{\alpha}_2$ of the polynomial $E_{t+1}(\hat{\alpha}, \tau)$ associated with the index $\tau$ are complex, we set them to the stationary point of the quadratic function

$$\hat{\alpha}_1(\tau) = \hat{\alpha}_2(\tau) = -\frac{u_2(\tau)}{2u_3(\tau)}.$$  \hfill (5.85)

We do the same with the roots $\hat{\alpha}_3$ and $\hat{\alpha}_4$ of the second polynomial $E_{t+1}(\hat{\alpha}, \tau')$ associated with the index $\tau'$, if these roots are complex. Next, regardless whether all the roots were originally real or not, we proceed to the rule (5.68) to form the
5.3 Simulation experiments

In this section we compare the proposed divergence estimators against the BB-MC method [1] within AMP [9], VAMP [10], CG-VAMP [106] and WS-CG-VAMP where the denoiser is chosen to be BM3D. We consider a similar inverse problem as in the previous chapters and aim to recover a natural image $x \in \mathbb{R}^N$ from Figure 2.1 (a) from a system of linear equations $y = Ax + w \in \mathbb{R}^M$ with the subsampling factor $\delta = \frac{M}{N} = 0.05$. The variance of the Gaussian measurement noise vector $w$ is set to achieve SNR $\frac{\|x\|^2}{\|w\|^2}$ of 40 dB and the measurement matrix $A$ is chosen to be FIJLT with a condition number $\kappa(A) = 1000$ as defined at the end of Section 2.3.1, except for the AMP case, where $A$ is an i.i.d. Gaussian matrix. Finally, in those experiments where we estimate the divergence of the denoiser with the BB-MC method, we choose $\epsilon$ in the same way as in Section 3.4 and use a single MC trial (additional execution of the denoiser) to form an estimate $\hat{\alpha}_t$.

In regard to the polynomial estimator, in all of the simulations we have observed that there is a pair $\tau \neq \tau'$ that satisfied (5.69). In fact, the experiments have shown that this condition was satisfied by any pair of indices and in the following simulations we will stick to the pair $(t, t-1)$, which led to a slightly better overall performance of SMP algorithms utilizing the polynomial estimator. Additionally, because the identification method proposed in Section 5.2.4 requires a pair $\tau \neq \tau'$, it cannot be implemented at $t = 0$ and, therefore, for this iteration we have used the Proposal 2 to identify the correct root.

In the following experiments, all the divergence estimators were compared against the ‘oracle’ estimator,

$$\alpha_{t,\text{oracle}} = \frac{h_t^T g_t(r_t)}{h_t^T h_t}$$

that is formed as a finite dimensional approximation of (2.104).

Lastly, the computing platform used to run the experiments is the same as mentioned in Section 3.4.
5.3.1 Polynomial vs algebraic estimators

We begin with the comparison of the polynomial estimator (5.68) against the algebraic estimators (5.37) with \( r_t = r_0 \) and \( r_t = r_{t-1} \). For this purpose, we consider the CG-VAMP algorithm recovering a natural image shown in Figure 2.1 (a) of dimension 2048 by 2048. We run a single CG-VAMP algorithm with \( i = 5 \) CG iterations and where \( \alpha_t \) is estimated by the polynomial estimator and, additionally, the two algebraic estimators are computed in parallel (these two values are not used within the algorithm and are only archived). For this experiment, we computed the normalized error

\[
\frac{(\hat{\alpha}_t - \alpha_{t, \text{oracle}})^2}{(\alpha_{t, \text{oracle}})^2},
\]

where \( \hat{\alpha}_t \) corresponds to either the estimate produced by the polynomial or by the two algebraic estimators, and the oracle divergence \( \alpha_{t, \text{oracle}} \) is from (5.86). The results averaged over 15 iterations are shown in Figure 5.1. As seen from the figure, the polynomial estimator demonstrates the best accuracy of estimating the oracle divergence (5.86), while the algebraic estimator with \( r_t = r_{t-1} \) demonstrates second to the best performance. On the other hand, the algebraic estimator with \( r_t = r_0 \) turns out to perform considerably worse than the other two and therefore is not recommended either for computing \( \alpha_t \) or for estimating the divergence of the denoiser \( g_t \) for its optimization via SURE.

Next, we assess the stability of CG-VAMP that uses different proposed divergence estimation methods. For this, we compare two CG-VAMP algorithms: one where \( \alpha_t \) is computed based on the polynomial estimator as in the previous experiment, and one where \( \alpha_t \) is estimated by the algebraic estimator with \( r_t = r_{t-1} \). Here, we computed the same error for \( \alpha_t \) and the Normalized MSE \( \frac{\| g_t(r_t) - x \|^2}{\| x \|^2} \). The two error measures averaged over 15 realizations are shown in Figure 5.2. As seen from the left plot depicting the NMSE, the CG-VAMP algorithm with the algebraic estimator with \( r_t = r_{t-1} \) diverges halfway through the execution, while the same algorithm but with the polynomial estimator demonstrates high stability. This result in combination with the previous experiment suggests that the algebraic estimator is capable of producing a relatively accurate estimate of \( \alpha_t \), if it is not used as the main correction method.
Figure 5.1: Divergence estimation error with the standard deviation error bars of the polynomial and the algebraic estimators

Figure 5.2: Left: the NMSE of CG-VAMP algorithms with the two correction methods. Right: divergence estimation error with the standard deviation error bars of the polynomial and the algebraic estimators
5.3.2 Black Box Monte Carlo and the polynomial methods for divergence estimation

Next we compare the performance of SMP algorithms when two different divergence estimation methods – BB-MC (5.2) and the proposed polynomial method (5.68), are used to estimate $\alpha_t$. First, we compare these two methods in terms of accuracy, by running two identical CG-VAMP algorithms but with two different divergence estimators. For this, we consider recovering the image ‘man’ of dimensions 1024 by 1024 shown in Figure 2.1 (a) and measured by an operator $A$ with three condition numbers $\kappa(A) = (100, 1000, 10000)$. As in the previous experiment, we used the fixed number of iterations for the CG algorithm $i = 5$. The NMSE of the algorithms averaged over 15 realizations is shown in Figure 5.3. As we see from the plot, the CG-VAMP algorithm with the polynomial divergence estimator demonstrates a similar reconstruction performance as CG-VAMP with the BB-MC estimator. Additionally, we depicted the error of estimating $\alpha_t$ oracle for the case $\kappa(A) = 1000$ in Figure 5.4. As seen from the plot, the polynomial estimator demonstrates higher accuracy of estimation for the initial iterations where CG-VAMP has a substantial per-iteration improvement and exhibits a similar accuracy when CG-VAMP is near the fixed point.
Next we keep the same inverse problem as in the last experiment with $\kappa(A) = 1000$ and compare the run time and the estimation accuracy of several SMP algorithms when the two divergence estimations methods are used. In particular, we consider the VAMP, CG-VAMP and WS-CG-VAMP algorithms. Each of these algorithms is executed separately with the BB-MC method and with the proposed polynomial method, and the results are averaged over 40 realisations. In Figure 5.5 we demonstrate the NMSE of the three pairs algorithms and in Table 5.1 we show the time required for all the algorithms to get to iteration $t = 15$. The first observation is that all the SMP algorithms demonstrate almost identical performance in terms of NMSE when we choose different methods for divergence estimation. Secondly, as seen from the table, the run time of the algorithms\(^1\) with the polynomial divergence estimator is almost twice as low as of the same algorithms but with the BB-MC divergence estimator (5.2). This confirms the initial goal of this chapter.

\(^1\)Even though here VAMP demonstrates the fastest time-wise convergence, implementing each iteration of the algorithm is only possible because we specifically designed $A$ to enable an efficient implementation of VAMP. If $A$ was a general matrix, it would be intractable to implement even a single iteration of VAMP when the size of the inverse problem is as large as in the experiment considered. Yet, we left the run-time performance for VAMP to illustrate the benefit of the proposed technique when one can design this type of measurement matrices $A$. 

---

Figure 5.4: Mean error with the standard deviation error bars of estimating the divergence $\alpha_t$ within CG-VAMP

![Graph showing mean error with standard deviation error bars for estimating divergence $\alpha_t$ within CG-VAMP](Image)
Figure 5.5: NMSE of VAMP, CG-VAMP and WS-CG-VAMP algorithms using two different divergence estimation methods.

Table 5.1: Time (in seconds) taken for SMP algorithms with two different divergence estimation methods to execute 15 iterations.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>BB-MC estimator</th>
<th>Polynomial estimator</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAMP</td>
<td>164.89</td>
<td>83.55</td>
</tr>
<tr>
<td>CG-VAMP</td>
<td>177.11</td>
<td>95.41</td>
</tr>
<tr>
<td>WS-CG-VAMP</td>
<td>178.28</td>
<td>96.7</td>
</tr>
</tbody>
</table>

5.3.3 AMP

Next, we consider the AMP case. As mentioned in Assumption A2 from page 25, the AMP dynamics is rigorously derived for those measurement systems \((5.1)\) where \(A^T A\) has the empirical eigenvalue distribution with the first \(2t\) moments equivalent to the same order moments of Marčenko-Pastur law [60]. In the following experiment, we keep the FIJL transform, generate a sequence of i.i.d. Marčenko-Pastur random values and assign the entries of the matrix \(S\) to be the square root of those random values. The rest of the parameters of the inverse problem are kept the same. The NMSE averaged over 15 realizations of the two version of AMP are shown in Figure 5.6. As seen from the plot, the two algorithms demonstrate almost identical reconstruction results. However, the AMP version
with the polynomial estimator takes 16.23 seconds on average to execute 15 iterations, while the same algorithm but with BB-MC estimator (5.2) takes 32.1 seconds for the same work.

5.3.4 CG-VAMP with Adaptive CG and the Polynomial divergence estimator

In Section 3.4.2 we conducted a numerical experiment that studied the effect of using different strategies of choosing the number of CG iterations $i[t]$ at each outerloop iteration $t$ of CG-VAMP. There, we have observed that by adding two stopping criteria (3.16) and (3.17) to CG, the resulting time taken by CG-VAMP reduced by 15% compared to not using these stopping criteria. In that experiment, we used the BB-MC divergence estimator and we mentioned that the time-wise acceleration can be increased when the computational cost of the denoising step reduces (see the discussion in Section 3.3). As we have shown above, using the polynomial divergence estimator instead of BB-MC leads to almost halving the computational time of the denoising step. Therefore, next we carry out the same experiment as Section 3.4.2 but with $\kappa(A) = 10^3$, and compare the performance of CG-VAMP with the polynomial divergence estimator and different stopping strategies for CG. The NMSE and time taken for the MP algorithm to get to iteration $t$ averaged...
over 10 realizations is shown in Figure 5.7. As seen from the plot, CG-VAMP with both stopping criteria converges to the same fixed point as CG-VAMP with no stopping criteria (apart from maximum number of CG iterations $i_{\text{max}} = 100$) roughly 30% faster. This result highlights that the two proposed methods benefit from each other.

5.4 Conclusions

In this chapter we have proposed two alternatives to the traditional Black-Box Monte Carlo [1] methods for estimating the divergence of denoisers within SMP algorithms. Similarly to BB-MC, the proposed methods do not use any additional information about the denoiser apart from its input and output. However, contrary to the BB-MC method, the two suggested estimators do not require executing the denoiser additional times and, therefore, significantly accelerate the SMP algorithm when an expensive denoiser such as BM3D is used. The first method - the algebraic estimator – has a negligible computational cost and can produce a rough estimate of the divergence of a denoiser, which can potentially be further used to, for example, optimize the performance of the denoising part of SMP. The second estimation method – the polynomial estimator – complements the first one and demonstrates high robustness with respect to the dimensional-
ity of the inverse problem and a similar accuracy of estimation compared to the BB-MC method. The polynomial estimator generates two estimates (two roots) and to identify which of the two estimates is associated with the divergence $\alpha_t$, we developed a rigorous test that can be implemented online. Additionally, we developed the condition that determines the correct root theoretically in the case of VAMP and MF-OAMP algorithms when the MMSE denoiser is used and for a certain regime of operation of the VAMP algorithm.
Chapter 6

Conclusions and further work

In this thesis we considered the problem of solving a large-scale Compressed Sensing inverse problem using the Message Passing framework. We produced a series of theoretical results that provided an insight on the asymptotic behaviour of MP algorithms and that were used to upscale, stabilize and accelerate those algorithms. To summarize, the following questions and problems have been addressed.

The emergence of Approximate Message Passing [9, 52] has inspired much research to develop first-order MP algorithms that can provably solve the CS inverse problems. AMP has been shown to possess rigorous error dynamics, be provably convergent and Bayes-optimal [54, 64, 65, 66], and able to recover complex large-scale signals like natural images by utilizing Plug-and-Play denoisers [69]. Unfortunately, these properties were only shown to hold for, roughly speaking, random i.i.d. measurement operators [60, 115], which is quite restrictive in practice. Shortly after, Vector AMP [10, 79, 85] was introduced, which preserved most of AMP's advantages while being able to operate with a larger class of random matrices called right-orthogonally invariant (ROI) from Assumption B3 that also includes ill-conditioned operators. Recently, the allowed class of matrices for VAMP was extended to semi-random matrices called Fast Johnson Lindenstrauss Transforms [11, 12], which have a fast matrix-vector multiplication implementation. However, VAMP’s practical applicability was limited to relatively small problem sizes due to the need to compute the the LMMSE estimator that involves matrix inversion of an $M$ by $M$ matrix. Conjugate Gradient VAMP [92] represents
an upscaled VAMP algorithm that uses Conjugate Gradient with a few iterations to approximate LMMSE. This allowed a reduction of per-iteration computational cost of the MP algorithm down to AMP’s while being more flexible with respect to the measurement operator, but the originally proposed variant of CG-VAMP suffers from the stability issues and its fixed point tends to be much worse than VAMP’s when the measurement operator is even mildly ill-conditioned.

In chapter 3 we developed a series of rigorous tools for making CG-VAMP stable and able to approximate the performance of VAMP with almost arbitrary accuracy under the assumption that \( \mathbf{A} \) is ROI. Additionally, we proposed a combination of stopping criteria for CG that are based on comparing the intrinsic variance in CG-VAMP across both the inner-loop and outer-loop iterations. These criteria were used to construct the Adaptive CG that chooses such a number of iterations that leads to faster time-wise convergence of CG-VAMP without sacrificing the accuracy of estimation. However, all the above ideas for pushing the performance of CG-VAMP closer to VAMP’s were based on increasing the number of CG iterations, which naturally leads to the increased computational cost of the algorithm.

To tackle this drawback, in Chapter 4 we developed a warm-started framework for approximating LMMSE that is based on reusing the information from the previous iterations of MP to improve the quality of approximation while keeping a low computational cost. This framework includes the WS CG and the WS Gradient Descent methods, besides other possible examples. We show that when an instance of the proposed WS framework is used within MP to approximate the LMMSE, a fixed point of the algorithm is a fixed point of VAMP. Importantly, this result is independent of the number of inner-loop iterations of this approximation framework. Additionally, we showed that the approach for approximating LMMSE in the recently proposed MAMP algorithm [14] is an instance of our framework. We generalized previously developed tools for CG to the context of the WS framework to make the resulting MP algorithms stable in practice and numerically demonstrated the advantages of using WS in MP.

Chapter 5 was devoted to the sub-problem that arises in any MP considered in this thesis – estimating the divergence of the denoiser required to form the Onsager correction ensuring the main asymptotic properties of the algorithm. In the literature on MP algorithms, the only method that has been suggested for
estimating the divergence of a PnP denoiser such as BM3D, NLM, DnCNN is the black-box Monte Carlo method [1] that requires executing the denoiser at least one additional time at every iteration. In this chapter we developed the asymptotic theory behind the divergence of a denoiser in MP and proposed two models of this quantity. Next, we used these models to form two estimators – the algebraic and the polynomial divergence estimators that can be used within any MP and do not require additional executions of the denoiser. The former example has a slightly smaller computational cost compared to the latter one, but demonstrates lower robustness with respect to the decreased dimensionality. We numerically compared the performance of the proposed methods, and demonstrated that an MP algorithm equipped with the polynomial estimator remained stable following the predicted SE performance while converging almost twice as fast (in time) compared to the same algorithm but with the BB-MC divergence estimator.

6.1 Further work

Next, we would like to mention several open questions and problems related to the results in this thesis and provide some intuition on how one could potentially approach them. We begin with the questions related to the warm-started methodology and to long-memory MP algorithms in general.

6.1.1 Warm-starting and LM MP algorithms

1. Convergence of MP algorithms with WS methods

While Theorem 7 implies the potential optimality of the fixed points of MP algorithms utilizing the WS framework (4.37), (4.40) to approximate the LMMSE, so far we have not obtained rigorous results proving the convergence of those algorithms. In fact, in Section 4.3.3 we had to use the damping strategy (4.56) with \( l = 2 \) in order to stabilize the WS-GD-VAMP algorithm with \( i = 1 \) inner-loop iteration.

The convergence aspect of MP algorithms has been recently solved in the works [13, 89] in a general case, where it was proved that one can be flexible with the choice of the denoiser \( g_B \) and the function \( g_A \) approximating
LMMSE if the updates of both $\mathbf{x}^t_{A\rightarrow B}$ and $\mathbf{x}^t_{B\rightarrow A}$ are followed by the sufficient statistics step. In the same works its was shown that the damping step (4.56) used for MAMP and WS-GD-VAMP in Sections 4.3.3 and 4.3.4 represents the truncated $l < t$ version of this sufficient statistics step. If one uses (4.56) with $l = t$ to update $\mathbf{x}^t_{B\rightarrow A}$ and constructs a similar update for $\mathbf{x}^t_{A\rightarrow B}$, then the sequences of $\mathbf{x}^t_{A\rightarrow B}$ and $\mathbf{x}^t_{B\rightarrow A}$ are provably convergent under Assumptions B1-B3. Yet, it might be of independent interest to work out why WS-CG-VAMP with $i = 1$ is stable, while WS-GD-VAMP with $i = 1$ is not. Answering this question will automatically determine why MAMP requires damping as well.

2. Stability of LM MP algorithms

Another problem that we have observed with implementing different LM MP algorithms is the sensitivity of those algorithms to reducing the dimensionality $N$ and $M$ of the inverse problem. While the theory derived in Chapter 4 guarantees certain behaviour of the MP algorithms utilizing WS methods, this theory is derived under Assumption B1 stating that the dimension $M = \delta N$ with $\delta = O(1)$ approaches infinity. This assumption guarantees that certain stochastic error generated in the algorithm has a negligible impact on the dynamics, but that is not the case when we operate in the finite dimensional setting. In the latter case, each iteration of MP generates additional error that is not tracked by the SE and this error increases as the algorithm progresses. It has been observed by multiple authors [13, 14] and in Chapter 4 in this thesis that the LM MP algorithms are sensitive to this kind of error and rarely remain stable for more than a hundred outer-loop iterations $t$ when no oracle information about the measurement system $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{w}$ is provided. We hypothesise that a major part of this stability problem in LM MP is related to the process of estimating $t$ Onsager correction terms (given we estimate the variance $\nu^t_{A\rightarrow B}$ as proposed in Section 4.1.2).

To the best of our knowledge, all the available techniques for estimating the set of $t$ Onsager correction terms $\{\gamma^{t,\tau,i}_{A\rightarrow B}\mathbf{x}^\tau_{B\rightarrow A}\}_{\tau=0}^t$ can be split into three groups. The first group, the most popular one, has been used in the works
of different authors including [14, 55, 63, 93, 100], and is based on the idea of using the original definition of the scalars $\gamma^{t,\tau,i}_A$ being the divergence of $g_A$ with respect to $x_{B \to A}^\tau$. In this approach, one derives the asymptotic model of $g_A$, which is usually a polynomial of $(AA^T)$ up to some order $i^*$, and uses this model to form the closed form solution for $\gamma^{t,\tau,i}_A$, which in turn is a polynomial of the spectral moments $\chi_j$ and statistical parameters of MP, like $v_{B \to A}^\tau$. The drawback of this approach is that the resulting estimator formed this way uses higher order spectral moments $\chi_j$ that are hard to estimate and that magnify other stochastic error in the algorithm.

An alternative method for estimating $\gamma^{t,\tau,i}_A$ has been proposed in this thesis and it is built on a different identity for this scalar, namely (4.53). In the context of WS-CG-VAMP, this LSL identity has led to an iterative estimation method based on Theorem 6 that does not explicitly involve the spectral moments $\chi_j$, but requires manipulating an inverse of the cross-correlation matrix $\Psi$ with $\Psi_{\tau,\tau'} = \frac{1}{N} q_{\tau}^T q_{\tau'}$. As has been pointed out in Section 4.1.2, this matrix becomes more ill-conditioned as an MP algorithm progresses and estimating its inverse gets more sensitive to the stochastic error generated in the algorithm.

We see three possible research directions for making LM MP algorithms more stable in practice to push their performance closer to the theoretical one. The first one would be to extend the analysis from the works [116, 117] where it was considered the evolution of AMP in the finite dimensional case and studied how the stochastic error ignored in the LSL analysis evolves with $t$. By better understanding this stochastic error, one potentially could design estimators that are more robust in practice.

The second approach is to refer to the third group of estimators of the divergence of a function – Monte Carlo methods [1]. This idea was used in [94] to estimate $\gamma^t_A$ when $g_A$ is zero-initialized CG. The important advantage of this method is that it operates as a black box and is not dependent on any parameters of the chosen MP algorithm. As a result, it is not affected by the accumulated stochastic error in the algorithm and does not amplify this error further. The main challenge with this approach is that if we wanted
Conclusions and further work

to estimate the divergence of a function \( g_A(x_{B\to A}^0, x_{B\to A}^1, \ldots, x_{B\to A}^t) \) with respect to some \( x_{\tau\to A}^\tau \), then we would need to re-execute this function with a perturbed input \( x_{\tau\to A}^\tau + \mathbf{e} \) for a certain vector \( \mathbf{e} \). But since in LM MP the function \( g_A(x_{B\to A}^0, x_{B\to A}^1, \ldots, x_{B\to A}^t) \) is convoluted, it might be necessary to re-execute the whole chain of \( g_A \) for each of its \( t \) inputs. Still, one might wonder whether it is possible to avoid such an expensive operation by storing some intermediate results.

Lastly, the third path would be approximating the complete Onsager correction step in LM MP as was proposed in Section 4.1.3. While in this thesis we considered the most naive realization of this idea, we believe that it can be extended further. For example, one could consider the error vector matrix \( \mathbf{Q}_{t+1} = (q_0, \ldots, q_t) \) and work out its \( k \leq t \) singular vectors that carry most of the power and correct with respect to those \( k \) vectors, not with respect to each \( q_\tau \) for \( \tau \leq t \). Although these error vectors are not directly available, one can still relatively reliably estimate their inner-products \( \frac{1}{N} q_\tau^T q_{\tau'} \) (see Section 4.1.1), which would be the primal tool for extracting these leading singular vectors.

3. Towards the optimal WS approximation of LMMSE

Another possible line of work concerns the optimization of methods used for approximating the LMMSE within MP algorithms. In this thesis we have used CG, WS-CG, WS-GD and the general WS unified framework (4.37), (4.40) as the approximation methods, and all of these examples have the associated internal parameters, \( \{a_i^t\}_{i=0}^t \) and \( \{b_i^t\}_{i=0}^t \) (see Section 4.2.2), that affect the quality of approximation. For example, in CG, these scalars are optimized in a way so that every new search direction is conjugate to all the previous ones (see Section 2.4) and the step towards this direction minimizes the quadratic cost (4.36). As a result, CG represents the optimal first-order method for approximating a system of linear equation by minimizing a specific quadratic cost [105]. However, in the MP algorithms we are interested in minimizing the SE, which is different to the quadratic cost function (4.36).

Based on the numerical experiments in this thesis, WS-CG method demonstrated great approximation potential. Therefore, we suggest keeping the
key ingredient of CG – the conjugate directions – while adapting the method to minimize the SE instead of the specific quadratic cost function. As was shown in [95], the conjugacy property of CG is ensured by the scalars \( \{b^i_t\}_{j=0} \), while the set of scalar \( \{a^i_t\}_{j=0} \) is chosen to minimize the quadratic cost. Thus, we recommend using the unified WS framework (4.37), (4.40) and changing the computation of \( b^i_t \) to [95]

\[
 b^i_t = \frac{1}{a^i_t} \frac{(r^{i+1}_t)^T r^{i+1}_t}{(p^i_t)^T W_t p^i_t}
\]  

(6.1)

where \( r^i_t = z_t - W_t p^i_t \). The difference between this form of \( b^i_t \) and the one from CG from Algorithm 3 is that (6.1) preserves the conjugacy of the vectors \( p^i_t \) in (4.37) for any \( a^i_t \) [95]. The last step would be optimizing \( a^i_t \) to minimize the SE and this is left for further work. A similar idea was carried out in MAMP [14], CAMP [93] and the rotationally invariant AMP [63], where the authors optimized their approximation step with respect to the SE. These works might be a good starting point for developing the optimal WS approximation of LMMSE within LM MP.

6.1.2 Divergence estimation in MP

Next we outline several open questions related to the divergence estimation in MP

1. Robustness of divergence estimators

The two divergence estimators proposed in Chapter 5 were derived under the large system limit assumption. However, when they are applied to finite dimensional problems, their estimation accuracy substantially differ and might drop in certain scenarios. These facts lead to the following questions:

(a) Why is the polynomial estimator more robust with respect to the decreased dimensionality? Is it possible to modify the fast algebraic estimator accordingly to increase its robustness?

(b) From the thorough numerical study (not demonstrated in this thesis), we have found that the polynomial estimator can sometimes demonstrate worse performance than BB-MC for certain denoisers \( g_t \) and when
the MP algorithm is close to the fixed point. For example, the polynomial estimator performs better when $g$ is BM3D or NLM denoiser, and is worse when it is the DnCNN denoiser. Then, what are the properties of a denoiser that benefit the polynomial estimator and how to modify the latter to improve the estimation accuracy?

The two questions concern the finite dimensional approximation of the LSL models and one way of studying the robustness of the proposed estimation methods would be by applying the non-asymptotic analysis from the works [116, 117] developed for AMP. However, this approach might be very challenging, since the proposed divergence estimators are designed for every MP algorithm fitting the model (5.5)-(5.6). Thus, first, the ideas from [116, 117] would have to be extended to the unified MP framework from [60] and considered for each popular choice of the linear function $f$, and then these results could be used to obtain a better understanding of the proposed divergence estimators in the finite dimensional setting.

2. *Universality of the root identification in the polynomial divergence estimator*

   In Section 5.2.5 we considered two examples of MP – MF-OAMP and VAMP – and showed that one can theoretically identify the right root in the polynomial estimator without the need for the second polynomial. There, these results were rigorously shown for the MMSE denoiser and an additional assumption for VAMP that it operates in a certain regime. However, based on thorough numerical study (not demonstrated in this thesis), it has been observed that the proposed identification approach holds for many non-MMSE PnP denoisers such as BM3D, NLM etc, and for any regime of these algorithms. Moreover, these results have been witnessed to hold for AMP, CG-VAMP, WS-CG-VAMP and MAMP as well. Therefore, an additional research is required to prove or disprove the fact that the root identification rule from Proposal 2 in Section 5.2.5 is universal.
6.1.3 Extending the range of inverse problems

In this thesis we focused on solving the inverse problems where the measurement operator $A$ possesses sufficient randomness required to prove most of the LSL properties of MP. At the same time, in many real-world problems, the model of $A$ is dictated by the physics of the application and there is no freedom for designing the measurement operator. Importantly, in many of those cases the matrix $A$ is highly structured, such as in the MRI application where $A$ can be modeled as a subsampled Fourier operator. Therefore, it is of a great interest to develop new mathematical tools for adapting the ideas from this thesis to more general settings. In this regard, recently, several works proposed heuristics for adapting AMP [118, 119, 120], VAMP [109, 121] and the Generalized Expectation Consistent approximation [122] (an algorithm similar in nature to MP) to the case, where $y = Pf + w$ with $P$ being a subsampling matrix and $F$ being the Fourier operator.

The empirical results demonstrated in these works suggest that there is a room for extending the range of applications suitable for MP algorithms and it would be an interesting challenge to prove those empirical findings rigorously.

Lastly, so far we have only considered the Compressed Sensing scenario where the signal $x$ is observed through a so-called additive white Gaussian noise output channel [123] where $j$-th measurement is obtained as $y_j = a_j^T x + w_j$ with $w_j$ being white Gaussian noise. However, there are applications where the transformed signal $a_j^T x$ is corrupted in some other ways. For example:

1. Binary classification [63, 124, 125, 126], where $y_j = sgn(a_j^T x + w_j)$ and $sgn(v)$ outputs 1 if $v \geq 0$ and $-1$ if $v < 0$.

2. Quantized compressed sensing [127, 128]. Here the measurement $y_j = q(a_j^T x + w_j)$ is affected by the quantizer $q(\cdot)$.

3. Phase retrieval [61, 114, 129, 130] where $y_j = |a_j^T x + w_j|$ i.e. we obtain only the magnitude of the linear noisy measurement $a_j^T x + w_j$.

The above measurement systems and many other examples [131] are sometimes referred to as Generalized Linear Models (GLM). It has been shown that AMP [123, 129], VAMP [131, 132, 133, 134], rotationally invariant AMP [62, 63, 100] and
Conclusions and further work

MAMP [135] can be extended to handle GLM and preserve the main theoretical properties – existence of the SE in the large system limit. We believe that WS-CG-VAMP, WS-GD-VAMP and other LM MP algorithms utilizing the unified WS approximation framework (4.37), (4.40) can be adapted to handle GLM as well. Similarly, we think it is possible to extend the proposed divergence estimators from Chapter 5 to operate in MP algorithms recovering a signal measured by GLM. Rigorously adapting the ideas presented in this thesis to GLM is left for further work.
Appendix A

Supplementary material

In this appendix we prove or mention some technical lemmas that are used throughout the thesis. We begin with defining the Strong Law of Large Numbers (SLLN).

Lemma 5. [136, Theorem 2.3.10] Let \( X_N = (x_1, ..., x_N) \) be a sequence of independent random variables whose expectation \( E[x_n] \) and variance \( Var[x_n] \) exist for every \( n \in (1, ..., N) \). Define \( \bar{X}_N = \sum_{n=1}^{N} x_n \). Then

\[
\bar{X}_N - E[\bar{X}_N] \xrightarrow{(a.s.)} 0 \quad (A.1)
\]

if

\[
\sum_{n=1}^{N} \frac{Var[x_n]}{N^2} < \infty. \quad (A.2)
\]

Next, we define the multidimensional version of the Stein’s Lemma from [59].

Lemma 6. [93, Lemma 2]: Let \( z = (z_1, ..., z_t)^T \sim N(0, \Sigma) \) for a positive-definite matrix \( \Sigma \). Let \( g \) be a Lipschitz-continuous function, \( f: \mathbb{R}^T \rightarrow \mathbb{R} \). Then,

\[
E[z_1 g(z)] = \sum_{\tau=1}^{t} E[z_1 z_{\tau}] E\left[\frac{\partial g(z)}{\partial z_{\tau}}\right]. \quad (A.3)
\]

Next, we present some standard random matrix results for the Gaussian Orthogonal Ensemble from the following definition.

Definition 1. Let \( B \in \mathbb{R}^{N \times N} \) be an i.i.d. zero-mean Gaussian matrix whose diagonal and off-diagonal entries have the variance \( 2 \frac{1}{N} \) and \( \frac{1}{N} \) respectively. Then \( B \) is from the Gaussian Orthogonal Ensemble (GOE).
A GOE matrix can be formed in multiple ways including the one dictated by the following lemma.

**Lemma 7.** Let $A \in \mathbb{R}^{M \times N}$ be a matrix with zero-mean i.i.d. Gaussian entries with variance $\frac{1}{M}$. Then a matrix $B = M N (I - A^T A) \in \mathbb{R}^{N \times N}$ is a GOE matrix as in Definition 1.

**Proof.** First, consider the entry $(i, j)$ of the product $A^T A$

$$
(A^T A)_{i,j} = \sum_{k=1}^{M} A_{i,k} A_{j,k}.
$$

(A.4)

Next, we would like to confirm that the above sum converges to a normal random variable. For this, we need to verify that the first and the second moments of $A_{i,k} A_{j,k}$ are finite. We have two cases: $i = j$, for which $E[A_{i,k}^2] = \frac{1}{M} \leq \infty$ and $E[A_{i,k}^4] = 3 \left( E[A_{i,k}^2] \right)^2 = 3 \frac{1}{M^2} \leq \infty$, where we used the standard relationship of the fourth and the second moments of a zero-mean Gaussian variable. For the case $i \neq j$, we have that $A_{i,k}$ and $A_{j,k}$ are independent so $E[A_{i,k} A_{j,k}] = E[A_{i,k}]E[A_{j,k}] = 0$ and $E[(A_{i,k} A_{j,k})^2] = E[A_{i,k}^2]E[A_{j,k}^2] = \frac{1}{M^2}$. Thus, we have that

$$
\lim_{N \to \infty} (A^T A)_{i,j} \xrightarrow{(d)} N(\mu, \sigma^2).
$$

(A.5)

The next goal is to identify the mean $\mu$ and the variance $\sigma^2$ of this Gaussian variable. Again, we need to consider two cases and we begin with $i = j$ so that

$$
E[(A^T A)_{i,i}] = E[\sum_{k=1}^{M} A_{i,k}^2] = \sum_{k=1}^{M} \frac{1}{M} = 1,
$$

(A.6)

while for $i \neq j$, we have

$$
E[(A^T A)_{i,j}] = E[\sum_{k=1}^{M} A_{i,k} A_{j,k}] = \sum_{k=1}^{M} E[A_{i,k}]E[A_{j,k}] = 0.
$$

(A.7)

We follow the same steps for the variance of $(A^T A)_{i,j}$ and begin with $i = j$, for
Supplementary material

which

\[
E[(A^T A)_{i,i}^2] = E\left[\left( \sum_{k=1}^{M} A_{i,k}^2 \right)^2 \right] = E\left[ \sum_{k=1}^{M} \sum_{u=1}^{M} A_{i,k}^2 A_{i,u}^2 \right] \]

\[
= \sum_{k=1}^{M} \sum_{u=1}^{M} E[A_{i,k}]^2 E[A_{i,u}]^2 + \sum_{k=1}^{M} E[A_{i,k}^4] = \sum_{k=1}^{M} \sum_{u=1}^{M} \frac{1}{M^2} + \sum_{k=1}^{M} 3 \frac{1}{M^2} 
\]

\[
= 1 - \frac{1}{M} + 3 \frac{1}{M} = 1 + 2 \frac{1}{M},
\]

(A.8)

where (a) holds since \(A_{i,k}\) and \(A_{i,u}\) are independent for \(u \neq k\). Similarly for the case \(i \neq j\) we have

\[
E[(A^T A)_{i,j}^2] = E\left[\left( \sum_{k=1}^{M} A_{i,k} A_{j,k} \right)^2 \right] = E\left[ \sum_{k=1}^{M} \sum_{u=1}^{M} A_{i,k} A_{j,k} A_{i,u} A_{j,u} \right] 
\]

\[
= \sum_{k=1}^{M} \sum_{u=1}^{M} E[A_{i,k}] E[A_{j,k}] E[A_{i,u}] E[A_{j,u}] + \sum_{k=1}^{M} E[A_{i,k}^2 A_{j,k}^2] 
\]

\[
= 0 + \sum_{k=1}^{M} \frac{1}{M^2} = \frac{1}{M^2},
\]

(A.9)

Lastly, one can follow exactly the same steps to verify that the correlation between \((A^T A)_{i,j}\) and \((A^T A)_{i',j'}\) for \(i \neq i'\) and \(j \neq j'\) is zero.

Finally, combining the results above for \(A^T A\), we conclude that \(B = I - A^T A\) is a zero-mean matrix with independent entries and whose off-diagonal entries have variance \(\frac{1}{N}\) and the diagonal entries have variance \(2\frac{1}{N}\), which implies \(B\) is a GOE matrix.

Next, we prove a lemma that defines the effect of applying a GOE matrix to a fixed vector.

**Lemma 8.** Let \(A \in \mathbb{R}^{N \times N}\) be a GOE matrix and \(x \in \mathbb{R}^{N}\) be a fixed vector independent of \(A\) such that \(\frac{1}{N}||x||^2 = 1\). Then, in the limit, \(y = Ax\) is a vector of standard normal variables, \(y \overset{(d)}{\rightarrow} \mathcal{N}(0, I)\)

**Proof.** First, we can use the Lyapunov Central Limit Theorem to show that in the
limit, each entry $y_i$, $i \in (1,...,N)$ is a Gaussian variable

$$y_i = (Ax)_i = \sum_{k=1}^{N} A_{i,k} x_k \overset{(d)}{\to} N(\mu, \Sigma).$$  \hfill (A.10)

Then, we need to identify the mean $\mu$ and the covariance $\Sigma$ of this normal density. We start with the former one:

$$\mu = \mathbb{E}[Ax] = \mathbb{E}[A]x = 0.$$  \hfill (A.11)

Next, we consider the entry $(i, j)$ of the covariance $\Sigma$. When $i = j$, we have

$$\mathbb{E}[(Ax)^2_i] = \mathbb{E} \left[ \sum_{k=1}^{N} \sum_{u=1}^{N} A_{i,k} A_{i,u} x_k x_u \right] = \sum_{k=1}^{N} \mathbb{E}[A_{i,k}^2] x_k^2$$

$$= \sum_{k=1, k \neq i}^{N} \mathbb{E}[A_{i,k}^2] x_k^2 + \mathbb{E}[A_{i,i}^2] x_i^2 = \frac{1}{N} \sum_{k=1, k \neq i}^{N} x_k^2 + \frac{1}{N} x_i^2$$

$$= \frac{1}{N} \|x\|^2 + \frac{1}{N} x_i^2 = 1 + \frac{1}{N} x_i^2,$$  \hfill (A.12)

while for $i \neq j$ we have

$$\mathbb{E}[(Ax)_i(Ax)_j] = \mathbb{E} \left[ \sum_{k=1}^{N} \sum_{u=1}^{N} A_{i,k} A_{j,u} x_k x_u \right] = \sum_{k=1}^{N} \sum_{u=1}^{N} \mathbb{E}[A_{i,k} A_{j,u}] x_k x_u.$$  \hfill (A.13)

To compute the above sum, note that $A$ is symmetrical, so that $A_{i,j} = A_{j,i}$. Next, note that because $A_{i,k}$ and $A_{j,u}$ are independent when $(i, k) \neq (j, u)$ or $(i, k) \neq (u, j)$, we have that all components $\mathbb{E}[A_{i,k} A_{j,u}]$ are zero except for the case when $k = j$ and $u = i$, so that $\mathbb{E}[A_{i,k} A_{j,u}] = \mathbb{E}[A_{i,j} A_{j,i}] = \mathbb{E}[A_{i,k}^2] = \frac{1}{N}$. Therefore we have that

$$\mathbb{E}[(Ax)_i(Ax)_j] = \frac{1}{N} x_j x_i,$$  \hfill (A.14)

which completes the proof.
Appendix B

Proofs of Lemma 1 and Theorem 2

Before proving Theorem 2, first, we prove Lemma 1 that serves as the basis for the proposed iterative LSL model of the divergence $\gamma^{t,i}_A$ in CG-VAMP.

B.1 Proof of Lemma 1

Next, we consider Lemma 1 originally formulated for the case where $g^i_A$ is the CG algorithm with $i$ iterations approximating the SLE $W_t \mu_A = z_t$ where

\[ z_t = y - Ax^i_{B \rightarrow A} \]  \hspace{1cm} (B.1)
\[ W_t = \hat{v}_w I + \hat{v}^t_{B \rightarrow A} AA^T \] \hspace{1cm} (B.2)

Here, however, we prove the lemma for a general case, where $g_A(x^t_{B \rightarrow A}, \hat{v}^t_{B \rightarrow A})$ has a well-defined Jacobian and depends on $x^t_{B \rightarrow A}$ through $z_t$ so that

\[ g_A(x^t_{B \rightarrow A}, \hat{v}^t_{B \rightarrow A}) = g_A(z_t, \hat{v}^t_{B \rightarrow A}), \]  \hspace{1cm} (B.3)

which is confirmed for CG in (2.139), but also holds for VAMP, where

\[ g_A(x^t_{B \rightarrow A}, \hat{v}^t_{B \rightarrow A}) = W_t^{-1} z_t \]

and for MF-OAMP [79], where

\[ g_A(x^t_{B \rightarrow A}, \hat{v}^t_{B \rightarrow A}) = z_t, \]
besides other possible MP algorithms. Note that by using the SVD of $A = USV^T$, the vector $z_t$ can be equivalently written as

$$z_t = y - Ax_{B 	o A}^t = w - Aq_t = w - USV^Tq_t$$

$$= w - USb_t. \quad (B.4)$$

Recall from Theorem 11 that under Assumptions B1-B3 from page 42, the vector $b_t = V^Tq_t$ asymptotically acts as a zero-mean i.i.d. Gaussian vector with the variance $v_{B 	o A}^t$. This implies that in the limit, $z_t$ is zero-mean i.i.d. Gaussian. With this in mind, consider the inner-product $\frac{1}{N}q_t^TAg_A(x_{B 	o A}^t, \hat{v}_{B 	o A}^t)$ in the regime $N \to \infty$,

$$\lim_{N \to \infty} \frac{1}{N}q_t^TA^Tg_A(x_{B \to A}^t, \hat{v}_{B \to A}^t) = \lim_{N \to \infty} \frac{1}{N}b_tS^TU^Tg_A(z_t, \hat{v}_{B \to A}^t)$$

$$\overset{\text{a.s.}}{= \lim_{N \to \infty}} \frac{1}{N} \text{Tr}\left\{b_t(g_A(z_t, \hat{v}_{B \to A}^t))^TUS\right\} \quad (B.5)$$

$$\overset{\text{a.s.}}{=} \lim_{N \to \infty} \frac{1}{N} \text{Tr}\left\{\text{USE}[b_tz_t^T]J_{z_t}(g_A(z_t, \hat{v}_{B \to A}^t))\right\}$$

$$\overset{\text{a.s.}}{=} \lim_{N \to \infty} \frac{1}{N} \text{Tr}\left\{\text{USE}[b_t(w - USb_t)]^TJ_{z_t}(g_A(z_t, \hat{v}_{B \to A}^t))\right\} \quad (B.6)$$

$$\overset{\text{a.s.}}{=} \lim_{N \to \infty} -v_{B \to A}^t \frac{1}{N} \text{Tr}\left\{\text{USS}^TU^TJ_{z_t}(g_A(z_t, \hat{v}_{B \to A}^t))\right\} \quad (B.7)$$

$$\overset{\text{a.s.}}{=} \lim_{N \to \infty} -v_{B \to A}^t \frac{1}{N} \text{Tr}\left\{AA^TJ_{z_t}(g_A(z_t, \hat{v}_{B \to A}^t))\right\} \quad (B.8)$$

$$\overset{\text{a.s.}}{=} \lim_{N \to \infty} v_{B \to A}^t \frac{1}{N} \text{Tr}\left\{A^TJ_{X_{B \to A}}(g_A(z_t, \hat{v}_{B \to A}^t))\right\} \quad (B.9)$$

$$= v_{B \to A}^t \gamma_{A^i} \quad (B.10)$$

where in (B.5) we used Lemma 6, (B.6) follows from the fact that $USb_t$ and $w$ are asymptotically independent as shown in Theorem 11,

$$\lim_{N \to \infty} \frac{1}{N}w^TAq_t = \lim_{N \to \infty} \frac{1}{N}w^TUSb_t \overset{\text{a.s.}}{=} 0. \quad (B.11)$$
Proofs of Lemma 1 and Theorem 2

(B.7) is due to $E[\mathbf{b}_t \mathbf{b}_t^T] = \frac{1}{N} \mathbf{b}_t^T \mathbf{b}_t \mathbf{I}$ shown in [60], in (B.8) we used the observation that $-\mathbf{J}_t(\mathbf{g}_A(\mathbf{z}_t, \hat{\mathbf{v}}_{B \rightarrow A})) \mathbf{A} = \mathbf{J}_t^{x_{B \rightarrow A}}(\mathbf{g}_A(\mathbf{x}_{B \rightarrow A}^t, \hat{\mathbf{v}}_{B \rightarrow A}^t))$, since $\mathbf{z}_t = \mathbf{y} - \mathbf{A}\mathbf{x}_{B \rightarrow A}^t$, and (B.9) is due to the fact that the Jacobian is a linear operator. Finally, dividing both sides of (B.10) by $\mathbf{v}_{B \rightarrow A}^t$ proves the desired statement

$$\lim_{N \to \infty} \gamma_{t,i}^{t,i} \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} \mathbf{q}_t^T \mathbf{A}^T \mathbf{g}_A^{t,i}(\mathbf{x}_{B \rightarrow A}^t, \hat{\mathbf{v}}_{B \rightarrow A}^t).$$

(B.12)

\[\square\]

### B.2 Proof of Theorem 2

In the following, all the statements are made under Assumptions B1-B3 from page 42. Recall that based on Lemma 1, the divergence $\gamma_{t,i}^{t,i}$ of $\mathbf{g}_A(\mathbf{x}_{B \rightarrow A}^t, \hat{\mathbf{v}}_{B \rightarrow A}^t)$ with respect to $\mathbf{x}_{B \rightarrow A}$ can be equivalently represented as

$$\lim_{N \to \infty} \gamma_{t,i}^{t,i} \overset{a.s.}{=} \lim_{N \to \infty} \frac{\nu_{t}^i - \frac{1}{N} \mathbf{z}_t^T \mathbf{\mu}_{t,i}^{A}}{\mathbf{v}_{B \rightarrow A}^t},$$

(B.13)

provided the asymptotic result of the inner-product $\nu_{t}^i = \frac{1}{N} \mathbf{w}^T \mathbf{p}_{t}^i$. Note that based on the definition of $\mathbf{\mu}_{t,i}^{A}$ from Algorithm 3, we have for $i > 0$ that

$$\lim_{N \to \infty} \nu_{t}^i = \lim_{N \to \infty} \frac{1}{N} \mathbf{w}^T (\mathbf{\mu}_{t,i}^{i-1} + \alpha_{t}^{i-1} \mathbf{p}_{t}^{i-1})$$

$$= \lim_{N \to \infty} \nu_{t}^{i-1} + \alpha_{t}^{i-1} \eta_{t}^{i-1},$$

(B.14)

where we defined $\eta_{t}^i = \frac{1}{N} \mathbf{w}^T \mathbf{p}_{t}^i$, which can be further expanded using the definition of $\mathbf{p}_{t}^i$ from Algorithm 3 as

$$\lim_{N \to \infty} \eta_{t}^i = \lim_{N \to \infty} \frac{1}{N} \mathbf{w}^T \mathbf{p}_{t}^i$$

$$= \lim_{N \to \infty} \frac{1}{N} \mathbf{w}^T (\mathbf{z}_t - \mathbf{W}_t \mathbf{\mu}_{t,i}^{A} + \beta_{t}^{i-1} \mathbf{p}_{t}^{i-1})$$

$$\overset{a.s.}{=} \delta v_w + \lim_{N \to \infty} b_{t}^{i-1} \eta_{t}^{i-1} - \frac{1}{N} \mathbf{w}^T \mathbf{W}_t \mathbf{\mu}_{t,i}^{A},$$

(B.15)

where we used $\lim_{N \to \infty} \frac{1}{N} \mathbf{w}^T \mathbf{z}_t \overset{a.s.}{=} \delta v_w$. This results follows from (B.4), the asymptotic independence (B.11) and Lemma 5 applied to a Gaussian vector $\mathbf{w}$ to show
that \( \lim_{N \to \infty} \frac{1}{M} W^T w \overset{a.s.}{=} v_w \). Next, we expand the matrix \( W_t \) to arrive at
\[
\lim_{N \to \infty} \frac{1}{N} W^T W_t \mu_A^{t,i} = v_w^i v_t^i + v_{B \to A} \lim_{N \to \infty} \frac{1}{N} W^T \Phi^T \mu_A^{t,i}. \tag{B.16}
\]
Next, using the asymptotic model of CG from (2.139) yields
\[
\lim_{N \to \infty} \frac{1}{N} w^T \Phi^T \mu_A^{t,i} \overset{a.s.}{=} \sum_{j=0}^i t^{l,i}_t[j] \lim_{N \to \infty} \frac{1}{N} w^T \Phi^{j+1} w \tag{B.17}
\]
\[
= v_w \sum_{j=0}^i r_t^{l,i}_t[j] \lim_{N \to \infty} \frac{1}{N} \text{Tr}\{\Phi^{j+1}\} \tag{B.18}
\]
\[
\overset{(b)}{=} v_w \sum_{j=0}^i r_t^{l,i}_t[j] \gamma_{j+1} \overset{(c)}{=} -v_w \gamma_t^{l,i} \tag{B.19}
\]
where (B.17) follows from (B.11), (B.18) is due to Lemma 6, in (b) we used (4.49) and (c) follows from the asymptotic definition (B.12) of \( \gamma_t^{l,i} \). Next, substituting (B.19) into (B.16) gives
\[
\lim_{N \to \infty} \frac{1}{N} W^T W_t \mu_A^{t,i} = \lim_{N \to \infty} v_w^i v_t^i - v_{B \to A} v_w \gamma_t^{l,i} \overset{a.s.}{=} \lim_{N \to \infty} v_w \left( 1 - \frac{1}{N} z_t^T \mu_A^{t,i} \right) \tag{B.20}
\]
\[
= v_w \lim_{N \to \infty} \frac{1}{N} z_t^T \mu_A^{t,i}, \tag{B.21}
\]
where (B.20) comes from (B.13) and (B.21) follows from the definition of \( \nu_t^i \). Then (B.21) together with (B.15) implies
\[
\lim_{N \to \infty} \eta_t \overset{a.s.}{=} \delta v_w + \lim_{N \to \infty} b_{t}^{i-1} \eta_t^{i-1} - v_w \frac{1}{N} z_t^T \mu_A^{t,i} \overset{a.s.}{=} \lim_{N \to \infty} v_w \left( \delta - \frac{1}{N} z_t^T \mu_A^{t,i} \right) + b_{t}^{i-1} \eta_t^{i-1} \tag{B.22}
\]
which matches the iteration (3.8) from Theorem 2. Additionally, by reapplying (B.14) to itself \( i \) times, we arrive at
\[
\lim_{N \to \infty} \frac{1}{N} w^T \mu_A^{t,i} \overset{a.s.}{=} \lim_{N \to \infty} \sum_{j=0}^{i-1} a_t^j \eta_t^j. \tag{B.23}
\]
Lastly, based on the definition of $\eta^t$, we have the following limiting initialization
\[
\lim_{N \to \infty} \eta^0_t = \lim_{N \to \infty} \frac{1}{N} w^T \mathbf{p}^0_t = \lim_{N \to \infty} \frac{1}{N} w^T \mathbf{z}_t \overset{a.s.}{=} \delta v_{in},
\]
which completes the proof. ■
Appendix C

Proof of Theorem 3

Next, we prove Theorem 3 that provides an alternative version of the limit

\[ v_{t,i}^{A \to B} \overset{a.s.}{=} (\gamma_{t,i}^A)^{-2} \lim_{N \to \infty} \frac{1}{N} \| A^T \mu_{t,i}^A \|^2 - v_{B \to A}' \]  \hspace{1cm} (C.1)

For this, we define \( \Phi = AA^T \) and note that based on the definition of \( W_t = \hat{v}_w I + \hat{v}_{B \to A}' AA^T \) we have

\[ \Phi = \frac{W_t - \hat{v}_w I_M}{\hat{v}_{B \to A}'} . \]  \hspace{1cm} (C.2)

Then, by expanding the norm in (C.1) and using (C.2) gives

\[ v_{t,i}^{A \to B} \overset{a.s.}{=} \lim_{N \to \infty} \frac{(\mu_{t,i}^A)^T W_t \mu_{t,i}^A - \hat{v}_w \| \mu_{t,i}^A \|^2}{N \hat{v}_{B \to A}' (\gamma_{t,i}^A)^2} - v_{B \to A}' . \]  \hspace{1cm} (C.3)

As discussed in Section 2.4, the CG output \( \mu_{t,i}^A \) can be represented as a linear combination of the conjugate direction vectors

\[ \mu_{t,i}^A = \sum_{j=0}^{i} a_{ij} p_{ij} \]

where \( (p_i)^T W_i p_j = 0 \) for any \( i \neq j \). Thus, the inner product \( (\mu_{t,i}^A)^T W_t \mu_{t,i}^A \) can be equivalently represented as

\[ (\mu_{t,i}^A)^T W_t \mu_{t,i}^A = \sum_{j=0}^{i} (a_{ij})^2 (p_i)^T W_i p_j . \]  \hspace{1cm} (C.4)
Additionally, using the definition of $a_t^j = \frac{||r_t^j||^2}{(p^r)^T \mathbf{W}_t p_t}$ from Algorithm 3, we can further simplify (C.4) to

$$\left(\mu_A^{i,t}\right)^T \mathbf{W}_t \mu_A^{i,t} = \sum_{j=0}^{i} a_t^j ||r_t^j||^2.$$  \hspace{1cm} (C.5)

Then, by defining $\zeta_t^i = \zeta_t^{i-1} + a_t^i ||r_t^i||^2$ with $\zeta_t^0 = 0$ and substituting these results back into (C.3) completes the proof. \hfill \blacksquare
Appendix D

Proofs of Theorem 4 and of Theorem 8

In the following we will prove Theorem 4 stating that in the LSL, the output of WS-CG $\mu^{t,i}_A$ corresponds to a linear mapping of all the vector $z_\tau$, $\tau \leq t$. Then we will generalize this result to a unified approximation framework from Section 4.2.2.

The WS-CG algorithm that we will consider can be summarized as

\[ \mu^{t,i}_A = \mu^{t-1,i}_A + a^{i-1}_t p^{i-1}_t, \]  
\[ p^i_t = z_t - W_t \mu^{t,i}_A + b^{i-1}_t p^{i-1}_t, \]  

where $a^{i}_t$ and $b^{i}_t$ are the scalars as in Algorithm 3 and we use the initializations (4.1)-(4.2) for $t \geq 1$. For $t = 0$, when we use the zero initialization for CG, Theorem 4 was proved in [92]. For $t \geq 1$, the following analysis is based on the observation that given the sets of scalars \{a^{i}_t\}_{j=0} and \{b^{i}_t\}_{j=0}, the updates (D.1)-(D.2) become linear mappings. Because they are linear, it is possible to separate the impact of each input vector $z_t$, $\mu^{t-1,i}_A$, and $p^{i-1}_t$ on $\mu^{t,i}_A$ and $p^i_t$ and reformulate (D.1)-(D.2) as

\[ \mu^{t,i}_A = F^{t,i}_z(z_t) + F^{t,i}_\mu(\mu^{t-1,i}_A) + F^{t,i}_p(p^{i-1}_t), \]  
\[ p^i_t = G^{t,i}_z(z_t) + G^{t,i}_\mu(\mu^{t-1,i}_A) + G^{t,i}_p(p^{i-1}_t), \]  

where $F^{t,i}_z(\cdot)$ and $G^{t,i}_z(\cdot)$ are some functions and we omitted their dependence on
Lemma 9. For the WS-CG algorithm (D.1)-(D.2) with the initializations (4.1)-(4.2), we have:

- For $F_{z,i}^t$ and $G_{z,i}^t$
  \[ F_{z,i}^t(z_t) = F_{z,i}^{t-1}(z_t) + a_{i}^{t-1}G_{z,i}^{t-1}(z_t) \]  
  \[ G_{z,i}^t(z_t) = z_t - W_tF_{z,i}^t(z_t) + b_{i}^{t-1}G_{z,i}^{t-1}(z_t) \]  
  with $F_{z,i}^0(z_t) = 0_M$ and $G_{z,i}^0(z_t) = z_t$.

- For $F_{p,i}^t$ and $G_{p,i}^t$
  \[ F_{p,i}^t(p_{i-1}^0) = F_{p,i}^{t-1}(p_{i-1}^0) + a_{i}^{t-1}G_{p,i}^{t-1}(p_{i-1}^0) \]  
  \[ G_{p,i}^t(p_{i-1}^0) = z_t - W_tF_{p,i}^t(p_{i-1}^0) + b_{i}^{t-1}G_{p,i}^{t-1}(p_{i-1}^0) \]  
  with $F_{p,i}^0(p_{i-1}^0) = 0_M$ and $G_{p,i}^0(p_{i-1}^0) = b_{i-1}^{t-1}p_{i-1}^0$.

**Proof.** We prove this lemma only for the mappings $F_{z,i}^t(z_t)$ and $G_{z,i}^t(z_t)$, since the proof for the rest is similar. The proof is by induction and begins with $i = 0$, for which we have

\[ \mu_A^{i,0} = 0 \cdot z_t + \mu_A^{i-1,0} + 0 \cdot p_{i-1}^0 \]  
\[ p_i^0 = z_t - W_t\mu_A^{i-1,0} + b_{i-1}^{t-1}p_{i-1}^0 \]  

as follows from (4.1) and (4.2). Therefore, if we define

\[ F_{z,i}^t(z_t) = 0_M \quad G_{z,i}^t(z_t) = z_t, \]  

then the functions $F_{z,i}^t$ and $G_{z,i}^t$ fully cover the dependence of $\mu_A^{i,0}$ and $p_i^0$ from (D.3) and (D.4) on $z_t$. Next we assume (D.5) and (D.6) hold up to $i = k$ and we
prove it for \( i = k + 1 \). Since (D.1) is linear with respect to \( \mu_t^{i-1} \) and \( p_t^{i-1} \) given the scalar \( a_t^k \), we have that

\[
F_z^{t,k+1}(z_t) = F_z^{t,k}(z_t) + a_t^k G_z^{t,k}(z_t),
\]

which confirms (D.5). In the same way we prove (D.6).

The above lemma implies that we can analyze the impact of each input vector \( z_{t-1}, \mu_t^{i-1} \) and \( p_t^{i-1} \) on the output vectors \( \mu_t^i \) and the conjugate direction vector \( p_t^i \) separately. To establish the exact structure of the above linear mappings, we define three pairs of scalar sequences. The first one is

\[
f_z^{t,i}[j] = f_z^{t,i-1}[j] + a_t^i g_z^{t,i-1}[j]
\]

and

\[
g_z^{t,i}[j] = \delta_{j,0} - v_w f_z^{t,i}[j] - v_B f_z^{t,i}[j - 1] + b_t^i g_z^{t,i-1}[j]
\]

with \( f_z^{t,0}[0] = 0, g_z^{t,0}[0] = 1, f_z^{t,i}[j] = 0 \) and \( g_z^{t,i}[j] = 0 \) for \( j \not\in \{0, ..., i\} \) and with \( \delta_{j,0} \) defining the Kronecker delta. The second sequence is defined as

\[
f_\mu^{t,i}[j] = f_\mu^{t,i-1}[j] + a_t^i g_\mu^{t,i-1}[j]
\]

and

\[
g_\mu^{t,i}[j] = -v_w f_\mu^{t,i}[j] - v_B f_\mu^{t,i}[j - 1] + b_t^i g_\mu^{t,i-1}[j]
\]

with \( f_\mu^{t,0}[0] = 1, g_\mu^{t,0}[0] = -v_w, g_\mu^{t,0}[1] = -v_B, f_\mu^{t,i}[j] = 0 \) for \( j \not\in \{0, ..., i\} \) and \( g_\mu^{t,i}[j] = 0 \) for \( j \not\in \{0, ..., i \} \). Lastly, the third sequence is

\[
f_p^{t,i}[j] = f_p^{t,i-1}[j] + a_t^i g_p^{t,i-1}[j]
\]

and

\[
g_p^{t,i}[j] = -v_w f_p^{t,i}[j] - v_B f_p^{t,i}[j - 1] + b_t^i g_p^{t,i-1}[j]
\]

with \( f_p^{t,0}[0] = 0, g_p^{t,0}[0] = b_t^{i-1}, f_p^{t,i}[j] = 0 \) and \( g_p^{t,i}[j] = 0 \) for \( j \not\in \{0, ..., i \} \).

With these definitions, first, we present the alternative linear forms of \( F_z^i(z_t) \) and \( G_z^i(z_t) \) proved in [92].

**Lemma 10.** Lemma 1 in [92]: Let \( \Phi = \mathbf{A} \mathbf{A}^T \). Then \( F_z^i(z_t) \) and \( G_z^i(z_t) \) from (D.5) and (D.6) can be alternatively represented as

\[
F_z^i(z_t) = \sum_{j=0}^{i} f_z^{t,i}[j] \Phi^j z_t
\]

and

\[
G_z^i(z_t) = \sum_{j=0}^{i} g_z^{t,i}[j] \Phi^j z_t.
\]
We follow the same idea and present the alternative mappings for the pairs $F^i_t(\mu_{A}^{t,i})$ and $G^i_t(\mu_{A}^{t,i})$, and $F^i_p(p_{t-1}^{i})$ and $G^i_p(p_{t-1}^{i})$.

**Lemma 11.** Let $\Phi = AA^T$. Then we have

\[
F^i_t(\mu_{A}^{t,i}) = \sum_{j=0}^{i} f^t_{\mu}[j] \Phi^j \mu_{A}^{t-1,i} \quad \text{(D.23)}
\]

\[
G^i_t(\mu_{A}^{t,i}) = \sum_{j=0}^{i+1} g^t_{\mu}[j] \Phi^j \mu_{A}^{t-1,i} \quad \text{(D.24)}
\]

\[
F^i_p(p_{t-1}^{i}) = \sum_{j=0}^{i} f^t_{p}[j] \Phi^j p_{t-1}^{i} \quad \text{(D.25)}
\]

\[
G^i_p(p_{t-1}^{i}) = \sum_{j=0}^{i} g^t_{p}[j] \Phi^j p_{t-1}^{i} \quad \text{(D.26)}
\]

**Proof.** The proof is done by induction. First, we consider the recursion (D.7)-(D.8) and prove the mappings (D.23)-(D.24). We begin the induction with $i = 0$, for which we set $f^t_{\mu}[0] = 1$ and use the right hand side of (D.23) to obtain

\[
f^t_{\mu}[0] \Phi^0 \mu_{A}^{t-1,i} = \mu_{A}^{t-1,i}, \quad \text{(D.27)}
\]

which matches the initialization $F^i_t(\mu_{A}^{t,i}) = \mu_{A}^{t,i}$ from Lemma 9. Similarly, setting

\[
g^t_{\mu}[0] = -v_w
\]

\[
g^t_{\mu}[1] = -v_{B \rightarrow A}^t
\]

and using the right hand side of (D.24) gives

\[
g^t_{\mu}[0] \Phi^0 \mu_{A}^{t-1,i} + g^t_{\mu}[1] \Phi^1 \mu_{A}^{t-1,i} = -v_w g^t_{\mu}[0] \Phi^0 \mu_{A}^{t-1,i} - v^t_{B \rightarrow A} \Phi^1 \mu_{A}^{t-1,i} = -W_t \mu_{A}^{t-1,i},
\]

which matches the initialization $G^i_t(\mu_{A}^{t,i}) = -W_t \mu_{A}^{t,i}$ from Lemma 9.

Next, we assume (D.23)-(D.24) hold up to $i = k$ and we aim to confirm it for
\( i = k + 1 \). From (D.23) we have
\[
\sum_{j=0}^{k+1} f_{t,k+1}^j \Phi^j A^{-1,i} = \sum_{j=0}^{k+1} (f_{t,k}^j + a_t^k g_{t,k}^j) \Phi^j A^{-1,i} \\
= \sum_{j=0}^{k} f_{t,k}^j \Phi^j A^{-1,i} + a_t^k \sum_{j=0}^{k+1} g_{t,k}^j \Phi^j A^{-1,i} \\
= \Phi F_{t,1}^j (A^{-1,i}) + a_t^k G_{t,1}^j (A^{-1,i}),
\]
where (a) comes from (D.17), (b) is due to \( f_{t,k}^{k+1} = 0 \) by definition and (c) is by the induction hypothesis. By comparing this result to (D.7), we confirm the validity of (D.23). Similarly, using (D.24) yields
\[
\sum_{j=0}^{k+2} g_{t,k}^{j+1} \Phi^j A^{-1,i} = \sum_{j=0}^{k+2} \left( -v_w - v_B f_{t,k+1} + b_t^k g_{t,k}^j \right) \Phi^j A^{-1,i}. \tag{D.29}
\]
Here we have that
\[
\sum_{j=0}^{k+2} f_{t,i+1}^j \Phi^j A^{-1,i} = \sum_{j=0}^{k+1} f_{t,i+1}^j \Phi^j A^{-1,i} \\
= \Phi F_{t,1}^i (A^{-1,i}), \tag{D.30}
\]
where (a) is due to \( f_{t,i+1}^{i+2} = 0 \) as follows from the definition. Also
\[
\sum_{j=0}^{k+2} g_{t,i}^{j+1} \Phi^j A^{-1,i} = \sum_{j=0}^{k+1} g_{t,i}^{j+1} \Phi^j A^{-1,i} \\
= \Phi G_{t,1}^i (A^{-1,i}), \tag{D.31}
\]
where (a) is because \( g_{t,i}^{k+2} = 0 \) by definition. Lastly, by making the substitution \( e = j - 1 \), we can obtain
\[
\sum_{j=0}^{k+2} f_{t,k+1}^{j-1} \Phi^j A^{-1,i} = \sum_{e=-1}^{k+1} f_{t,k+1}^{e+1} \Phi^e A^{-1,i} \\
= \Phi F_{t,1}^{e+1} (A^{-1,i}), \tag{D.32}
\]
where (a) is due to \( f_{t,k+1}^{e+1} = 0 \) by definition. Inserting (D.30), (D.31) and
Proofs of Theorem 4 and of Theorem 8

(D.32) into (D.29) gives

\[
\sum_{j=0}^{k+2} g^{t+1}_\mu[j] \Phi^j \mu^{t-1,i}_A = -v_w (F_{t+1}^{t,i} (\mu^{t-1,i}_A) - v_B^{t,i} A \Phi F_{t+1}^{t,i} (\mu^{t-1,i}_A) + b^k_\mu \Phi^i \mu^{t-1,i}_A
\]

\[
= -W_t F_{t+1}^{t,i} (\mu^{t-1,i}_A) + b^k_\mu \Phi^i \mu^{t-1,i}_A
\]

\[
= C^{t+1}_\mu (\mu^{t-1,i}_A)
\]

which completes the proof for (D.23)-(D.24). To prove a similar result for (D.25)-(D.26), first, we note that the sequences (D.19)-(D.20) are exactly the same as (D.17)-(D.18) so the induction part of the proof will be the same as for proving (D.23)-(D.24). The only difference between the two sequences are the initializations. Following the same idea, we set

\[
f^{t,0}_p[0] = 0
\]

and use the right hand side of (D.25) to show

\[
f^{t,0}_p[0] \Phi^0 p^{t-1}_i = 0_M,
\]

which matches the initialization \( F^{t,0}_p(p^{t-1}_i) = 0_M \) from Lemma 9. Similarly, let \( g^{t,0}_p[0] = b^{t-1}_i \) so that the right hand side of (D.26) is equal to

\[
g^{t,0}_p[0] \Phi^0 p^{t-1}_i = b^{t-1}_i \Phi^0 p^{t-1}_i = b^{t-1}_i p^{t-1}_i,
\]

which is consistent with the initialization \( G^{t,0}_p(p^{t-1}_i) = b^{t-1}_i p^{t-1}_i \) from Lemma 9.

Lemmas 10 and 11 together with (D.3)-(D.4) imply that the CG output \( \mu^{t,i}_A \) and the vector \( p^{t,i}_i \) can be represented as linear mappings of the vectors \( z_t, \mu^{t-1,i}_A \) and \( p^{t-1,i}_i \) given the set of scalars \( \{a^{t,i}_j\}_{j=0} \) and \( \{b^{t,i}_j\}_{j=0} \) as

\[
\mu^{t,i}_A = F^{t,i}_z z_t + F^{t,i}_\mu \mu^{t-1,i}_A + F^{t,i}_p p^{t-1}_i
\]

\[
p^{t,i}_i = G^{t,i}_z z_t + G^{t,i}_\mu \mu^{t-1,i}_A + G^{t,i}_p p^{t-1}_i.
\]

Now we can finish the derivation of Theorem 4 in two steps. First, we repeatedly apply Lemma 11 to (D.35) to obtain a mapping from the set \( \{z_t\}_{t=0}^{t} \) to \( \mu^{t,i}_A \). And next, we group up the terms that scale \( \Phi \) with the same power \( k \). For this, we
present two sequences $r^{t,i}_\tau[k]$ and $u^{t,i}_\tau[k]$ that for $0 \leq \tau < t$ are defined as

$$
\begin{align*}
    r^{t,i}_\tau[k] &= \sum_{j=0}^{d} f^{t,i}_\mu[j] r^{t-1,i}_\tau[k-j] + f^{t,i}_p[j] u^{t-1,i-1}_\tau[k-j] \quad \text{(D.37)} \\
    u^{t,i}_\tau[k] &= \sum_{j=0}^{d} g^{t,i}_\mu[j] r^{t-1,i}_\tau[k-j] + g^{t,i}_p[j] u^{t-1,i-1}_\tau[k-j] \quad \text{(D.38)}
\end{align*}
$$

with $r^{t,i}_\tau[k] = 0$ and $u^{t,i}_\tau[k] = 0$ for $k \not\in \{0, ..., (t-\tau)i\}$. For $\tau = t$ we set $r^{t,i}_\tau[k] = f^{t,i}_z[k]$ and $u^{t,i}_\tau[k] = g^{t,i}_z[k]$. With these definitions, we can show by induction that

$$
\begin{align*}
    \mu^{t,i}_A &= \sum_{\tau=0}^{t} \sum_{k=0}^{(t-\tau)i} r^{t,i}_\tau[k] \Phi^k z_\tau \quad \text{(D.39)} \\
    p^{t}_i &= \sum_{\tau=0}^{t} \sum_{k=0}^{(t-\tau)i} u^{t,i}_\tau[k] \Phi^k z_\tau. \quad \text{(D.40)}
\end{align*}
$$

We start the induction with $t = 0$, for which WS-CG and the regular CG are the same, and in (D.35)-(D.36) we only have the mappings $F^{t,i}_z$ and $G^{t,i}_z$. In this case, by setting $r^{0,i}_\tau[k] = f^{0,i}_z[k]$ and $u^{0,i}_\tau[k] = g^{0,i}_z[k]$, and following Lemma 10, we confirm (D.39) and (D.40) for $t = 0$.

Next, assume (D.39) and (D.40) hold up to iteration $t$ and we would like to confirm them for the iteration $t + 1$. The mapping from $z_{t+1}$ to $\mu^{t+1,i}_A$ can be confirmed by setting $r^{t+1,i}_\tau[k] = f^{t+1,i}_z[k]$ and applying Lemma 10 to (D.35). For the mapping from $\{z_\tau\}_{\tau=0}^{t}$ to $\mu^{t+1,i}_A$ we apply the results from Lemma 11 to (D.39) and (D.40) to obtain

$$
\begin{align*}
    F^{t+1,i}_\mu \mu^{t+1,i}_i + F^{t+1,i}_p p^{t-1}_i &= F^{t,i}_\mu \sum_{\tau=0}^{t} \sum_{k=0}^{(t-\tau)i} r^{t,i}_\tau[k] \Phi^k z_\tau + F^{t,i}_p \sum_{\tau=0}^{t} \sum_{k=0}^{(t-\tau)i} u^{t,i-1}_\tau[k] \Phi^k z_\tau \\
    &= \sum_{\tau=0}^{t} \sum_{k=0}^{(t-\tau)i} \sum_{j=0}^{i} f^{t,i}_\mu[j] \Phi^j r^{t,i}_\tau[k] \Phi^k z_\tau + \sum_{\tau=0}^{t} \sum_{k=0}^{(t-\tau)i} \sum_{j=0}^{i} f^{t,i}_p[j] \Phi^j u^{t,i-1}_\tau[k] \Phi^k z_\tau \\
    &= \sum_{\tau=0}^{t} \sum_{k=0}^{(t-\tau)i} \left( f^{t,i}_\mu[j] r^{t,i}_\tau[k] + f^{t,i}_p[j] u^{t,i-1}_\tau[k] \right) \Phi^{k+j} z_\tau \\
    &= \sum_{\tau=0}^{t} \sum_{k=0}^{(t-\tau)i} \sum_{j=0}^{i} \phi^{t,i}_\tau[j,k] \Phi^{k+j} z_\tau. \quad \text{(D.41)}
\end{align*}
$$

Proofs of Theorem 4 and of Theorem 8...
Proofs of Theorem 4 and of Theorem 8 where we defined an auxiliary scalar function

\[
\phi^{t,i}_{\tau}[j,k] = f^{t,i}_{\mu}[j]r^{t,i}_{\tau}[k] + f^{t,i}_{p}[j]u^{t,i-1}_{\tau}[k].
\]  

(D.42)

Note that the functions \( f^{t,i}_{\mu}[j] \) and \( g^{t,i}_{\mu}[j] \) are finite for finite \( i \) since \( a^t_i, b^t_i, v_w \) and \( v^t_{B \rightarrow A} \) are finite and (D.15)-(D.20) involve only summation and subtraction operations. Then, because \( r^{t,i}_{\tau}[k] \) and \( u^{t,i}_{\tau}[k] \) are linear combinations of \( f^{t,i}_{\mu}[j] \) and \( g^{t,i}_{\mu}[j] \), we have that \( \phi^{t,i}_{\tau}[j,k] \) is also finite. Therefore, we can change the order of summation with respect to \( k \) and \( j \) in (D.41) given \( i \) is finite. Then, by the change of variables \( k' = k + j \), we can arrive at

\[
F^{t+1,i}_{\mu} \mu^i_t + F^{t+1,i}_{p} p^{i-1}_t = \sum_{\tau=0}^{t} \sum_{j=0}^{i} \sum_{k'=j}^{(t-\tau)i+j} \phi^{t,i}_{\tau}[j,k'-j] \Phi^{k'}_{z_{\tau}}.
\]  

(D.43)

Next, note that from the definition of \( r^{t,i}_{\tau}[k] \) and \( u^{t,i}_{\tau}[k] \) we have that \( \phi^{t,i}_{\tau}[j,k] = 0 \) for \( k \not\in \{0, \ldots, (t-\tau)i\} \), which implies that in the last result we can set the lower bound of the summation for \( k' \) to zero without affecting the result. Similarly, we can increase the upper bound of the same summation. Because in (D.41) the maximum value of \( j \) is \( i \), we change the upper bound of the summation for \( k' \) to \((t-\tau)i + i = (t + 1 - \tau)i \). With these changes we obtain

\[
F^{t+1,i}_{\mu} \mu^i_t + F^{t+1,i}_{p} p^{i-1}_t = \sum_{\tau=0}^{t} \sum_{j=0}^{i} \sum_{k'=0}^{(t+1-\tau)i} \phi^{t,i}_{\tau}[j,k'-j] \Phi^{k'}_{z_{\tau}}.
\]

Lastly, by changing the order of the summation with respect to \( j \) and \( k' \) and defining

\[
r^{t,i}_{\tau}[k'] = \sum_{j=0}^{i} \phi^{t,i}_{\tau}[j,k'-j],
\]  

(D.44)

we arrive at (D.39) for \( t + 1 \).

Because the structure of \( r^{t,i}_{\tau}[k] \) and of \( u^{t,i}_{\tau}[k] \) are similar and because the structure of the updates for \( \mu^{t,i}_{\mu} \) and for \( p^{t}_i \) are the same, the proof of (D.40) follows exactly the same steps as for proving (D.39) considered above. 

\[\blacksquare\]
D.1 Proof of Theorem 8

Next we prove that the unified approximation framework (4.37), (4.40), can be written as a linear mapping of all the residual vectors \( \{ z_\tau \}_{\tau=0} \). Note that this framework exactly matches the iterative scheme (D.1)-(D.2) and the initialization for WS-CG provided the matrix \( W_t \) and the pair of sets of scalars \( \{ a_{i}^t \}_{i=0} \) and \( \{ b_{i}^t \}_{j=0} \). Therefore, it is straightforward to show that Lemma 9 holds for the unified framework (4.37), (4.40). What changes is the form of the three pairs of sequences (D.15)-(D.20) due to the change in the matrix \( W_t \). Recall that we used WS-CG to approximate the solution \( \mu_t A \) to the SLE

\[
W_t \mu_t A = z_t, \quad (D.45)
\]

where

\[
W_t = v_w I + v_{B \rightarrow A}^t AA^T. \quad (D.46)
\]

The unified WS framework is also used to approximate the solution to the SLE (D.45), but where the matrix \( W_t \) is a scaled version of (D.46),

\[
W_t = \rho_t I + AA^T, \quad (D.47)
\]

where \( \rho_t = \frac{v_w}{v_{B \rightarrow A}} \) is a finite non-zero scalar. It is straightforward to verify that by changing \( v_w \) to \( \rho_t \) and \( v_{B \rightarrow A}^t \) to 1, the Lemmas (10) and (11) can be adapted for the unified WS framework. For this, we define three pairs of scalar sequences. The first one is

\[
f_{x^t}^{i,j}[j] = f_{x^t}^{i,j-1}[j] + a_{i}^t g_{x^t}^{i,j-1}[j] \quad (D.48)
\]

\[
g_{x^t}^{i}[j] = \delta_{j,0} - \rho_t f_{x^t}^{i,j}[j] - f_{x^t}^{i,j}[j-1] + b_{i}^t g_{x^t}^{i,j-1}[j] \quad (D.49)
\]

with \( f_{x^t}^{i,0}[0] = 0, g_{x^t}^{i,0}[0] = 1, f_{x^t}^{i,j}[j] = 0 \) and \( g_{x^t}^{i,j}[j] = 0 \) for \( j \notin \{0, ..., i\} \) and with \( \delta_{j,0} \) defining the Kronecker delta. The second sequence is defined as

\[
f_{\mu}^{i,j}[j] = f_{\mu}^{i,j-1}[j] + a_{i}^t g_{\mu}^{i,j-1}[j] \quad (D.50)
\]

\[
g_{\mu}^{i}[j] = -\rho_t f_{\mu}^{i,j}[j] - f_{\mu}^{i,j}[j-1] + b_{i}^t g_{\mu}^{i,j-1}[j] \quad (D.51)
\]
with \( f^{t,0}_\mu[0] = 1, g^{t,0}_\mu[0] = -\rho, f^{t,0}_\mu[1] = -1, f^{t,i}_\mu[j] = 0 \) for \( j \not\in \{0, \ldots, i\} \) and \( g^{t,i}_\mu[j] = 0 \) for \( j \not\in \{0, \ldots, i+1\} \). Lastly, the third sequence is

\[
\begin{align*}
 f^{t,i}_\mu[j] &= f^{t,i-1}_\mu[j] + a^i_t g^{t,i-1}_\mu[j] \\
 g^{t,i}_\mu[j] &= -\rho_t f^{t,i}_\mu[j] - f^{t,i}_\mu[j-1] + b^i_t g^{t,i-1}_\mu[j]
\end{align*}
\]  

(D.52)

(D.53)

with \( f^{t,0}_\mu[0] = 0, g^{t,0}_\mu[0] = b^i_t - 1, f^{t,i}_\mu[j] = 0 \) and \( g^{t,i}_\mu[j] = 0 \) for \( j \not\in \{0, \ldots, i\} \).

With these definitions, one can follow exactly the same steps as in the proof of Lemmas 10-11 to establish that for the unified WS approximation framework (4.37), (4.40), the output \( \mu^{t,i}_A \) and the memory vector \( p^i_t \) at the inner-loop iteration \( i \) can be equivalently represented as

\[
\begin{align*}
 \mu^{t,i}_A &= F^{t,i}_\mu \beta_t + F^{t,i}_{\mu} \mu^{t-1,i}_A + F^{t,i}_p p^{t-1}_i \\
 p^i_t &= G^{t,i}_\mu \beta_t + G^{t,i}_{\mu} \mu^{t-1,i}_A + G^{t,i}_p p^{t-1}_i
\end{align*}
\]  

(D.54)

(D.55)

where

\[
\begin{align*}
 F^{t,i}_\mu(z_t) &= \sum_{j=0}^{i} f^{t,i}_\mu[j] \Phi^j z_t \\
 G^{t,i}_\mu(z_t) &= \sum_{j=0}^{i} g^{t,i}_\mu[j] \Phi^j z_t \\
 F^{t,i}_{\mu}(\mu^{t-1,i}_A) &= \sum_{j=0}^{i} f^{t,i}_{\mu}[j] \Phi^j \mu^{t-1,i}_A \\
 G^{t,i}_{\mu}(\mu^{t-1,i}_A) &= \sum_{j=0}^{i+1} g^{t,i}_{\mu}[j] \Phi^j \mu^{t-1,i}_A \\
 F^{t,i}_p(p^{t-1}_i) &= \sum_{j=0}^{i} f^{t,i}_p[j] \Phi^j p^{t-1}_i \\
 G^{t,i}_p(p^{t-1}_i) &= \sum_{j=0}^{i} g^{t,i}_p[j] \Phi^j p^{t-1}_i
\end{align*}
\]  

(D.56)

(D.57)

(D.58)

(D.59)

(D.60)

(D.61)

with \( f^{t,i}_\mu[j], g^{t,i}_\mu[j], f^{t,i}_{\mu}[j], g^{t,i}_p[j], f^{t,i}_p[j] \) and \( g^{t,i}_p[j] \) being as in (D.48)-(D.53). Simi-

163
Proofs of Theorem 4 and of Theorem 8

larly, we can define the final sequences $v^{t,d}_τ[k]$ and $e^{t,d}_τ[k]$ for $0 ≤ τ < t$ as

$$v^{t,d}_τ[k] = \sum_{j=0}^{d} f^{t,d}_µ[j]v^{t-1,d}_τ[k - j] + f^{t,d}_p[j]e^{t-1,i-1}_τ[k - j] \tag{D.62}$$

$$e^{t,d}_τ[k] = \sum_{j=0}^{d} g^{t,d}_µ[j]v^{t-1,d}_τ[k - j] + g^{t,d}_p[j]e^{t-1,i-1}_τ[k - j] \tag{D.63}$$

with $v^{t,i}_τ[k] = 0$ and $e^{t,i}_τ[k] = 0$ for $k \not\in \{0, ..., (t-τ)i\}$. For $τ = t$ we set $v^{t,i}_τ[k] = f^{t,i}_z[k]$ and $e^{t,i}_τ[k] = g^{t,i}_z[k]$. With these definitions, we can follow the same steps as in the proof of (D.39)-(D.40) to show that (D.54)-(D.55) are equivalent to

$$\mu^{t,i}_A = \sum_{τ=0}^{t} \sum_{k=0}^{(t-τ)i} v^{t,i}_τ[k]\Phi^k Z_τ \tag{D.64}$$

$$p^i = \sum_{τ=0}^{t} \sum_{k=0}^{(t-τ)i} e^{t,i}_τ[k]\Phi^k Z_τ. \tag{D.65}$$

This completes the proof of Theorem 8. ■
Appendix E

Proofs of Theorem 5 and of Theorem 9

Here, we prove Theorem 5 that establishes the closed-form solution (4.10) for the divergence $\gamma_{\tau}^{t,i}$ of WS-CG with respect to $q_{\tau}$ and the SE (4.11) of the intrinsic variance $v_{\tau}^{t,i}$ and $v_{A \rightarrow B}$. Later, we prove that a similar closed-form solution for $\gamma_{\tau}^{t,i}$ and a SE for $v_{A \rightarrow B}$ hold for the unified WS approximation framework (4.37), (4.40). We begin with the former case and in the following we let Assumptions B1-B3 from page 42 hold. By applying (4.9) to (4.5) and noting that $z_{\tau} = w - Aq_{\tau}$ we obtain

$$
\lim_{N \to \infty} \gamma_{\tau}^{t,i} = \lim_{N \to \infty} \frac{1}{N} \nabla q_{\tau} \cdot \left( A^T \sum_{\tau' = 0}^{t} \sum_{k=0}^{(t-\tau)} r_{\tau'}^{t,i}[k] \Phi^k z_{\tau'} \right)
$$

$$
= \lim_{N \to \infty} - \frac{1}{N} \text{Tr} \left\{ A^T \sum_{k=0}^{(t-\tau)} r_{\tau}^{t,i}[k] \Phi^k A \right\}
$$

$$
= \lim_{N \to \infty} - \frac{1}{N} \text{Tr} \left\{ \sum_{k=0}^{(t-\tau)} r_{\tau}^{t,i}[k] U A^{k+1} U^T \right\}
$$

$$
= \lim_{N \to \infty} - \sum_{k=0}^{(t-\tau)} r_{\tau}^{t,i}[k] \frac{1}{N} \text{Tr} \left\{ A^{k+1} \right\}
$$

$$
= \lim_{N \to \infty} - \sum_{k=0}^{(t-\tau)} r_{\tau}^{t,i}[k] x_{k+1}, \quad (E.1)
$$
where $\Phi = AA^T$, $\chi_k$ is as in (4.49) and we used the fact that $Tr\{UA^kU^T\} = Tr\{A^k\}$ for any orthonormal matrix $U$. This proves (4.10).

Next, to work out the result (4.11) for the variance $v_{A \rightarrow B}^{t,i}$, we only need to show that $\frac{1}{N} \left| A^T \mu_{A}^{t,i} \right|^2 = \Omega_i^t$, where $\Omega_i^t$ is as in (4.12). Then, by using (4.9), we can obtain

$$
\lim_{N \rightarrow \infty} \frac{1}{N} \left| A^T \mu_{A}^{t,i} \right|^2 = \lim_{N \rightarrow \infty} \frac{1}{N} \left| \cdot \right| = \lim_{N \rightarrow \infty} \frac{1}{N} \left| \cdot \right|
$$

where in (E.2) we used (B.11). Next, by defining an auxiliary function $g(q_{\tau}) = \Phi^{k+j+1}Aq_{\tau}$, using (4.6) together with (4.5), and following the same steps as in the proof of Lemma 1 in Appendix B, we can show that

$$
\lim_{N \rightarrow \infty} \frac{1}{N} q_{\tau}^T A^T g(q_{\tau}) = \lim_{N \rightarrow \infty} \psi_{\tau', \tau} \frac{1}{N} \nabla_{q_{\tau}} \cdot \left( A^T g(q_{\tau}) \right) = \lim_{N \rightarrow \infty} \psi_{\tau', \tau} \frac{1}{N} Tr \left\{ A^T \Phi^{k+j+1}A \right\} = \lim_{N \rightarrow \infty} \psi_{\tau', \tau} \chi_{k+j+2}.
$$

where

$$
\psi_{\tau', \tau} = \frac{1}{N} q_{\tau'}^T q_{\tau}
$$

Additionally, since $w$ is a zero-mean Gaussian vector, we can use Lemma 6 to obtain

$$
\lim_{N \rightarrow \infty} \frac{1}{N} w^T \Phi^{k+j+1}w \overset{a.s.}{=} v_w \lim_{N \rightarrow \infty} \frac{1}{N} Tr \left\{ \Phi^{k+j+1} \right\} = v_w \chi_{k+j+1}.
$$
Substituting (E.4) and (E.6) into (E.3) gives the result as in (4.11).

E.1 Asymptotics of $a^j_t$ and $b^j_t$ in WS-CG-VAMP

Just above we confirmed the asymptotic result (4.11), but for it to be a proper SE, we need all the components involved to be functions of either $v_w$, $\chi_j$ or $\psi_{\tau,\tau'}$. In (4.11) we have the function $\Omega^{i}_{t}$ that depends on $r^{i}_{\tau}$, which is a function of $f^{i}_{\mu}[j]$ (and other similar functions). This function is formulated in terms of $v_w$ and $v^{t}_{B\rightarrow A}$, but also in terms of $\{a^{j}_{t}\}_{j=0}^{i}$ and $\{b^{j}_{t}\}_{j=0}^{i}$ from the CG algorithm. Therefore, to complete the derivation of the SE, we need to define the asymptotic behaviour of these two sets of scalars. We begin with

$$a^i_t = \frac{\|r^i_t\|^2}{(p^i_t)^T W_i p^i_t} = \frac{\|z_t - W_{i} \mu_{A}^{i,t}\|^2}{(p^i_t)^T W_i p^i_t},$$

(E.7)

where we used the definition of the residual vector $r^i_t$ from the CG algorithm. The norm in the numerator of (E.7) corresponds to

$$\lim_{N\to\infty} \frac{1}{N} \|z_t - W_{i} \mu_{A}^{i,t}\|^2 = \lim_{N\to\infty} \frac{1}{N} \left( \|z_t\|^2 - 2z^T T W_{i} \mu_{A}^{i,t} + \|W_{i} \mu_{A}^{i,t}\|^2 \right)$$

$$= E^{i}_{1,t} - 2E^{i}_{2,t} + E^{i}_{3,t} = E^{i}_{t},$$

(E.8)

where we used the asymptotic independence of $w$ and $A_{q,t}$ from (B.11), the fact that $\frac{1}{N} Tr\{AA^T\} = 1$ and Lemma 5 to show that $E^{i}_{1,t} = \lim_{N\to\infty} \frac{1}{N} \|z_t\|^2 \overset{a.s.}{=} \delta v_w + v_{B\rightarrow A}$. Next, for $t = 0$ we can use the conjugacy property (2.134) of $p^i_t$ and the fact that we initialize $p^0_0 = z_0$ to obtain $z^{T}_t W_{i} \mu_{A}^{i,t} = \|z_0\|^2$ so that

$$E^{i}_{2,0} = \lim_{N\to\infty} \frac{1}{N} \|(z_0)\|^2 \overset{a.s.}{=} \delta v_w + v_{B\rightarrow A}.$$  

(E.9)

For $t > 0$, this result does not hold due to the initializations (4.1) and (4.2). Instead, we use (4.9) to obtain

$$E^{i}_{2,t} = \lim_{N\to\infty} \frac{1}{N} z^{T}_t W_{i} \mu_{A}^{i,t}$$

$$= \lim_{N\to\infty} \frac{1}{N} \sum_{\tau=0}^{t} \sum_{k=0}^{(t-\tau)i} r^{i}_{\tau}[k] z^{T}_t W_{i} \Phi^{k}_t z_{\tau}.$$
Next, we can use the definitions

\[ W_t = v_w I + v_{B \rightarrow A}^t A A^T \]  
\[ z_t = w - A q_t, \]

and follow the same steps as in (E.4) and (E.6) to show that

\[
\lim_{N \to \infty} \frac{1}{N} z_t^T W_t \Phi^k z_t \xrightarrow{a.s.} \lim_{N \to \infty} \frac{1}{N} (w^T W_t \Phi^k w + q_t^T W_t \Phi^k q_t) \\
= v_w^2 \chi_k + v_w v_{B \rightarrow A}^t \chi_{k+1} + v_w \psi_{t, \tau} \chi_{k+1} + v_{B \rightarrow A}^t \psi_{t, \tau} \chi_{k+2},
\]

where in the first step we used the asymptotic independence of \( w \) and \( A q_{\tau} (5.16). \)

Thus, \( E_{2,t}^i \) is equal to

\[
E_{2,t}^i \xrightarrow{a.s.} \lim_{N \to \infty} \sum_{t=0}^t \sum_{k=0}^{(t-\tau)i} r_{\tau}^{t,i}[k] \Xi_{t,\tau}^k
\]

for \( t > 0 \). Following exactly the same steps as above, we can show that the norm

\[
\lim_{N \to \infty} \frac{1}{N} \| W_t \mu_{A}^{t,i} \|^2
\]

almost surely converges to

\[
E_{3,t}^i \xrightarrow{a.s.} \lim_{N \to \infty} \frac{1}{N} \| W_t \mu_{A}^{t,i} \|^2
\]

where

\[
\Delta_{t,\tau'}^{j,i} = v_w^3 \chi_j + 3 v_w^2 v_{B \rightarrow A}^t \chi_{j+1} + 3 v_w (v_{B \rightarrow A}^t)^2 \chi_{j+2} \\
+ v_{\psi_{t, \tau}}^2 \psi_{t, \tau} \chi_{j+1} + v_w v_{B \rightarrow A}^t \psi_{t, \tau} \chi_{j+2} + (v_{B \rightarrow A}^t)^2 \psi_{t, \tau} \chi_{j+3}.
\]

Next, we can use (D.40) to show that the denominator of \( a_t^i \) almost surely
converges to
\[
L_t^i = \lim_{N \to \infty} \frac{1}{N} (p_t^i)^T W_t^i p_t^i \\
= \lim_{N \to \infty} \frac{1}{N} \sum_{\tau, \tau'=0}^t \sum_{j=0}^{(t-\tau)} \sum_{k=0}^{(t-\tau')} u_t^i[j] u_t^i[k] z_t^T \Phi^i W_t \Phi^k z_{\tau, \tau'}^i \\
= \lim_{N \to \infty} \sum_{\tau, \tau'=0}^t \sum_{j=0}^{(t-\tau)} \sum_{k=0}^{(t-\tau')} u_t^i[j] u_t^i[k] \gamma_{\tau, \tau'}^{j+k}
\]
with
\[
\gamma_{\tau, \tau'}^{j} = v_t^2 \chi_j + v_t^w v_{B \to A} \chi_{j+1} + \psi_{\tau, \tau'} v_t^w \chi_{j+1} + \psi_{\tau, \tau'} v_{B \to A} \chi_{j+2}.
\]
Lastly, recall that
\[
b_t^i = \frac{\|z_t - W_t \mu_t^{i+1}\|^2}{\|z_t - W_t \mu_t^i\|^2}, \tag{E.14}
\]
which can be defined using the results above. Thus, we can show that \(a_t^i\) and \(b_t^i\) almost surely converge to
\[
\lim_{N \to \infty} a_t^i \overset{a.s.}{=} \frac{E_t^i}{L_t^i}, \\
\lim_{N \to \infty} b_t^i \overset{a.s.}{=} \frac{E_t^{i+1}}{E_t^i}, \tag{E.15}
\]
which are function of \(v_t, \chi_j\) and \(\psi_{\tau, \tau'}\) only.

### E.2 Proof of Theorem 9

Next we prove Theorem 9. From Theorem 8 we know that the output \(\mu_t^{Li}\) of the unified WS approximation framework (4.37), (4.40) is equivalent to
\[
\mu_t^{Li} = \sum_{\tau=0}^t \sum_{k=0}^{(t-\tau)} v_{\tau}^{Li}[k] (AA^T)^k z_{\tau}. \tag{E.16}
\]
Due to the close similarly of the this identity with (4.9) for WS-CG, one can change \(r_{\tau}^{Li}[k]\) to \(v_{\tau}^{Li}[k]\) and follow exactly the same steps as in (E.1) to prove (4.48).

Similarly, because the structure of (E.16) is identical to (4.9), the proof of (4.50) takes the same steps as in (E.3)-(E.6) with the change of the scalar function.
Proofs of Theorem 5 and of Theorem 9

Eventually, by the assumption in Theorem 9, we have that $a^j_t$ and $b^j_t$ for every $0 \leq j \leq i$ converge to some $\bar{a}^j_t$ and $\bar{a}^j_t$ respectively. This completes the proof of the SE (4.50). ■
Appendix F

Proofs of Theorem 6 and Theorem 10

Here we prove the iterative structure of the divergences $\gamma_{\tau,i}^{t,\tau,i}$ in both WS-CG-VAMP and in MP utilizing the unified WS approximation framework. We begin with the former case and prove Theorem 6 using the generalization of the identity (B.12) for the zero-initialized CG,

$$\lim_{N \to \infty} \frac{1}{N} q_i^T A^T \mu_A^{t,i} \overset{a.s.}{=} \lim_{N \to \infty} \sum_{\tau=0}^{t} \psi_{t,\tau} \gamma_{\tau,i}^{t,\tau,i},$$

where $\psi_{t,\tau'} = \frac{1}{N} q_i^T q_{\tau'}$, $q_{\tau'} = x_{\tau' \to A}^T - x$ and $\mu_A^{t,i} = g_A^i(X_{\tau' \to A}^t, \hat{v}_{\tau' \to A}^t)$ is the output of the WS-CG algorithm after $i$ iterations. The proof of (4.6) follows the same steps as in the proof of Lemma 1 in Appendix B. First, recall from Section 4.1.1 that due to the structure of WS-CG, the function $g_A^i(X_{\tau' \to A}^t, \hat{v}_{\tau' \to A}^t)$ at iteration $t$ can be written as

$$g_A^i(X_{\tau' \to A}^t, \hat{v}_{\tau' \to A}^t) = g_A^i(z_0, z_1, ..., z_t, \hat{v}_{\tau' \to A}^t).$$

Additionally, note that $z_\tau = w - A q_\tau = w - U S V^T q_\tau = w - U b_\tau$, where we used the SVD of $A = U S V^T$ and defined $b_\tau = V^T q_\tau$. Importantly, from Theorem 11 we know that $b_\tau$ is zero-mean i.i.d. Gaussian under Assumptions B1-B3 from page 42. Then, under the same assumptions, we can follow the steps leading to
Proofs of Theorem 6 and Theorem 10

(B.10) to obtain a similar result

\[
\lim_{N \to \infty} \frac{1}{N} \mathbf{q}_i^T \mathbf{A}^T \mathbf{g}_A^i (\mathbf{x}_{B \to A}^t, \mathbf{\tilde{v}}_{B \to A}^t)
= \lim_{N \to \infty} \frac{1}{N} \mathbf{b}_i \mathbf{S}^T \mathbf{U}^T \mathbf{g}_A^i (\mathbf{z}_0, \mathbf{z}_1, ..., \mathbf{z}_t, \mathbf{\tilde{v}}_{B \to A}^t)
= \frac{1}{N} \text{Tr} \left\{ \mathbf{b}_i \left( \mathbf{g}_A^i (\mathbf{z}_0, \mathbf{z}_1, ..., \mathbf{z}_t, \mathbf{\tilde{v}}_{B \to A}^t) \right)^T \mathbf{U} \mathbf{S} \right\}
\]

\[
\overset{\text{a.s.}}{=} \frac{1}{N} \text{Tr} \left\{ \sum_{\tau=0}^t \mathbb{E} \left[ \mathbf{b}_i \mathbf{z}_\tau^T \right] \mathbf{J}_{\mathbf{z}_\tau} \left( \mathbf{g}_A^i (\mathbf{z}_0, \mathbf{z}_1, ..., \mathbf{z}_t, \mathbf{\tilde{v}}_{B \to A}^t) \right) \mathbf{U} \mathbf{S} \right\} \quad \text{(F.3)}
\]

\[
\overset{\text{a.s.}}{=} \frac{1}{N} \text{Tr} \left\{ \sum_{\tau=0}^t \mathbb{E} \left[ \mathbf{b}_i \mathbf{b}_\tau^T \right] \mathbf{S}^T \mathbf{U}^T \mathbf{J}_{\mathbf{z}_\tau} \left( \mathbf{g}_A^i (\mathbf{z}_0, \mathbf{z}_1, ..., \mathbf{z}_t, \mathbf{\tilde{v}}_{B \to A}^t) \right) \mathbf{U} \mathbf{S} \right\}
\]

\[
\overset{\text{a.s.}}{=} \sum_{\tau=0}^t \psi_{t,\tau} \frac{1}{N} \text{Tr} \left\{ \mathbf{J}_{\mathbf{b}_\tau} \left( \mathbf{V} \mathbf{U}^T \mathbf{S}^T \mathbf{g}_A^i (\mathbf{b}_0, \mathbf{b}_1, ..., \mathbf{b}_t, \mathbf{\tilde{v}}_{B \to A}^t) \right) \mathbf{V}^T \right\}
\]

\[
\overset{\text{a.s.}}{=} \sum_{\tau=0}^t \psi_{t,\tau} \frac{1}{N} \text{Tr} \left\{ \mathbf{J}_{\mathbf{q}_\tau} \left( \mathbf{A}^T \mathbf{g}_A^i (\mathbf{b}_0, \mathbf{b}_1, ..., \mathbf{b}_t, \mathbf{\tilde{v}}_{B \to A}^t) \right) \right\}
\]

\[
= \sum_{\tau=0}^t \psi_{t,\tau} \gamma_{A,\tau}^{t,i} \quad \text{(F.4)}
\]

where in (F.3) is due to Lemma 6, while the rest of the steps follow the same reasoning as in the derivation of (B.10). This completes the proof of the identity (4.6).

With (F.1) in hand, we can finish the proof of Theorem 6. Based on (F.1), the vector of all \( t \) divergences \( \gamma_{A,\tau}^{t,i} \), where \( \left( \gamma_{A,\tau}^{t,i} \right)_{\tau} = \gamma_{A,\tau}^{t,i} \), follows

\[
\lim_{N \to \infty} \gamma_{A,\tau}^{t,i} \overset{\text{a.s.}}{=} \lim_{N \to \infty} \Psi^{-1} \left( \nu_i^t \mathbf{1}_t - \frac{1}{N} \mathbf{Z}_i^T \mu_{A,i}^{t,i} \right),
\]

where \( (\Psi_t)_{\tau,\tau'} = \frac{1}{N} \mathbf{q}_t \mathbf{q}_{\tau'} \) and \( \nu_i^t = \frac{1}{N} \mathbf{w}^T \mu_{A,i}^{t,i} \) that we need to define iteratively. Note that the steps (B.14)-(B.16) are the same for both zero-initialized CG and WS-CG since the CG routine from Algorithm 3 is the same in both algorithms. What changes is the asymptotic result of the inner-product \( \frac{1}{N} \mathbf{w}^T \Phi \mu_{A,i}^{t,i} \), which due
Proofs of Theorem 6 and Theorem 10

to (4.46) is asymptotically equivalent to

$$\lim_{N \to \infty} \frac{1}{N} w^T \Phi \mu_A^{t,i} = \lim_{N \to \infty} \sum_{\tau=0}^{t} \sum_{k=0}^{(t-\tau)i} r_{\tau}^{t,i}[k] \frac{1}{N} w^T \Phi \Phi k z_{\tau}$$

$\equiv a.s. \sum_{\tau=0}^{t} \sum_{k=0}^{(t-\tau)i} r_{\tau}^{t,i}[k] \frac{1}{N} w^T \Phi \Phi k+1 w$ (F.5)

$\equiv a.s. v_w \sum_{\tau=0}^{t} \sum_{k=0}^{(t-\tau)i} r_{\tau}^{t,i}[k] \frac{1}{N} \text{Tr} \{ \Phi k+1 \}$ (F.6)

$= v_w \sum_{\tau=0}^{t} \sum_{k=0}^{(t-\tau)i} r_{\tau}^{t,i}[k] \chi_{k+1} = -v_w \sum_{\tau=0}^{t} \gamma_{A}^{t,i} = -v_w \| \gamma_{A}^{t,i} \|_1$, (F.7)

where (F.5) follows from (5.16), (F.6) is due to Lemma 6, and (4.10) motivates the step (a). Then, by combining (B.14), (B.15), (F.7) and (4.53), we arrive at the following recursion

$$\nu_i^{t} = \nu_i^{t-1} + a_i \eta_i^{t-1}$$

$$\gamma_{A}^{t,i} = \lim_{N \to \infty} \Psi_{i}^{-1} (\nu_i^{t,i} 1_t - \frac{1}{N} Z_i^T \mu_A^{t,i})$$

$$\eta_i^{t} = v_w (\delta - \nu_i^{t} + v_i^{t} B_A \nu_i^{t,i}) + b_i \eta_i^{t-1}$$

Lastly, we consider the initialization of the recursion. For $t > 0$ we have $\mu_A^{t,0} = \mu_A^{t-1,i}$ so that

$$\nu_i^{0} = \frac{1}{N} w^T \mu_A^{0} = \frac{1}{N} w^T \mu_A^{t-1,i} = \nu_i^{t-1}.$$ (F.8)

Next, recall that we initialize $p_i^{0} = z_t - W_i \mu_A^{t-1,i} + b_i \eta_i^{t-1} 1_t$. Then we have

$$\lim_{N \to \infty} \eta_i^{0} = \lim_{N \to \infty} \frac{1}{N} w^T p_i^{0}$$

$$= \lim_{N \to \infty} \frac{1}{N} w^T (z_t - W_i \mu_A^{t-1,i} + b_i \eta_i^{t-1})$$

$\equiv a.s. \delta v_w - v_w \nu_i^{t-1} + v_i^{t} B_A v_w \| \gamma_{A}^{t-1,i} \|_1 + b_i \eta_i^{t-1}$

$= v_w (\delta - \nu_i^{t-1} + v_i^{t} B_A \| \gamma_{A}^{t-1,i} \|_1) + b_i \eta_i^{t-1}$,
Proofs of Theorem 6 and Theorem 10

where we used (B.16) and (F.7) to show that
\[
\lim_{N \to \infty} \frac{1}{N} w^T W_i \mu_A^{t-1,i} = v_w \nu_{t-1} + v_{B \to A}^t \lim_{N \to \infty} \frac{w^T \Phi^T \mu_A^{t-1,i}}{N} = v_{B \to A}^t v_w \gamma_A^{t-1,i}.
\]

For \( t = 0 \), we have \( \mu_{-1}^t = p_{-1}^t = 0 \) so \( \nu_0^0 = \frac{1}{N} w^T \mu_0^0 = 0 \). Lastly, in the limit, we initialize \( \lim_{N \to \infty} \eta_0^0 = \lim_{N \to \infty} \frac{1}{N} w^T p_0^0 = \lim_{N \to \infty} \frac{1}{N} w^T z_0 \overset{a.s.}{=} \delta v_w \), which completes the proof. ■

\[ \tag{F.7} \]

\[ \tag{F.8} \]

\[ \tag{F.9} \]

\[ \tag{F.10} \]

\[ \tag{F.11} \]

\[ \tag{F.12} \]

F.1 Proof of Theorem 10

The proof of Theorem 10 closely follows the above steps of the proof of Theorem 6 and of Theorem 2. Since an MP algorithm utilizing the unified WS approximation framework (4.37), (4.40) is a long-memory MP, we use the identity (F.1) to construct our iterative estimator. Next, since the unified framework is identical to the CG algorithm from Algorithm 3 provided the scalars \( a_t^j \) and \( b_t^j \), the steps (B.14)-(B.15) are the same for both CG/WS-CG and for the unified framework. However, recall that the unified framework approximates the SLE

\[ W_t \mu_A^t = z_t, \quad (F.9) \]

where

\[ W_t = \rho_t I + AA^T \quad (F.10) \]

with \( \rho = \frac{v_w}{v_{B \to A}} \), while both CG in CG-VAMP and WS-CG in WS-CG-VAMP approximated the SLE (F.9) where

\[ W_t = v_w I + v_{B \to A}^t AA^T. \quad (F.11) \]

This difference affects the inner-product \( \frac{1}{N} w^T W_t \mu_A^t \) in the step (B.16), which now takes the following form

\[
\lim_{N \to \infty} \frac{1}{N} w^T W_t \mu_A^t = \rho_t \nu^t_t + \lim_{N \to \infty} \frac{1}{N} w^T \Phi^T \mu_A^t. \quad (F.12)
\]

Next, we can reuse the result (F.7) to define the asymptotic behaviour of the inner-product \( \lim_{N \to \infty} \frac{1}{N} w^T \Phi^T \mu_A^t \). This in combination with (F.12) implies that

\[ 174 \]
Proofs of Theorem 6 and Theorem 10

\( \eta_t^i \) from (B.15) is equivalent to
\[
\lim_{N \to \infty} \eta_t^i = \delta v_w + \lim_{N \to \infty} b_t^{i-1} \eta_t^{i-1} - \rho_t \nu_t^i + v_w \| \gamma_t^{i,i} \|_1 \\
= v_w \left( \delta - \frac{\nu_t^i}{v_B^t} + \| \gamma_t^{i,i} \|_1 \right) + \lim_{N \to \infty} b_t^{i-1} \eta_t^{i-1}.
\]
This together with (B.14) and (F.1) proves the iterative scheme (4.52)-(4.54).

Lastly, one can follow the steps at the end of the previous section to verify that the initialization of the iterative scheme (4.52)-(4.54) is the same as for WS-CG except for \( \eta_t^0 \) for \( t > 0 \) due to the change in the matrix \( W_t \). Due to the initialization (4.40), the initialization for \( \eta_t^0 \) takes the following form
\[
\lim_{N \to \infty} \eta_t^0 = \lim_{N \to \infty} \frac{1}{N} w^T p_t^0 \\
= \lim_{N \to \infty} \frac{1}{N} w^T (z_t - W_t \mu_t^{l-1,i} + b_t^{l-1,i} p_t^{l-1}) \\
\overset{a.s.}{=} \delta v_w - \frac{v_t^l}{v_B^t} v_{B \to A} + v_w \| \gamma_t^{l-1,i} \|_1 + b_t^{l-1} \eta^{l-1}_t \\
= v_w \left( \delta - \frac{v_t^l}{v_B^t} + \| \gamma_t^{l-1,i} \|_1 \right) + b_t^{l-1} \eta^{l-1}_t.
\]
which completes the proof.
Appendix G

Proof of Lemma

In this section, we prove Lemma 2 which states that Theorem 11 holds for the error vectors \( \tilde{h}_t = \tilde{r}_t - x \) and \( \tilde{q}_t = \tilde{s}_t - x \), where

\[
\tilde{r}_t = \frac{1}{C_r} \left( A^T f_t(\tilde{S}_{t+1}, y) - \tilde{S}_{t+1} \gamma_t \right) \tag{G.1}
\]

\[
\tilde{s}_{t+1} = \frac{1}{C_s} \left( g_t(\tilde{r}_t) - \tilde{r}_t \hat{\alpha}_t \right) \tag{G.2}
\]

and \( \hat{\alpha}_t \) satisfies

\[
\lim_{N \to \infty} \hat{\alpha}_t \overset{a.s.}{=} \lim_{N \to \infty} \alpha_t \tag{G.3}
\]

at every iteration under Assumptions B1-B3 from page 42. The proof is by induction: we assume that \( \frac{1}{N} \| r_t - \tilde{r}_t \|^2 \overset{a.s.}{=} 0 \) holds and that this implies \( \frac{1}{N} \| s_{t+1} - \tilde{s}_{t+1} \|^2 \overset{a.s.}{=} 0 \). The implication that \( \frac{1}{N} \| s_{t+1} - \tilde{s}_{t+1} \|^2 \overset{a.s.}{=} 0 \) leads to \( \frac{1}{N} \| r_{t+1} - \tilde{r}_{t+1} \|^2 \overset{a.s.}{=} 0 \) is proved in the same way. The beginning of the induction, i.e. proving \( \frac{1}{N} \| s_t - \tilde{s}_t \|^2 \overset{a.s.}{=} 0 \) for \( t = 0 \) is automatic, since \( s_0 = \tilde{s}_0 = 0 \) by definition.

Using the triangular inequality, we can bound the difference of \( s_{t+1} \) and \( \tilde{s}_{t+1} \) from (5.6) and (G.2) respectively as

\[
\| s_{t+1} - \tilde{s}_{t+1} \|^2 = \left\| \frac{1}{C_s} (g_t(r_t) - r_t \alpha_t) - \frac{1}{C_s} (g_t(\tilde{r}_t) + \tilde{r}_t \hat{\alpha}_t) \right\|^2 \\
\leq \left\| \frac{1}{C_s} g_t(r_t) - \frac{1}{C_s} g_t(\tilde{r}_t) \right\|^2 + \left\| \frac{1}{C_s} \tilde{r}_t \hat{\alpha}_t - \frac{1}{C_s} r_t \alpha_t \right\|^2. \tag{G.4}
\]
Similarly, we can show the following for the first inner-product from above
\[
\left\| \frac{1}{C_s} g_t(r_t) - \frac{1}{\tilde{C}_s} g_t(\tilde{r}_t) \right\|^2 = \left\| \frac{1}{C_s} g_t(r_t) - \frac{1}{C_s} g_t(\tilde{r}_t) - \frac{1}{C_s} g_t(\tilde{r}_t) + \frac{1}{C_s} g_t(\tilde{r}_t) \right\|^2
\]
\[
= \left\| \frac{1}{C_s} (g_t(r_t) - g_t(\tilde{r}_t)) + \left( \frac{1}{C_s} - \frac{1}{\tilde{C}_s} \right) g_t(\tilde{r}_t) \right\|^2
\]
\[
\leq \left\| \frac{1}{C_s} (g_t(r_t) - g_t(\tilde{r}_t)) \right\|^2 + \left\| \left( \frac{1}{C_s} - \frac{1}{\tilde{C}_s} \right) g_t(\tilde{r}_t) \right\|^2.
\]

(G.5)

Here, we can use Assumption B2 from page 42 about $g_t$ being a Lipschitz continuous function, which implies
\[
\lim_{N \to \infty} \frac{1}{C_s^2 N} \left\| g_t(r_t) - g_t(\tilde{r}_t) \right\|^2 \leq \lim_{N \to \infty} \frac{L}{C_s^2} \frac{1}{N} \left\| r_t - \tilde{r}_t \right\|^2
\]
\[
a.s. \Rightarrow 0,
\]

(G.6)

where $L = O(1)$ is some constant and the last step follows from the induction hypothesis. Recall that the definition of the scalars $C_s$ and $\tilde{C}_s$ are [10, 58]
\[
C_s = 1 - \alpha_t \quad \tilde{C}_s = 1 - \hat{\alpha}_t,
\]
so that $\frac{1}{C_s} - \frac{1}{\tilde{C}_s} = \frac{\hat{\alpha}_t - \alpha_t}{(1 - \alpha_t)(1 - \hat{\alpha}_t)}$. Then we can use (G.3) and Assumption B2 stating that the norm $\frac{1}{N} \left\| g_t(\tilde{r}_t) \right\|^2$ is asymptotically bounded to obtain
\[
\lim_{N \to \infty} \left( \frac{1}{C_s} - \frac{1}{\tilde{C}_s} \right)^2 \frac{1}{N} \left\| g_t(\tilde{r}_t) \right\|^2 = \lim_{N \to \infty} \left( \frac{\hat{\alpha}_t - \alpha_t}{(1 - \alpha_t)(1 - \hat{\alpha}_t)} \right)^2 \frac{1}{N} \left\| g_t(\tilde{r}_t) \right\|^2
\]
\[
a.s. \Rightarrow 0.
\]

(G.7)

Thus, the first component of (G.4) almost surely converges to zero. In the same way, we can analyze the second component. Following the same steps, we obtain
\[
\left\| \frac{1}{C_s} \tilde{r}_t \hat{\alpha}_t - \frac{1}{C_s} r_t \alpha_t \right\|^2 = \left\| \frac{1}{C_s} \tilde{r}_t \hat{\alpha}_t - \frac{1}{C_s} r_t \alpha_t \right\|^2
\]
\[
= \left\| \left( \frac{1}{C_s} - \frac{1}{C_s} \right) \tilde{r}_t \hat{\alpha}_t + \frac{1}{C_s} (\tilde{r}_t \hat{\alpha}_t - r_t \alpha_t) \right\|^2
\]
\[
\leq \left( \frac{1}{C_s} - \frac{1}{C_s} \right)^2 \hat{\alpha}_t^2 \tilde{r}_t^2 + \frac{1}{C_s^2} \left\| \tilde{r}_t \hat{\alpha}_t - r_t \alpha_t \right\|^2.
\]

(G.8)

Since $r_t = x + h_t$, and both of these vectors have bounded second order moments
as follows from Theorem 11 and Assumption B2, we have that
\[
\lim_{N \to \infty} \left( \frac{1}{C_s} - \frac{1}{\tilde{C}_s} \right)^2 \tilde{\alpha}^2 \frac{1}{N} \| \tilde{r}_t \|^2 \overset{a.s.}{=} 0. \tag{G.9}
\]
Similarly, we have
\[
\| \tilde{r}_t \tilde{\alpha}_t - r_t \alpha_t \|^2 = \| \tilde{r}_t \tilde{\alpha}_t - r_t \alpha_t + r_t \tilde{\alpha}_t - r_t \alpha_t \|^2
\]
\[
= \| (\tilde{r}_t - r_t) \tilde{\alpha}_t + (\tilde{\alpha}_t - \alpha_t) r_t \|^2
\]
\[
\leq \tilde{\alpha}_t^2 \| \tilde{r}_t - r_t \|^2 + (\tilde{\alpha}_t - \alpha_t)^2 \| r_t \|^2 \overset{a.s.}{=} 0, \tag{G.10}
\]
where we used the induction hypothesis and (G.3). Combining all the above results confirms that (G.4) almost surely converges to zero under the induction hypothesis. Since \( f_t(\tilde{S}_{t+1}, y) \) is a linear mapping, which further implies that \( \gamma_t \) is Lipschitz continues as follows from the definition (5.11), the proof of that \( \frac{1}{N} \| s_{t+1} - \tilde{s}_{t+1} \|^2 \overset{a.s.}{=} 0 \) implies \( \frac{1}{N} \| r_{t+1} - \tilde{r}_{t+1} \|^2 \overset{a.s.}{=} 0 \) follows exactly the same steps as above. \( \blacksquare \)
Appendix H

Proof of Lemma 3

Next, we prove Lemma 3. Recall from Section 5.2.3 that we define the vector \( \mathbf{s}_{t+1}(\hat{\alpha}, \tau) \) as

\[
\mathbf{s}_{t+1}(\hat{\alpha}, \tau) = \mathbf{g}_t(\mathbf{r}_t) - (\mathbf{r}_t - \mathbf{s}_\tau).
\] (H.1)

For this vector, next, we aim to simplify the function

\[
E_{t+1}(\hat{\alpha}, \tau) = J_{t+1}^1(\hat{\alpha}, \tau) - J_{t+1}^2(\hat{\alpha}, \tau),
\] (H.2)

where

\[
J_{t+1}^1(\hat{\alpha}, \tau) = \frac{1}{N} \| \mathbf{r}_t - \mathbf{s}_{t+1}(\hat{\alpha}, \tau) \|^2 - v_h,
\]

\[
J_{t+1}^2(\hat{\alpha}, \tau) = \frac{1}{N} \| \mathbf{y} - \mathbf{A}\mathbf{s}_{t+1}(\hat{\alpha}, \tau) \|^2 - \delta v_w.
\]

To increase the readability, in the following we use \( \mathbf{g}_\ell \) to refer to \( \mathbf{g}_\ell(\mathbf{r}_t) \) and drop the dependence of \( \mathbf{s}_{t+1}(\hat{\alpha}, \tau) \) on \( \hat{\alpha} \) and \( \tau \). First, we expand the norm in \( J_{t+1}^2(\hat{\alpha}, \tau) \) to obtain

\[
\| \mathbf{r}_t - \mathbf{s}_{t+1} \|^2 = \| \mathbf{r}_t - \mathbf{g}_\ell + \hat{\alpha}(\mathbf{r}_t - \mathbf{s}_\tau) \|^2
\]

\[
= \| \mathbf{r}_t - \mathbf{g}_\ell \|^2 + 2(\mathbf{r}_t - \mathbf{g}_\ell)^T(\mathbf{r}_t - \mathbf{s}_\tau)\hat{\alpha} + \| \mathbf{r}_t - \mathbf{s}_\tau \|^2\hat{\alpha}^2.
\]

Thus, \( J_{t+1}^1(\hat{\alpha}, \tau) \) is equivalent to

\[
J_{t+1}^1(\hat{\alpha}, \tau) = k_0 + k_1\hat{\alpha} + k_2\hat{\alpha}^2,
\] (H.3)
Proof of Lemma 3

where

\[ k_0 = \frac{1}{N} \| \mathbf{r}_t - \mathbf{g}_t \|^2 - \nu_{ht}, \]  
\[ k_1 = 2 \frac{1}{N} (\mathbf{r}_t - \mathbf{g}_t)^T (\mathbf{r}_t - \mathbf{s}_r), \]  
\[ k_2 = \frac{1}{N} \| \mathbf{r}_t - \mathbf{s}_r \|^2. \]  

(H.4)  

(H.5)  

(H.6)

In the same way, we can show that

\[ \| \mathbf{y} - \mathbf{A}\mathbf{s}_{t+1}(\hat{\alpha}, \tau) \|^2 = \| \mathbf{y} - \mathbf{A}\mathbf{g}_t + \hat{\alpha}\mathbf{A}(\mathbf{r}_t - \mathbf{s}_r) \|^2 \]
\[ = \| \mathbf{y} - \mathbf{A}\mathbf{g}_t \|^2 + 2(\mathbf{y} - \mathbf{A}\mathbf{g}_t)^T \mathbf{A}(\mathbf{r}_t - \mathbf{s}_r)\hat{\alpha} + \| \mathbf{A}(\mathbf{r}_t - \mathbf{s}_r) \|^2 \hat{\alpha}^2, \]  
\[ (H.7) \]

which implies

\[ J_{t+1}^2(\hat{\alpha}, \tau) = d_0 + d_1\hat{\alpha} + d_2\hat{\alpha}^2, \]  
\[ (H.8) \]

where

\[ d_0 = \frac{1}{N} \| \mathbf{y} - \mathbf{A}\mathbf{g}_t \|^2 - \delta\nu_w \]
\[ d_1 = 2 \frac{1}{N} (\mathbf{y} - \mathbf{A}\mathbf{g}_t)^T \mathbf{A}(\mathbf{r}_t - \mathbf{s}_r) \]
\[ d_2 = \frac{1}{N} \| \mathbf{A}(\mathbf{r}_t - \mathbf{s}_r) \|^2. \]

Combining these results, we can show that (H.2) is equivalent to

\[ E_{t+1}(\hat{\alpha}, \tau) = u_0 + u_1\hat{\alpha} + u_2\hat{\alpha}^2, \]  
\[ (H.9) \]

with

\[ u_0 = k_0 - d_0 = \frac{1}{N} \| \mathbf{r}_t - \mathbf{g}_t \|^2 - \frac{1}{N} \| \mathbf{y} - \mathbf{A}\mathbf{g}_t \|^2 - \nu_{ht} + \delta\nu_w \]
\[ u_1 = k_1 - d_1 = 2 \frac{1}{N} (\mathbf{r}_t - \mathbf{s}_r)^T \left( \mathbf{r}_t - \mathbf{g}_t - \mathbf{A}^T (\mathbf{y} - \mathbf{A}\mathbf{g}_t) \right) \]
\[ u_2 = k_2 - d_2 = \frac{1}{N} \| \mathbf{r}_t - \mathbf{s}_r \|^2 - \frac{1}{N} \| \mathbf{A}(\mathbf{r}_t - \mathbf{s}_r) \|^2, \]

which completes the proof.  
\[ \blacksquare \]
Appendix I

Proof of Theorem 13

In the following we will study the interaction of the error vectors

\[ h_t = r_t - x \quad \text{(I.1)} \]
\[ q_t = s_t - x, \quad \text{(I.2)} \]

where \( r_t \) and \( s_t \) are as in (5.5) and (5.6) respectively. Additionally, we will frequently refer to the whole history of these vectors

\[ H_{t+1} = (h_0, h_1, ..., h_t) \quad \text{(I.3)} \]
\[ Q_{t+1} = (q_0, q_1, ..., q_t) \quad \text{(I.4)} \]

and their mapped versions

\[ M_{t+1} = V^T H_{t+1} \quad \text{(I.5)} \]
\[ B_{t+1} = V^T Q_{t+1}, \quad \text{(I.6)} \]

where \( V \) is the right singular vector matrix of \( A = USV^T \). Note that these four error vector matrices can be related as

\[ (M_{r'}, B_r) = V (H_{r'}, Q_r). \]

With this relationship between the error vectors, one can represent the effect of applying the matrix \( V \) through the so-called conditioning technique [9, 10]. For
this, define two vectors

\[ \beta_\tau = Q_\tau^\dagger q_\tau \]  
\[ \rho_\tau = M_\tau^\dagger m_\tau \]  

and the set

\[ G_{\tau,\tau'} = \{ B_{\tau'}, Q_{\tau'}, H_{\tau'}, x, \tilde{w}, S \mid (M_{\tau'}, B_{\tau}) = V(H_{\tau'}, Q_{\tau}) \} , \]  

where \( \tilde{w} = U^T w \). Lastly, for a matrix \( R \), let \( \Phi_R^\perp \) be the set of the left-singular vectors associated with the zero singular values of \( R \). With these definitions, one can obtain the following asymptotic result for \( V \) and \( V^T \).

**Lemma 12.** [58]: Let Assumptions B1-B3 from page 42 hold. Define a vector \( v \in \mathbb{R}^N \) such that \( \lim_{N \to \infty} \frac{1}{\sqrt{N}} \|v\|^2 = \sigma \leq \infty \). Then, for \( \tau = 0, 1, \ldots \) and \( \tau' = 0, 1, \ldots, \tau \) we have

1. The matrix \( V^T \) conditioned on the set \( G_{\tau,\tau} \) almost surely converges to

\[ \lim_{N \to \infty} V^T_{[G_{\tau,\tau}] a.s.} = \lim_{N \to \infty} (M_{\tau, B_{\tau}}) \left( H_{\tau}^\dagger \right) + \Phi_{(M_{\tau}, B_{\tau})}^\perp V \Phi_{(H_{\tau}, Q_{\tau})}^\perp \]  

with \( \tilde{V} \) being Haar distributed and independent of \( G_{\tau,\tau} \). Additionally, we have

\[ p = \Phi_{(M_{\tau}, B_{\tau})}^\perp V \Phi_{(H_{\tau}, Q_{\tau})}^\perp v = \tilde{p} + \beta(v) , \]  

where \( \tilde{p} \sim \mathcal{N}(0, \sigma I) \) is independent of \( G_{\tau,\tau} \) and \( \lim_{N \to \infty} \beta(v) a.s. = 0 \).

2. The matrix \( V \) conditioned on the set \( G_{\tau+1,\tau} \) almost surely converges to

\[ \lim_{N \to \infty} V_{[G_{\tau+1,\tau}] a.s.} = (H_{\tau}, Q_{\tau+1}) \left( M_{\tau}^\dagger \right) + \Phi_{(H_{\tau}, Q_{\tau+1})}^\perp V \Phi_{(M_{\tau}, B_{\tau+1})}^\perp \]  

with \( \tilde{V} \) being Haar distributed and independent of \( G_{\tau+1,\tau} \). Additionally, we have

\[ p = \Phi_{(H_{\tau}, Q_{\tau+1})}^\perp V \Phi_{(M_{\tau}, B_{\tau+1})}^\perp v = \tilde{p} + \beta(v) , \]  

where \( \tilde{p} \sim \mathcal{N}(0, \sigma I) \) is independent of \( G_{\tau+1,\tau} \) and \( \lim_{N \to \infty} \beta(v) a.s. = 0 \).

With this lemma and Theorem 11, we aim to study the behaviour of the following
Proof of Theorem 13

parametrized denoising step and its error
\[ s_{t+1}(\hat{\alpha}, \tau') = g_\tau(r_\tau) - \hat{\alpha}(r_\tau - s_\tau') \]  
\[ q_{t+1}(\hat{\alpha}, \tau') = s_{t+1}(\hat{\alpha}, \tau') - x. \]  

(I.14)  
(I.15)

In particular, we are interested in the roots of the function
\[ E_{t+1}(\hat{\alpha}, \tau') = \bar{J}_{t+1}(\hat{\alpha}, \tau') - \bar{J}_{t+1}(\hat{\alpha}, \tau'), \]  

(I.16)

where
\[ \bar{J}_{t+1}(\hat{\alpha}, \tau') = \frac{1}{N} \left( \|s_{t+1}(\hat{\alpha}, \tau') - r_\tau\|^2 - \|h_\tau\|^2 \right) \]  
\[ \bar{J}_{t+1}(\hat{\alpha}, \tau') = \frac{1}{N}\|y - A\bar{s}_{t+1}(\hat{\alpha}, \tau')\|^2 - \delta v_w. \]  

(I.17)  
(I.18)

Before beginning the analysis, we define two vectors that will arise in the derivation
\[ \bar{\beta}_{t+1}(\hat{\alpha}, \tau') = \frac{1}{N} Q_{t+1}^\dagger q_{t+1}(\hat{\alpha}, \tau') \]  
\[ \bar{\nu}_{t+1}(\hat{\alpha}, \tau') = \frac{1}{N} H_{t+1}^\dagger q_{t+1}(\hat{\alpha}, \tau'). \]  

(I.19)  
(I.20)

From Theorem 11 we know that the matrices $Q_{t+1}$ and $H_{t+1}$ are asymptotically full rank, so the pseudo-inverses above are well-defined in the limit $N \to \infty$. Additionally, we have that
\[ \lim_{N \to \infty} \frac{1}{N} \|q_{t+1}(\hat{\alpha}, \tau')\|^2 = \lim_{N \to \infty} \frac{1}{N} \|s_{t+1}(\hat{\alpha}, \tau') - x\|^2 \]
\[ = \lim_{N \to \infty} \frac{1}{N} \|g_\tau(r_\tau) - \hat{\alpha}(r_\tau - s_\tau') - x\|^2 \]
\[ = \lim_{N \to \infty} \frac{1}{N} \|g_\tau(x + h_\tau) - \hat{\alpha}(h_\tau - q_\tau') - x\|^2 \]
\[ \leq \lim_{N \to \infty} \frac{1}{N} \|g_\tau(x + h_\tau)\|^2 - \hat{\alpha}^2 \left( \frac{1}{N} \|h_\tau\|^2 - \frac{1}{N} \|q_\tau'\|^2 \right) - \frac{1}{N} \|x\|^2, \]

(I.21)

where the bound comes from Theorem 11 stating that in the limit $h_\tau$ and $q_\tau'$ have finite variances, and Assumption B2 in combination with (5.17). Thus, in the limit, the vectors (I.19) and (I.20) are almost surely finite.

In the following we simplify the notations and drop the dependence of $s_{t+1}(\hat{\alpha}, \tau'), q_{t+1}(\hat{\alpha}, \tau'), \bar{\beta}_{t+1}(\hat{\alpha}, \tau')$ and $\bar{\nu}_{t+1}(\hat{\alpha}, \tau')$ on $\hat{\alpha}$ and $\tau'$.  

183
I.1 Analysis of (I.17)

First, we consider the function $\bar{J}^1_{\tau+1}$ and its limiting behaviour. Since $\mathbf{s}_{t+1} = \mathbf{x} + \mathbf{q}_{t+1}$ and $\mathbf{r}_\tau = \mathbf{x} + \mathbf{h}_\tau$, we can rewrite (I.17) as

$$\bar{J}^1_{\tau+1}(\hat{\alpha}, \tau') = \frac{1}{N} \left( \|\mathbf{q}_{t+1} - \mathbf{h}_\tau\|^2 - \|\mathbf{h}_\tau\|^2 \right)$$

$$= \frac{1}{N} \left( \|\mathbf{q}_{t+1}\|^2 - 2\frac{1}{N} \mathbf{h}_\tau^T \mathbf{q}_{t+1} \right)$$

$$= \frac{1}{N} \|\mathbf{q}_{t+1}\|^2 + e^1_{\tau+1}. \quad \text{(I.22)}$$

Next we consider $e^1_{\tau+1}(\hat{\alpha})$ from the last result in the LSL. First, we can follow the same steps as in the proof of Theorem 12 to obtain

$$\lim_{N \to \infty} \frac{1}{N} \mathbf{h}_T \mathbf{g}_t(\mathbf{r}_t) \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} \mathbf{h}_T \mathbf{g}_t(\mathbf{x} + \mathbf{h}_\tau) \overset{a.s.}{=} \lim_{N \to \infty} v_{h_t} \alpha_t. \quad \text{(I.23)}$$

where $v_{h_t} = \frac{1}{N} \|\mathbf{h}_t\|^2$. Additionally, following the same steps and using (5.15) yields

$$\lim_{N \to \infty} \frac{1}{N} \mathbf{h}_T(\mathbf{r}_t - \mathbf{s}_\nu) = \lim_{N \to \infty} \frac{1}{N} \mathbf{h}_T(\mathbf{h}_t - \mathbf{q}_\nu)$$

$$\overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} \mathbf{h}_T \mathbf{h}_t \overset{a.s.}{=} \lim_{N \to \infty} v_{h_t}. \quad \text{(I.24)}$$

Combining (I.23) and (I.24) implies

$$\lim_{N \to \infty} e^1_{l+1}(\hat{\alpha}, \tau') = - \lim_{N \to \infty} 2 \frac{1}{N} \mathbf{h}_T(\mathbf{g}_t(\mathbf{r}_t) - \hat{\alpha}(\mathbf{r}_t - \mathbf{s}_\nu) - \mathbf{x})$$

$$\overset{a.s.}{=} \lim_{N \to \infty} -2v_{h_t}(\alpha_t - \hat{\alpha}). \quad \text{(I.25)}$$

Thus, the error term $e^1_{l+1}(\hat{\alpha})$ converges to a linear function of $(\alpha_t - \hat{\alpha})$ in the limit $N \to \infty$. We will return to this result shortly.

I.2 Analysis of (I.18)

Next, we analyze $\bar{J}^2_{\tau+1}(\hat{\alpha}, \tau')$ which involves the following norm

$$\frac{1}{N} \|\mathbf{y} - \mathbf{A}\mathbf{s}_{t+1}\|^2 = \frac{1}{N} \|\mathbf{w} - \mathbf{A}\mathbf{q}_{t+1}\|^2$$

$$= \frac{1}{N} \|\mathbf{w}\|^2 + \frac{1}{N} \|\mathbf{A}\mathbf{q}_{t+1}\|^2 - 2\frac{1}{N} \mathbf{w}^T \mathbf{A}\mathbf{q}_{t+1}, \quad \text{(I.26)}$$

184
where we used the fact that \( y = Ax + w \). Next, let Assumptions B1-B3 from page 42 hold so we could use (I.10) to obtain

\[
\lim_{N \to \infty} V^T_{[G_{t+1},t+1]} \mathbf{q}_{t+1} = \lim_{N \to \infty} (M_{t+1}, B_{t+1}) \left( H^\dagger_{t+1} \mathbf{q}_{t+1} + \Phi^\dagger_{(M_{t+1}, B_{t+1})} \tilde{V} \Phi^\dagger_{(H_{t+1}, Q_{t+1})} \mathbf{q}_{t+1} \right)
\]

\( \overset{a.s.}{=} \lim_{N \to \infty} M_{t+1} \mathbf{p}_{t+1} + B_{t+1} \mathbf{b}_{t+1} + p_{t+1}, \) \hspace{1cm} (I.27)

where we used (I.20) and (I.19), and defined

\[
\mathbf{p}_{t+1} = \Phi^\dagger_{(M_{t+1}, B_{t+1})} \tilde{V} \Phi^\dagger_{(H_{t+1}, Q_{t+1})} \mathbf{q}_{t+1}, \hspace{1cm} (I.30)
\]

Note that based on Lemma 12, \( p_{t+1} \) asymptotically acts as a zero-mean i.i.d. Gaussian vector independent of \( M_{t+1}, B_{t+1}, U, S \) and \( w \). With this result we can obtain the following

\[
\lim_{N \to \infty} \frac{1}{N} \|A \mathbf{q}_{t+1}\|^2 = \lim_{N \to \infty} \frac{1}{N} \|SV^T \mathbf{q}_{t+1}\|^2 \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} \|S(B_{t+1} \mathbf{b}_{t+1} + M_{t+1} \mathbf{v}_{t+1} + p_{t+1})\|^2 \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} \left( \|S \tilde{B}_{t+1} \mathbf{b}_{t+1}\|^2 + \|Sp_{t+1}\|^2 \right)
\]

\[
+ \|SM_{t+1} \mathbf{v}_{t+1}\|^2 \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} \left( \| \tilde{B}_{t+1} \mathbf{b}_{t+1}\|^2 + \|p_{t+1}\|^2 \right)
\]

\[
+ \|SM_{t+1} \mathbf{v}_{t+1}\|^2 \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} \left( \| \tilde{B}_{t+1} \mathbf{b}_{t+1}\|^2 + \|p_{t+1}\|^2 \right)
\]

where in (I.31) we used the asymptotic model of the error vectors \( b_{t} \) from Theorem 11, (I.32) follows from the fact that \( p_{t+1} \) is asymptotically independent of \( \tilde{B}_{t+1} \) and \( M_{t+1} \) and in (I.33) we used the normalization \( \frac{1}{N}Tr\{S^T S\} = 1 \), Lemma 6 and
\[
\begin{align*}
\lim_{N \to \infty} \frac{1}{N} \left\| S \tilde{B}_{t+1} \beta_{t+1} \right\|^2 & \overset{\text{a.s.}}{=} \lim_{N \to \infty} \frac{1}{N} \left\| \tilde{B}_{t+1} \beta_{t+1} \right\|^2 \\
\lim_{N \to \infty} \frac{1}{N} \left\| S p_{t+1} \right\|^2 & \overset{\text{a.s.}}{=} \lim_{N \to \infty} \frac{1}{N} \left\| p_{t+1} \right\|^2.
\end{align*}
\]

To relate (I.33) to the MSE of \( q_{t+1} \), next we consider the latter. Again, by referring to (I.27) and following the same steps as just above, we can obtain
\[
\begin{align*}
\lim_{N \to \infty} \frac{1}{N} \left\| q_{t+1} \right\|^2 & \overset{\text{a.s.}}{=} \lim_{N \to \infty} \frac{1}{N} \left\| B_{t+1} \beta_{t+1} + M_{t+1} \nu_{t+1} + p_{t+1} \right\|^2 \\
& \overset{\text{a.s.}}{=} \lim_{N \to \infty} \frac{1}{N} \left\| \tilde{B}_{t+1} \beta_{t+1} + M_{t+1} \nu_{t+1} + p_{t+1} \right\|^2 \\
& \overset{\text{a.s.}}{=} \lim_{N \to \infty} \frac{1}{N} \left\| \tilde{B}_{t+1} \beta_{t+1} \right\|^2 + \frac{1}{N} \left\| M_{t+1} \nu_{t+1} \right\|^2 + \frac{1}{N} \left\| p_{t+1} \right\|^2,
\end{align*}
\]
where we used the asymptotic independence of \( b_r \) and \( m_{r'} \), which follows from (5.15). By comparing the last result to (I.33), we find that
\[
\begin{align*}
\lim_{N \to \infty} \frac{1}{N} \left\| A q_{t+1} \right\|^2 & \overset{\text{a.s.}}{=} \lim_{N \to \infty} \frac{1}{N} \left\| q_{t+1} \right\|^2 + \frac{1}{N} \left\| S M_{t+1} \nu_{t+1} \right\|^2 - \frac{2}{N} \beta_{t+1}^T \tilde{B}_{t+1}^T S^T S M_{t+1} \nu_{t+1}.
\end{align*}
\]

Next, define a function
\[
\begin{align*}
e^2_{t+1}(\hat{\alpha}) &= \frac{1}{N} \left\| S M_{t+1} \nu_{t+1} \right\|^2 - \frac{1}{N} \left\| M_{t+1} \nu_{t+1} \right\|^2 \\
& + \frac{2}{N} \beta_{t+1}^T \tilde{B}_{t+1}^T S^T S M_{t+1} \nu_{t+1} - 2 \frac{1}{N} w^T A \tilde{q}_{t+1}.
\end{align*}
\]

Then, we can use (I.37), (I.36) and (I.26) to show that the function \( \tilde{J}_{t+1}^2 \) from (I.18) almost surely converges to
\[
\begin{align*}
\lim_{N \to \infty} \tilde{J}_{t+1}^2 & \overset{\text{a.s.}}{=} \lim_{N \to \infty} \frac{1}{N} \left\| \tilde{q}_{t+1} \right\|^2 + e^2_{t+1}(\hat{\alpha}).
\end{align*}
\]

Next, we analyze the behaviour of the error \( e^2_{t+1}(\hat{\alpha}) \). First, we consider the term \( \frac{1}{N} w^T A \tilde{q}_{t+1} \). Using the SVD of \( A = USV^T \), the model (I.27) and defining
\[ \text{Proof of Theorem 13} \]

\[ \tilde{w} = U^T w, \]

we can obtain

\[ \lim_{N \to \infty} \frac{1}{N} w^T A \tilde{q}_{t+1} \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} \tilde{w}^T S V^T \tilde{q}_{t+1} \]

\[ = \lim_{N \to \infty} \frac{1}{N} \tilde{w}^T S (M_{t+1} \bar{\nu}_{t+1} + B_{t+1} \bar{\beta}_{t+1} + p_{t+1}) \]

\[ \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} w^T S M_{t+1} \nu_{t+1}, \tag{I.39} \]

where we used the asymptotic independence of \( \tilde{w} \) and \( p_{t+1} \) and of \( \tilde{w} \) and \( S b_{\tau} \) as follows from Theorem 12 and (5.16) respectively. Next, we analyze the vector \( \nu_{t+1} \) in the large system limit

\[ \lim_{N \to \infty} \nu_{t+1} = H_+^T q_{t+1}, \tag{I.40} \]

Here, the vector \( \frac{1}{N} H_+^T q_{t+1} \) is composed of the elements \( \frac{1}{N} h_k^T q_{t+1} \). Following the same steps as in (I.25), we can show that this element almost surely converges to

\[ \lim_{N \to \infty} \frac{1}{N} h_k^T q_{t+1} \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} h_k^T h_t (\alpha_t - \hat{\alpha}), \tag{I.41} \]

which together with (I.40) implies that

\[ \lim_{N \to \infty} \nu_{t+1} \overset{a.s.}{=} e_{t+1} (\alpha_t - \hat{\alpha}), \tag{I.42} \]

where \( e_{t+1} \in \mathbb{R}^{t+1} \) is \((t + 1)\)th vector of the \( t + 1 \) dimensional natural basis. Since \( M_{t+1} e_{t+1} = m_t \), substituting this result into (I.39) leads to

\[ \lim_{N \to \infty} \frac{1}{N} w^T A \tilde{q}_{t+1} \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} \tilde{w}^T S m_t (\alpha_t - \hat{\alpha}) \]

\[ \overset{a.s.}{=} v_w (\alpha_t - \hat{\alpha}), \tag{I.43} \]

where we used the asymptotic result \( \lim_{N \to \infty} \frac{1}{N} \tilde{w}^T S m_t \overset{a.s.}{=} v_w \), which can be proved in a similar way as was proved the statement (F.4) in Appendix F. In particular, we consider a general long-memory SMP algorithm (which includes a short-memory MP algorithm as a special case) with the updates (5.5)-(5.6) for which the error vector \( h_t \) from (I.1) corresponds to

\[ h_t = \frac{1}{C_r} \left( A^T f_t (S_{t+1}, y) - \sum_{\tau=0}^{t} \gamma_{\tau} q_{\tau} \right), \tag{I.44} \]
Proof of Theorem 13

where \( C_t^r = -\sum_{\tau=0}^{t} \gamma_t^r \). Note that in all the SMP algorithms analyzed, mentioned or proposed in this thesis, the function \( f_t(\{s_{\tau}\}_r, y) \) depends on \( \{s_{\tau}\}_r \) and \( y \) through the set \( \{z_{\tau}\}_r^t \) where \( z_{\tau} = y - A s_{\tau} = w - A q_{\tau} \) \([9, 10, 14, 60, 79, 92, 93, 94]\). Therefore, for these SMP, (I.45) can be written as

\[
h_t = \frac{1}{C_t^r} \left( A^T f_t(\{z_{\tau}\}_r^t) - \sum_{\tau=0}^{t} \gamma_t^r q_{\tau} \right). \tag{I.45}
\]

Next, based on the definition of the vector \( m_t = V^T h_t \), the inner-product \( \frac{1}{N} \tilde{w}^T S m_t \) is equivalent to

\[
\lim_{N \to \infty} \frac{1}{N} \tilde{w}^T S m_t = \lim_{N \to \infty} \frac{1}{N} w A h_t
\]

\[
= \lim_{N \to \infty} \frac{1}{C_t^r} \left( \frac{1}{N} w A A^T f_t(\{z_{\tau}\}_r^t) - \sum_{\tau=0}^{t} \gamma_t^r \frac{1}{N} w A q_{\tau} \right)
\]

\[
= a.s. \lim_{N \to \infty} \frac{1}{C_t^r} \frac{1}{N} w A A^T f_t(\{z_{\tau}\}_r^t) \tag{I.46}
\]

\[
= \lim_{N \to \infty} \frac{1}{C_t^r} \frac{1}{N} \text{Tr}\left\{ w (\{z_{\tau}\}_r^t)^T A A^T \right\}
\]

\[
= a.s. \lim_{N \to \infty} \frac{1}{C_t^r} \frac{1}{N} \text{Tr}\left\{ \sum_{\tau=0}^{t} E[wz_{\tau}^T] J_{z_{\tau}} (f_t(\{z_{\tau}\}_r^t)) A A^T \right\} \tag{I.47}
\]

\[
= a.s. \lim_{N \to \infty} \frac{1}{C_t^r} \frac{1}{N} \text{Tr}\left\{ \sum_{\tau=0}^{t} E[wz_{\tau}^T] J_{z_{\tau}} (f_t(\{z_{\tau}\}_r^t)) A A^T \right\}
\]

\[
= \lim_{N \to \infty} v_w \frac{1}{C_t^r} \frac{1}{N} \text{Tr}\left\{ \sum_{\tau=0}^{t} J_{z_{\tau}} (f_t(\{z_{\tau}\}_r^t)) A A^T \right\} \tag{I.49}
\]

\[
= \lim_{N \to \infty} -v_w \frac{1}{C_t^r} \frac{1}{N} \text{Tr}\left\{ \sum_{\tau=0}^{t} J_{q_{\tau}} (A^T f_t(\{z_{\tau}\}_r^t)) \right\} \tag{I.50}
\]

\[
= \lim_{N \to \infty} -v_w \frac{1}{C_t^r} \sum_{\tau=0}^{t} \gamma_t^r \tag{I.51}
\]

\[
= v_w, \tag{I.52}
\]

where (I.46) follows from the asymptotic orthogonality of \( w \) and \( A q_{\tau} \) from Theorem 11; (I.47) due to Lemma 6; (I.48) motivated by the fact that \( z_{\tau} = w - A q_{\tau} \); (I.50) follows from \(-J_{z_{\tau}} (g_{A}^t(z_{\tau}, \hat{\nu}_{B \rightarrow A}^t)) A = J_{x_{\hat{\nu}_{B \rightarrow A}}^t} (g_{A}^t(z_{\tau}, \hat{\nu}_{B \rightarrow A}^t)) \) and the linearity of the operator \( J_{x_{\hat{\nu}_{B \rightarrow A}}^t} \); (I.51) is due to the definition of \( \gamma_t^r \) from (5.11) and the last
Proof of Theorem 13

Following similar steps as before, we can show that

$$
\lim_{N \to \infty} \frac{1}{N} \|SM_{t+1} \nu_{t+1}\|^2 - \frac{1}{N} \|M_{t+1} \nu_{t+1}\|^2 + \frac{2}{N} \beta_{t+1}^T B_{t+1}^T \nu_{t+1} SM_{t+1} \nu_{t+1}
$$

\begin{equation}
= \lim_{N \to \infty} \frac{1}{N} \|Sm_t\|^2 (\alpha_t - \hat{\alpha})^2 - \frac{1}{N} \|m_t\|^2 (\alpha_t - \hat{\alpha})^2 + \frac{2}{N} \beta_{t+1}^T B_{t+1}^T S^T Sm_t (\alpha_t - \hat{\alpha}).
\end{equation}

(I.53)

By combining this result with (I.43) we conclude that the error $e_2$ from (I.37) almost surely converges to

$$
\lim_{N \to \infty} e_2^2(t+1) (\hat{\alpha}) \approx \lim_{N \to \infty} \left( \frac{1}{N} \|Sm_t\|^2 - v_{ht}\right) (\alpha_t - \hat{\alpha})^2
$$

$$
+ 2 \left( \frac{1}{N} \beta_{t+1}^T B_{t+1}^T S^T Sm_t - v_{ww}\right) (\alpha_t - \hat{\alpha}).
$$

(I.54)

I.3 Roots of (I.16)

Combining (I.22) with (I.25) and (I.38) with (I.54), we can obtain the following asymptotic result for $E_{t+1}(\hat{\alpha}, \tau')$ from (I.16) under Assumptions B1-B3 from page 42

$$
\lim_{N \to \infty} E_{t+1}(\hat{\alpha}, \tau') \approx \lim_{N \to \infty} e_1^2(t+1) (\hat{\alpha}) - e_2^2(t+1) (\hat{\alpha})
$$

$$
= \lim_{N \to \infty} \left( v_{ht} - \frac{1}{N} \|Sm_t\|^2\right) (\alpha_t - \hat{\alpha})^2
$$

$$
- 2 \left( v_{ht} + \frac{1}{N} \beta_{t+1}^T B_{t+1}^T S^T Sm_t - v_{ww}\right) (\alpha_t - \hat{\alpha}).
$$

(I.55)

From this result, we immediately notice that the first root $\hat{\alpha}_1$ to $E_{t+1}(\hat{\alpha}, \tau')$ almost surely converges to $\lim_{N \to \infty} \hat{\alpha}_1 = \alpha_t$. Next, we aim to obtain the closed-form solution for the second root $\hat{\alpha}_2$ to this function. Consider the asymptotic behaviour
Proof of Theorem 13

\[ \lim_{N \to \infty} \mathcal{B}_{t+1}(\hat{\alpha}, \tau') = \lim_{N \to \infty} \mathcal{Q}_{t+1}^\dagger \mathcal{B}_{t+1}(\hat{\alpha}, \tau') \]
\[ = \lim_{N \to \infty} \mathcal{Q}_{t+1}^\dagger (\mathcal{S}_{t+1}(\hat{\alpha}, \tau') - x) \]
\[ = \lim_{N \to \infty} \mathcal{Q}_{t+1}^\dagger (\mathcal{G}_t(r_t) - \hat{\alpha}(r_t - s_{\tau'}) - x) \]
\[ = \lim_{N \to \infty} \mathcal{Q}_{t+1}^\dagger (g_t(r_t) - \hat{\alpha}(h_t - q_{\tau'}) - x) \]
\[ \overset{a.s.}{=} \lim_{N \to \infty} \mathcal{Q}_{t+1}^\dagger (g_t(r_t) + \hat{\alpha}q_{\tau'} - x), \quad (I.56) \]
where in we used (I.4) and (5.15) to obtain
\[ \lim_{N \to \infty} \mathcal{Q}_{t+1}^\dagger h_t = \lim_{N \to \infty} \left( \frac{1}{N} \mathcal{Q}_{t+1}^\dagger \mathcal{Q}_{t+1} \right)^{-1} \frac{1}{N} \mathcal{Q}_{t+1} h_t \overset{a.s.}{=} 0. \]

Additionally, note that
\[ \mathcal{Q}_{t+1}^\dagger q_{\tau'} = e_{\tau' + 1}, \quad (I.57) \]
where \( e_i \in \mathbb{R}^{t+1} \) is the \( i \)-th vector of the natural basis. Therefore (I.56) is equivalent to
\[ \mathcal{B}_{t+1}(\hat{\alpha}, \tau') = \lim_{N \to \infty} \mathcal{Q}_{t+1}^\dagger (g_t(r_t) - \hat{\alpha}e_{\tau' + 1}) \]
\[ = \beta_{t+1} + \hat{\alpha}e_{\tau' + 1}, \quad (I.58) \]
where we defined
\[ \beta_{t+1} = \lim_{N \to \infty} \mathcal{Q}_{t+1}^\dagger (g_t(r_t) - x). \quad (I.59) \]
By applying (I.58) to and grouping terms together in (I.55), we can rewrite the function \( E_{t+1}(\hat{\alpha}, \tau') \) as
\[ \lim_{N \to \infty} E_{t+1}(\hat{\alpha}, \tau') \overset{a.s.}{=} \lim_{N \to \infty} (c_0^t + \hat{\alpha}c_1^t(\tau'))(\alpha_t - \hat{\alpha}) + c_2^t(\alpha_t - \hat{\alpha})^2 \]
\[ = \lim_{N \to \infty} \left( c_0^t + \hat{\alpha}c_1^t(\tau') + c_2^t(\alpha_t - \hat{\alpha}) \right)(\alpha_t - \hat{\alpha}), \quad (I.60) \]
where
\[
c_0^t = -2(v_h - v_w + \frac{1}{N}\beta_{t+1}^T B_{t+1}^T S_t^T S_m)
\]  
(I.61)
\[
c_1^t(\tau') = -2\frac{1}{N}b_{\tau}^T S_t^T S_m
\]  
(I.62)
\[
c_2^t = v_h - \frac{1}{N}\|S_m\|^2.
\]  
(I.63)

Thus, the second root \(\hat{\alpha}_2\) of this function follows
\[
\lim_{N \to \infty} c_0^t + \hat{\alpha}_2 c_1^t(\tau') + c_2^t(\alpha_t - \hat{\alpha}_2) = 0.
\]  
(I.64)

Solving for \(\hat{\alpha}_2\) gives
\[
\lim_{N \to \infty} \hat{\alpha}_2(\tau') \overset{a.s.}{=} \lim_{N \to \infty} \frac{c_0^t + c_2^t\alpha_t}{c_2^t - c_1^t(\tau')}.
\]  
(I.65)

which completes the proof. ■
Appendix J

Proof of the asymptotic equivalence of (5.69) and (5.70)

Next, we show that for a pair of indices $\tau \neq \tau'$, $(\tau, \tau') \leq t$, the condition

$$\lim_{N \to \infty} \frac{1}{N} q_{\tau}^T A^T A_{\tau} \neq \lim_{N \to \infty} \frac{1}{N} q_{\tau'}^T A^T A_{\tau}$$

is asymptotically equivalent to

$$\lim_{N \to \infty} \frac{1}{N} z_{\tau}^T (y - A r_{t}) \neq \lim_{N \to \infty} \frac{1}{N} z_{\tau'}^T (y - A r_{t}).$$

Here we have $y = A x + w$, $z_{\tau} = y - A s_{t}$ and the error vectors are

$$q_{t} = s_{t} - x$$

$$h_{t} = r_{t} - x,$$

where $r_{t}$ and $s_{t}$ are from (5.5)-(5.6). In the following we let Assumptions B1-B3 from page 42 hold. Then, first, note that from the definitions of $z_{t}$, $y$ and $q_{t}$, we have $z_{t} = w - A q_{t}$. Then, we can show that the left hand side of (J.1) is
Proof of the asymptotic equivalence of (5.69) and (5.70)

equivalent to

\[
\lim_{N \to \infty} \frac{1}{N} q_t^T A^T A_\tau = \lim_{N \to \infty} -\frac{1}{N} (w - A_q \tau)^T A (r_t - \xi) + \frac{1}{N} w^T A_\tau
\]

\[\overset{\text{a.s.}}{=} \lim_{N \to \infty} -\frac{1}{N} z_t^T A (r_t - \xi) + v_w \quad (J.5)\]

\[= \lim_{N \to \infty} -\frac{1}{N} z_t^T (A r_t - A \xi + w - w) + v_w \]

\[= \lim_{N \to \infty} -\frac{1}{N} z_t^T (A r_t - y) + v_w - \frac{1}{N} z_t^T w \]

\[\overset{\text{a.s.}}{=} \lim_{N \to \infty} -\frac{1}{N} z_t^T (y - A r_t) + v_w - \delta v_w, \quad (J.6)\]

where (J.5) uses the result \(\lim_{N \to \infty} \frac{1}{N} w^T A_\tau \overset{\text{a.s.}}{=} v_w\) from (I.52), while the last step is based on

\[
\lim_{N \to \infty} \frac{1}{N} w^T z_t = \lim_{N \to \infty} \frac{1}{N} w^T (w - A_q \tau)
\]

\[\overset{\text{a.s.}}{=} \lim_{N \to \infty} \frac{1}{N} w^T w \overset{\text{a.s.}}{=} \delta v_w, \quad (J.7)\]

where we used the asymptotic orthogonality of \(w\) and \(A_q \tau\) from (5.16) and Lemma 5. Finally, applying (J.6) to both sides of (J.1) gives (J.2).
Appendix K

Proof of Theorem 14

The proof of Theorem 14 for MF-OAMP and VAMP is based on the asymptotic result formulated in the following theorem.

**Theorem 15.** Consider an SMP algorithm (5.5)-(5.6) with \( f_t(S_{t+1}, y) \) having a well-defined Jacobian with respect to every input vector and \( \alpha_t \) be the divergence of the denoiser \( g_t \) at \( r_t \). Define a corrected denoiser

\[
\tilde{g}_{t+1}(r_t) = g_t(r_t) - \alpha_t r_t \tag{K.1}
\]

and its error

\[
q_{t+1} = \tilde{g}_{t+1}(r_t) - x.
\]

Then, under Assumptions B1-B3 from page 42, the derivative (5.71) at \( \hat{\alpha} = \alpha_t \) almost surely converges to

\[
\lim_{N \to \infty} u_1 + 2u_2 \alpha_t = \lim_{N \to \infty} 2\left(v_{h_t} + \frac{1}{N} h_t^T A^T A q_{t+1} - v_w \right). \tag{K.2}
\]

**Proof.** In the following, let Assumptions B1-B3 hold and \( \alpha_t \) be the divergence of \( g_t(r_t) \). Additionally, let \( v_x = \lim_{N \to \infty} \frac{1}{N} \|x\| \) and \( v_{h_t} = \lim_{N \to \infty} \frac{1}{N} \|h_t\|^2 \). Then, we can use the definitions of \( u_1 \) and \( u_2 \) from Lemma 3 with \( \tau = 0 \) to show that the derivative (5.71) at \( \hat{\alpha} = \alpha_t \) is equal to

\[
u_1 + 2u_2 \alpha_t = \frac{2}{N} \left( \|r_t\|^2 - r_t^T (g_t(r_t) - \alpha_t r_t) - r_t^T A^T y + r_t^T A^T A(g_t(r_t) - \alpha_t r_t) \right). \tag{K.3}
\]
Next, we use (K.1) to define a corrected vector \( s_{t+1} = g_{t+1}(r_t) \) and rewrite (K.3) as

\[
u_1 + 2u_2 \alpha_t = \frac{2}{N} \left( \| r_t \|^2 - r_t^T s_{t+1} - r_t^T A^T (y - As_{t+1}) \right)
= \frac{2}{N} \left( \| r_t \|^2 - r_t^T \left( s_{t+1} + A^T (y - As_{t+1}) \right) \right).
\]

(K.4)

Note that we have \( r_t = x + h_t \) where \( x \) and \( h_t \) are asymptotically orthogonal as follows from the definition \( x = -q_0 \) and (5.15). Then, we can obtain

\[
\lim_{N \to \infty} \frac{1}{N} \| r_t \|^2 \overset{a.s.}{=} v_x + v_h.
\]

(K.5)

Next, we consider the term

\[
r_t^T \left( s_{t+1} + A^T (y - As_{t+1}) \right) = x^T \left( s_{t+1} + A^T (y - As_{t+1}) \right) + h_t^T \left( s_{t+1} + A^T (y - As_{t+1}) \right).
\]

(K.6)

First, using the definition of \( y = Ax + w \) and of \( s_{t+1} = x + q_{t+1} \), we can obtain \( y - As_{t+1} = w - Aq_{t+1} \). Recall from Section 5.2.1 that \( q_{t+1} \) is an error vector that follows the properties of Theorem 11, which implies

\[
\lim_{N \to \infty} \frac{1}{N} h_t^T s_{t+1} \overset{a.s.}{=} 0,
\]

(K.7)

where we used (5.15) to show that \( \lim_{N \to \infty} \frac{1}{N} h_t^T q_{t+1} \overset{a.s.}{=} 0 \). Thus, in the limit, the second component in (K.6) is equivalent to

\[
\lim_{N \to \infty} \frac{1}{N} h_t^T \left( s_{t+1} + A^T (w - Aq_{t+1}) \right) \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} h_t^T A^T (y - As_{t+1})
= \lim_{N \to \infty} \frac{1}{N} h_t^T A^T w - \frac{1}{N} h_t^T A^T Aq_{t+1}
\overset{a.s.}{=} v_w - \lim_{N \to \infty} \frac{1}{N} h_t^T A^T Aq_{t+1},
\]

(K.8)

where we used (1.52) to obtain the last step. On the other hand, the first component in (K.6) is almost surely equal to

\[
\lim_{N \to \infty} \frac{1}{N} x^T \left( x + q_{t+1} + A^T (y - As_{t+1}) \right) \overset{a.s.}{=} v_x - \lim_{N \to \infty} \frac{1}{N} \left( q_0^T q_{t+1} + q_0^T A^T w - q_0^T A^T Aq_{t+1} \right)
\overset{a.s.}{=} v_x,
\]

where we used the independence of \( w \) and \( Aq_{t+1} \) from (5.15) and (4.15) to show...
Proof of Theorem 14

that \( \lim_{N \to \infty} \frac{1}{N}(q_0^T q_{t+1} - q_0^T A^T A q_{t+1}) \) is 0. Substituting all the above results into (K.4) gives us (K.2).

While the above theorem can be shown to hold for every SMP algorithm mentioned, discussed or derived in this thesis, in the following we continue the analysis of \( u_1 + 2u_2 \) for two special cases: MF-OAMP [79] and VAMP. In both of these examples, the error vector \( h_t = r_t - x \) takes the form of

\[
h_t = q_t - \gamma_t^{-1} A^T W_t^{-1} z_t
\]

with

\[
\lim_{N \to \infty} \gamma_t = \lim_{N \to \infty} \frac{1}{N}q_t^T A T W_t z_t
\]

as follows from Lemma 1. Recall that for VAMP we have \( W_t = v_t I + v_t q_t A A^T \), while for MF-OAMP it is \( W_t = I \).

Next, note that we can use (4.15) and (K.9) to show that the component \( \frac{1}{N} h_t^T A^T A q_{t+1} \) in (K.2) is asymptotically equivalent to

\[
\lim_{N \to \infty} \frac{1}{N} q_{t+1}^T A^T A h_t = \lim_{N \to \infty} \frac{1}{N} q_{t+1}^T A^T A (q_t - \gamma_t^{-1} A^T W_t^{-1} z_t)
\]

\[
= \lim_{N \to \infty} \frac{1}{N} q_{t+1}^T A q_t - \gamma_t^{-1} \frac{1}{N} q_{t+1}^T A A^T W_t z_t
\]

\[
= \bar{\psi}_t - \lim_{N \to \infty} \gamma_t^{-1} \frac{1}{N} q_{t+1}^T A A^T W_t z_t,
\]

where in the last step we defined \( \bar{\psi}_t = \lim_{N \to \infty} \frac{1}{N} q_{t+1}^T q_t \). In the following, we analyze (K.2) in combination with (K.11) for the two SMP algorithms, VAMP and MF-OAMP, separately.
K.1 MF-OAMP

In MF-OAMP we have $W_t = I$ and by using the SVD of $A = USV^T$, we can show that the inner-product (K.11) is equivalent to

$$
\lim_{N \to \infty} \frac{1}{N} q_{t+1}^T A^T A A^T W_t^{-1} z_t = \lim_{N \to \infty} \frac{1}{N} q_{t+1}^T A^T A A^T (w - A q_t)
$$

\[= a.s. \lim_{N \to \infty} -\bar{\psi}_t \frac{1}{N} \text{Tr}\{A^2\} \tag{K.12}\]

\[= -\bar{\psi}_t \chi_2, \tag{K.13}\]

where $\chi_j$ is as in (4.49). In (K.12) we used

$$
\lim_{N \to \infty} \frac{1}{N} q_{t+1}^T A^T A A^T (w - A q_t) a.s. \to -\bar{\psi}_t \chi_2; \tag{K.14}\]

where in the first step we used the asymptotic independence of $w$ and $A q_t$ and the second step is due to (E.4) from Appendix E. Similarly, we can use (K.9) to show that

$$
v_{ht} \overset{a.s.}{=} \lim_{N \to \infty} \frac{1}{N} \|h_t\|^2 = \lim_{N \to \infty} \frac{1}{N} \|q_t + A^T z_t\|^2
$$

\[= a.s. \lim_{N \to \infty} v_q + 2 \frac{1}{N} q_t^T A^T z_t + \frac{1}{N} \|A^T z_t\|^2
\]

\[= a.s. \, v_w + v_q \chi_2 - v_q, \tag{K.15}\]

where we also used (5.16). Lastly, note that because in MF-OAMP we have $W_t = I$, the scalar $\gamma_t$ from (K.10) is equal to $-1$ since $\lim_{N \to \infty} \frac{1}{N} q_{t+1}^T A^T z_t a.s. = -v_q$. Then, substituting (K.11), (K.13) and (K.15) into (K.2) gives us the final result for MF-OAMP

$$
\lim_{N \to \infty} \frac{1}{2} (u_2 + 2 \alpha_t u_3) \overset{a.s.}{=} (\chi_2 - 1)v_q + \bar{\psi}_t - \bar{\psi}_t \chi_2
$$

\[= (\bar{\psi}_t - v_q) - \chi_2(\bar{\psi}_t - v_q)
\]

\[= (1 - \chi_2)(\bar{\psi}_t - v_q)\].
K.2 VAMP

In VAMP we have $W_t = v_wI + v_q AA^T$ so that

$$AA^T = \frac{W_t - v_wI}{v_q}, \quad (K.16)$$

which implies that (K.11) is equivalent to

$$\lim_{N \to \infty} \frac{1}{N} q_{t+1}^T A^T AA^T W_t z_t = \lim_{N \to \infty} \frac{1}{N} q_{t+1}^T A^T W_t W_t^{-1} z_t - v_w \frac{1}{N} q_{t+1}^T A^T W_t^{-1} z_t \quad a.s. \quad (K.17)$$

Next we refer to the result from [85] stating that in VAMP the variance $v_{ht}$ follows

$$v_{ht} \overset{a.s.}{=} \lim_{N \to \infty} \gamma_t^{-1} - v_{qt}, \quad (K.18)$$

Then, substituting (K.17) and (K.18) into (K.2) leads us to

$$\lim_{N \to \infty} \frac{1}{2} (u_2 + 2\alpha_t u_3) \overset{a.s.}{=} \psi_t + v_{ht} - \gamma_t^{-1} \psi_t \frac{v_w}{v_{qt}} + \frac{v_w \psi_t}{v_{qt}} - v_w. \quad (K.19)$$

Finally, we can show that

$$\frac{v_w \psi_t}{v_{qt}} - v_w = \frac{v_w}{v_{qt}} (\psi_t - v_q) \quad (K.20)$$

and

$$\psi_t + v_{ht} - \gamma_t^{-1} \frac{v_w}{v_{qt}} = \frac{\psi_t v_{qt} + v_{ht} v_{qt} - \gamma_t^{-1} \psi_t}{v_{qt}}$$

$$= \frac{(v_{qt} - \gamma_t^{-1}) \psi_t + v_{ht} v_{qt}}{v_{qt}}$$

$$= -v_{ht} \psi_t + v_{ht} v_{qt} \quad (K.22)$$

$$= -v_{ht} (\psi_t - v_q), \quad (K.23)$$

where in (K.22) we used (K.18). With these results, we obtain

$$\lim_{N \to \infty} \frac{1}{2} (u_2 + 2\alpha_t u_3) \overset{a.s.}{=} \frac{(v_w - v_{ht})}{v_{qt}} (\psi_t - v_q), \quad (K.24)$$

which completes the proof. $lacksquare$
Appendix L

Proof of Lemma 4

Here we prove Lemma 4, which provides the sufficient conditions for the difference $v_{ht} - v_{w}$ to be positive. We begin with noting that based on (K.18) we have that

$$ \lim_{N \to \infty} \gamma^{-1}_{t} - v_{q_{i}} - v_{w} = \lim_{N \to \infty} \frac{1 - \gamma_{t}(v_{q_{i}} + v_{w})}{\gamma_{t}}. \quad (L.1) $$

Because $\gamma_{t}$ is strictly positive [85], to ensure the positivity of (L.1) it is sufficient to show that $1 - \gamma_{t}(v_{q_{i}} + v_{w})$ is positive. Next, we notice that as $N \to \infty$, we can define $\gamma_{t}$ through the limiting eigenvalue distribution $p(\lambda)$ of $\Lambda = SS^{T}$ as [85]

$$ \lim_{N \to \infty} \gamma_{t} = \delta \int \frac{\lambda}{v_{w} + v_{q_{i}} \lambda} p(\lambda) d\lambda. \quad (L.2) $$

With (L.2), we can rewrite the term $\gamma_{t}(v_{q_{i}} + v_{w})$ as

$$ \gamma_{t}(v_{q_{i}} + v_{w}) = \delta \int \frac{\lambda(v_{q_{i}} + v_{w})}{v_{w} + v_{q_{i}} \lambda} p(\lambda) d\lambda $$

$$ = \delta \int \frac{v_{q_{i}} \lambda + v_{w} - v_{w} + v_{w} \lambda}{v_{w} + v_{q_{i}} \lambda} p(\lambda) d\lambda $$

$$ = \delta \left( 1 + \int \frac{v_{w} \lambda - v_{w}}{v_{w} + v_{q_{i}} \lambda} p(\lambda) d\lambda \right). \quad (L.3) $$
which implies that the nominator of (L.1) is equivalent to

\[
1 - \gamma_t(v_q + v_w) = \delta \left( \delta^{-1} - 1 - \int \frac{v_w \lambda - v_w}{v_w + v_q \lambda} p(\lambda) d\lambda \right)
\]

\[
= \delta \left( \int \frac{(\delta^{-1} - 1)(v_w + v_q \lambda) - v_w \lambda + v_w}{v_w + v_q \lambda} p(\lambda) d\lambda \right)
\]

\[
= \delta \left( \int \frac{(\delta^{-1} - 1)v_q - v_w + \delta^{-1}v_w}{v_w + v_q \lambda} p(\lambda) d\lambda \right). 
\]  

(L.4)

Since all the scalar variables in (L.4) are positive and \( p(\lambda) \) is a probability density function, it is sufficient to ensure that

\[
v_q \geq \frac{v_w}{\delta^{-1} - 1}
\]  

(L.5)

for (L.4) to be positive, which implies positivity of (L.1). 

\[\blacksquare\]
References


Proof of Lemma 4


Proof of Lemma 4


Proof of Lemma 4


Proof of Lemma 4


Proof of Lemma 4


Proof of Lemma 4


