UNIVERSITY OF CALIFORNIA

Los Angeles

Exact Analysis of Inverse Problems in High Dimensions with Applications to Machine Learning

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Electrical and Computer Engineering

by

Parthe Pandit

2021

© Copyright by

Parthe Pandit

2021

ABSTRACT OF THE DISSERTATION

Exact Analysis of Inverse Problems in High Dimensions

with Applications to Machine Learning

by

Parthe Pandit

Doctor of Philosophy in Electrical and Computer Engineering

University of California, Los Angeles, 2021

Professor Alyson K. Fletcher, Chair

Modern machine learning techniques rely heavily on iterative optimization algorithms to solve high dimensional estimation problems involving non-convex landscapes. However, in the absence of knowing the closed-form expression of the solution, analyzing statistical properties of the estimators remains challenging in most cases. This dissertation provides a framework, called Multi-layer Vector Approximate Message Passing (ML-VAMP), for analyzing optimization-based estimators for a broad class of inverse problems. This framework is based on new developments in random matrix theory. Importantly, it does not rely on convex analysis and applies more broadly to non-convex optimization problems. The ML-VAMP framework enables exact analysis in a certain high dimensional asymptotic regime for several problems of interest in machine learning and signal processing. In particular, the following problems have been explored in some detail,

1. Reconstruction of signals from noisy measurements using deep generative models,

2. Generalization error analysis of learned generalized linear models,

to demonstrate the analytical capabilities of the framework. Using this framework we can analyze the effect of important design choices such as degree of overparameterization, loss function, regularization, initialization, feature correlation, and a mismatch between train and test data in several problems of interest in machine learning.

The dissertation of Parthe Pandit is approved.

Sundeep Rangan

Arash A. Amini

Jonathan C. Kao

Abeer Alwan

Alyson K. Fletcher, Committee Chair

University of California, Los Angeles

2021

To my parents

Contents

List of Figures

Acknowledgements

This dissertation has been possible in large part due to the endless support from my advisors, mentors, teachers, colleagues, family, and friends.

To start off, I would like to thank my advisor Professor Alyson Fletcher, who gave me the opportunity to pursue graduate studies in machine learning at a vibrant place such as UCLA. I enjoyed a lot of intellectual autonomy working as her student, and her advise always kept me from getting stuck in local minimas. I thank her for this guidance throughout my PhD.

It was my utmost fortune to be guided by Professor Sundeep Rangan, whose technical prowess never ceases to amaze me. I hope that one day I can start seeing patterns in seemingly unrelated problems like him. His ability to make bold predictions and then prove them using the most elementary mathematical tools makes research look so simple.

It is hard to overstate the influence Professor Arash Amini has had in my training as a statistician. His acute attention to detail always provided me with clarity of thought. Conversations with him have been the most intellectually engaging during my time at UCLA.

I would also like to thank Professors Abeer Alwan and Jonathan Kao for agreeing to be on my committee and for providing his valuable input. I am also very grateful to have received advice from Professors Raghu Meka and Alexander Sherstov who provided a sandbox for learning advanced concepts in theoretical computer science. Professor Suhas Diggavi's class on Information Theory helped me overcome my fear of probability, which nudged me to take up statistics and machine learning. I would also like to thank Professor Phillip Schniter for being an amazing collaborator and for making NeurIPS 2018 a far less daunting experience.

I was lucky to have as mentors Professors Sam Coogan and Ankur Kulkarni who gave me the confidence to take up problems that I found interesting and make them into a cohesive and coherent story. The fundamental concepts and visualization techniques I learned from them has made it so much easier to understand and interpret a broad range ideas I encounter.

UCLA has an amazing Mathematics department which I greatly benefited from. Math 275 and 273 series taught by Professors Jun Yin and Wotao Yin respectively were crucial in my understanding of the landscape of problems considered in machine learning. Attending the Level Set Collective meetings every Tuesday afternoon at IPAM was, in hindsight, the best passive investment of my time, and introduced me to optimal control, optimal transport, and operator splitting. The reading group run by Professors Liza Rebrova and Palina Salanevich exposed me to Graph Signal Processing. I thank them all.

My internship experiences were amazing due to amazing mentors Sumeet Katariya, Nikhil Rao, Sheng Zha, He He, and Hua Zheng. Their guidance enriched my training as a researcher.

I acknowledge the financial support from the National Science Foundation, Office of Naval Research, Gurukrupa foundation, J. N. Tata endowment, and K. C. Mahindra foundation.

My colleagues, coauthors, and compadres: Mojtaba Sahraee-Ardakan and Melikasadat Emami have provided a great deal of support through what could have been a much lonelier experience. I have learned a lot by being in their presence. It was also great to be among PhD students like Michael, Amber, Sundar, Vijay, Srini, Tsang-Kai, Arsalan, and Vignesh, to have long-drawn discussions over several trips to Kerchoff, Synapse, and Southern Lights.

I was lucky to have brilliant roommates like Pratik Sathe and Mihir Laghate, who provided me the best home-away-from-home. My friends Navjot, Akshay, Shantanu, Janaki, and Pratik Chaudhari made LA more like Mumbai. I am extremely fortunate to have kind friends like Arpit, Niladri, Shantanu, Saurabh, and Vighnesh. I am greatly indebted to Chaphy and Vidya-aaji, whose unbounded affection always made me feel secure in Southern California. Rosie, who I inherited from Vidya-aaji, made life in LA far more enjoyable.

CAPS and the Ashe Center helped cope with the struggles of life as an international graduate student. I shall always remember the impact that the staff, program volunteers, and the group members I interacted with, had on my health and well being.

My partner Harini Alladi has been a constant source of inspiration and joy. I have grown a lot personally and professionally because of her. I am grateful that she is a part of my life.

Finally, I would like to thank my parents, and my sister for their unwavering support.

Vita

Education

Indian Institute of Technology, Bombay	July 2010 - July 2015
Bachelor of Technology in Electrical Engineering	
(minor in Computer Science and Engineering)	
Master of Technology in Electrical Engineering	
(Signal processing and Communication)	

Internships

Junior research fellow, Systems and Controls group, IIT Bombay	Aug 2015 - Aug 2016
Applied scientist II intern, Amazon Search	Mar - May 2020
Applied scientist II intern, Amazon AWS	June - Aug 2020
Quantitative research intern, Citadel LLC	June - Aug 2021

Awards

J. N. Tata endowment fellowship	2016
K. C. Mahindra foundation fellowship	2016
J. N. Tata endowment award	2019
Gurukrupa foundation fellowship	2019
Jack K. Wolf student paper award, ISIT	2019
HDSI-Simons postdoctoral fellowship, UC San Diego	2021

Chapter 1

Introduction

Throughout history, data analysis has played an integral role in science, engineering, business, and governance. In the past, collecting and analyzing data was extremely costly. However, we are now able to collect, model, and analyze vast amounts of data, at a fraction of the cost. Consequently, there is rising demand for sophisticated techniques to interpret and extract information from complex datasets. It is no surprise that the techniques for data analysis have changed drastically in its scale, and are evolving rapidly. Among these techniques, Machine Learning has received a lot of attention recently from academia and industry.

A key feature of contemporary techniques in machine learning and signal processing is their massive scale, both in terms of the number of model parameters as well as the size of the datasets used for training them. Models in recent developments in deep learning [80], for example, employ billions of parameters with hundreds of layers [143]. Signal processing tasks have evolved into estimation problems over millions of variables (e.g. pixels in a high resolution image or video). Using advances in convex and non-convex optimization techniques, large-scale learning and inference algorithms have demonstrated super-human performance on several perception tasks.

A general design principle practiced by machine learning engineers is, vaguely: general purpose models (neural networks), trained with general purpose iterative optimization algorithms (stochastic gradient descent), can work *well-enough* when trained on *large-enough* datasets. Here, general-purpose usually means that application-specific knowledge is not incorporated – at least not explicitly – during modelling or training. In coarse terms, this practice is called deep learning. The past decade has seen a dramatic rise in the application of deep learning to a variety of areas of science and engineering.

Yet, from a theoretical perspective, the properties of deep learning as well as many other methods in machine learning are largely unresolved. Although designed as general-purpose solutions, in reality, the performance of the learned models is often highly sensitive to the training procedure and the numerous hyperparameters involved. Locating the *sweet-spot* in the space of hyperparameters in which a method succeeds (or fails), and how this depends to the properties of the data distribution remains a mystery. In order to make machine learning *truly* general-purpose, it is important to build a principled understanding for such dependencies.

Unconstrained, convex and non-convex optimization problems play an important role in modelling in present-day machine learning methodology. Typical training procedures for these models deploy iterative, large-scale optimization algorithms such as stochastic gradient descent (SGD), Adam, and their variants.

With the goal to better understand the theoretical foundations and fundamental limits of the methodology in machine learning and signal processing, in this dissertation we present a statistical framework which enables an *exact* tractable analysis of optimization based estimators in a certain high dimensions. We call this framework Multi-Layer Vector Approximate Message Passing (ML-VAMP). It is based on a class of iterative estimation algorithms whose iterations possess an equivalent scalar representation for large random problem instances.

The regime of statistical analysis that this dissertation will focus on is the *proportional asymptotic* regime. In this regime, the number of known quantities (measurements/observations) and the number of unknown (variables/parameters) grow to infinity but with a fixed ratio.

In the rest of this chapter, we introduce inverse problems and put in perspective the roles of optimization and high dimensional statistics in modern machine learning and signal processing.

1.1 Inverse Problems

Inverse problems are a set of well studied techniques in applied mathematics, with a rich history. In essence, inverse problems enable us to specify phenomena without describing them explicitly, but by providing a model for how the phenomenon manifests into an observation.

While there may be deeper philosophical, cognitive, and linguistic implications of inverse problems to learning and intelligence, we restrict our focus to a subset of inverse problems that have received a lot of attention in signal processing and machine learning over the past few decades.

The class of inverse problems we focus on can be stated as:

$$\text{Find the unknown signal } \boldsymbol{x} \text{ from measurements } \boldsymbol{y} = \phi(\boldsymbol{A}\boldsymbol{x}; \xi) \tag{1.1}$$

where ξ is an unknown noise. The tuple $(\phi, \boldsymbol{A})$ is the model, where ϕ is a known nonlinear operator and $\boldsymbol{A}$ is a known linear operator. We refer to the proposed solution as $\widehat{\boldsymbol{x}}$.

1.1.1 Performance analysis: Accuracy and Efficiency

Suppose the unknown signal $\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^N$ and measurements $\boldsymbol{y} \in \mathcal{Y} \subseteq \mathbb{R}^M$. The performance of a reconstruction technique can be measured in several ways. To do so, one often starts by assuming that there exists a *ground truth* signal $\boldsymbol{x}^*$, which led to the measurements $\boldsymbol{y}$. One popular metric for performance analysis is the mean-squared error (MSE) or the normalized MSE:

$$\mathsf{MSE}(\widehat{\boldsymbol{x}}, \boldsymbol{x}^*) := \frac{1}{N} \left\|\widehat{\boldsymbol{x}} - \boldsymbol{x}^*\right\|^2 \qquad \mathsf{NMSE}(\widehat{\boldsymbol{x}}, \boldsymbol{x}^*) := \frac{\left\|\widehat{\boldsymbol{x}} - \boldsymbol{x}^*\right\|^2}{\left\|\boldsymbol{x}^*\right\|^2}, \tag{1.2}$$

More generally, this metric could be arbitrary and specific to the application. For e.g. the metric using in image denoising is often peak signal to noise ratio (PSNR).

The quantity M can be thought of as a measure of the *cost of acquisition*, whereas $\dim(\mathcal{X})$ indicates the *complexity* of the signal. A desirable property of any solution procedure is its ability to deal with high complexity $\mathcal{X}$ with low cost of acquisition. The above notion of cost can be called statistical efficiency. Similarly, there is a notion of computational efficiency which relates to the computational cost of achieving a desired level of performance. Consequently, from the point of view of design, we are interested in the most computationally efficient way of finding $\boldsymbol{x}$ given some constraints on the level of performance.

1.1.2 Optimization based solutions

There are several methodologies for solving inverse problems. One such technique is the *level set method*. A realization of this approach is to formulate the proposal $\widehat{\boldsymbol{x}}$ as the solution to an optimization formulation, so that the proposed solution minimizes a data fitting term that depends on the model $(\phi, \boldsymbol{A})$ and observations $\boldsymbol{y}$.

For example, a common solution concept for the linear model $\phi(u, v) = u + v$ is the ordinary least squares (OLS) solution:

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2 \tag{OLS}$$

A common theme in the theory of inverse problems is that the problem is ill-posed or under-determined, i.e., there is no unique solution to the observations. However, we are interested in certain structured proposals $\widehat{\boldsymbol{x}}$ which can best explain $\boldsymbol{y}$. For example, if $\boldsymbol{x} \in \mathbb{R}^N$ and $\boldsymbol{y} \in \mathbb{R}^M$ with $N < M$, and $\boldsymbol{x}^*$ is a minimizer of (OLS), then $\boldsymbol{x}^* + t\boldsymbol{\Delta}$ is also a minimizer for any $t \in \mathbb{R}$ and any $\boldsymbol{\Delta} \in \text{null}(\boldsymbol{A})$, which is non-empty when $N < M$.

To solve ill-posed problems, a common approach is to add a penalty term to the objective function of the minimization that penalizes large parameters. A popular solution is called

Tikhonov regularization, or ridge penalty:

$$\underset{\boldsymbol{x}}{\text{minimize}} \;\; \tfrac{1}{2}\left\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\right\|^2 + \lambda \left\|\boldsymbol{x}\right\|_2^2 . \tag{1.3}$$

where λ is the strength of the penalty. The solution for the above problem can be written in closed form as

$$\widehat{\boldsymbol{x}}_\lambda := (\boldsymbol{A}^\top \boldsymbol{A} + \lambda I)^{-1}\boldsymbol{A}^\top \boldsymbol{y} \tag{1.4}$$

For $\lambda \to 0$, the solution to the above problem converges to the $\boldsymbol{A}^\dagger \boldsymbol{y}$ where $\boldsymbol{A}^\dagger$ is the pseudo-inverse of $\boldsymbol{A}$. We can then perform analysis of this solution since we know it in closed form.

In imaging related problems, there are settings where the desired solution $\widehat{\boldsymbol{x}}$ is *sparse*, i.e., has few non-zero elements compared to the dimension of the ambient space. Another popular technique in imaging related problems is often called compressive sensing where the following optimization problem is solved:

$$\underset{\boldsymbol{x}}{\text{minimize}} \;\; \tfrac{1}{2}\left\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\right\|^2 + \lambda \left\|\boldsymbol{x}\right\|_1 \tag{LASSO}$$

This optimization problem is convex, but non-smooth, and is often referred to as the LASSO, short for least absolute shrinkage and selection operator.

An immediate challenge with this formulation is that we do not know the solution to (LASSO) in closed form in terms of $(\boldsymbol{y}, \boldsymbol{A})$. Consequently analysis is challenging. Our main focus will be on understanding properties of the solutions to (1.1) for large random operators $\boldsymbol{A}$. Several problems in machine learning can be expressed in their simplest form in (1.1) as described in the following section.

1.2 Analysis of Optimization in Machine Learning

We briefly introduce supervised machine learning and relate it to inverse problems. A typical characteristic in modern data analysis methodology is access to large sample sizes (n) with large number of covariates (p). In supervised machine learning, we are provided a sample of response-covariate pairs $\{y_i, \boldsymbol{x}_i\}_{i=1}^n$ and we wish to learn a function that predicts $\boldsymbol{x} \mapsto y$.

A simple way in which one can pose the learning problem as an inverse problem is to propose that the responses and covariates satisfy:

$$\boldsymbol{y} = \phi(\boldsymbol{X}\theta; \boldsymbol{\xi})$$

where $\boldsymbol{y}$ is a vector of responses y_i and $\boldsymbol{X}$ is a matrix with covariates $\boldsymbol{x}_i$ as rows. Thus the learning problem is to find the unknown θ. Observe that the above equation resembles (1.1).

1.2.1 Classical statistical estimation and analysis

A classical method in statistical estimation is the maximum likelihood estimator (MLE), which in an overwhelming number of scenarios can be expressed as:

$$\widehat{\theta}_{\mathsf{mle}} = \underset{\theta \in \mathbb{R}^p}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(\theta; \boldsymbol{x}_i, y_i) \tag{MLE}$$

where $\theta \mapsto \mathcal{L}(\theta, x_i)$ is a negative-log-likelihood function, and $\{\boldsymbol{x}_i, y_i\}_{i=1}^n$ is a generic sample. This minimization is equivalent to maximizing $\theta \mapsto \mathbb{P}_\theta(\{\boldsymbol{x}_i, y_i\}_{i=1}^n)$ – the likelihood of observations.

Traditional statistical methods such as the MLE are, under mild regularity assumptions, statistically consistent if $p = O(\log(n)) \ll n$. Note we are using the "big-Oh" notation[1]. Moreover, applying the central limit theorem shows that the deviations $\sqrt{n}(\widehat{\theta}_{\mathsf{mle}} - \theta^*)$ weakly converge to $\mathcal{N}(0, I(\theta^*)^{-1})$ where $I(\theta^*)$ is called the Fisher information matrix. This weak

[1]see Section 2.1 for a recap

convergence essentially provides the characterization,

$$\lim_{n\to\infty} \frac{1}{n}\sum_{i=1}^{n} \psi(\theta^*, \widehat{\theta}_{\mathsf{mle}}) = \mathbb{E}\psi(\theta^*, \theta^* + Z) \tag{1.5}$$

for a broad class of distortion metrics ψ (continuously bounded functions). Note $Z \sim \mathcal{N}(0, I(\theta^*)^{-1})$ and the right-hand side can be evaluated using simulation methods like the Markov Chain Monte Carlo (MCMC), when p is small. The above result is often referred to as the *asymptotic noramlity* property of the MLE.

The idea of MLE was generalized to M-estimators, short for "maximum-likelihood type", by Huber [66, 67] leading to the rich area of Robust Statistics.

We would like to note that, in the context of machine learning, the function $\mathcal{L}$ is called a *Loss-function*, and the objective function of (MLE) is seen as the plug-in for, or empirical version of, the *Risk function* which is $\mathbb{E}_{\mathbb{P}_\theta}\mathcal{L}(\theta, x)$. Consequently the estimator (MLE) is called the Empirical Risk Minimization (ERM) problem.

1.2.2 High-dimensional statistics: M-estimation and Regularization

If we were to directly apply the MLE from equation (MLE) when $p \nll n$, then we need $n = \Omega(e^p)$ for statistical consistency, which is often referred to as the *curse of dimensionality*. In what follows, we refer to $p \nll n$ as the high-dimensional regime.

In several problems of interest however, even though the model parameter θ lies in a large ambient space $\mathbb{R}^p$, i.e., when p is large enough, θ may often lend itself to a parsimonious representation. Some examples are θ is sparse, i.e., has very few non-zero entries, or is sparse in a basis, i.e., $\theta = \boldsymbol{D}\boldsymbol{u}$ for a sparse $\boldsymbol{u}$ and dictionary $\boldsymbol{D}$. In such scenarios, a modification of problem (MLE) performs well:

$$\widehat{\theta}_{\mathsf{r-mle}} = \operatorname*{argmin}_{\theta} \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(\theta; \boldsymbol{x}_i, y_i) + \lambda_n \mathcal{R}(\theta) \tag{1.6}$$

where $\mathcal{R}$ is a *regularization* function, which encourages a certain structure for the solution.

Some examples of $\mathcal{R}$ are:

(i) Hard sparsity $\|\theta\|_0 = \sum_{j=1}^{p} \mathbb{1}_{\{\theta_j \neq 0\}}$,

(ii) Ridge regularization $\frac{1}{2}\|\theta\|^2 = \frac{1}{2}\sum_{j=1}^{p} |\theta_j|^2$,

(iii) LASSO $\|\theta\|_1 = \sum_{j=1}^{p} |\theta_j|$,

(iv) Elastic-Net $(1-\alpha)\|\theta\|_1 + \frac{\alpha}{2}\|\theta\|^2$,

(v) Smooth, convex penalties $\|\theta\|_q^q = \sum_{j=1}^{p} |\theta_j|^q$, for $q > 1$,

(vi) Non-smooth, Non-convex sparsity penalties $\|\theta\|_q^q = \sum_{j=1}^{p} |\theta_j|^q$, for $q < 1$,

(vii) $\widetilde{\mathcal{R}}(\boldsymbol{Du}) + \delta_{\{\theta = \boldsymbol{Du}\}}$ for representations in span$(\boldsymbol{D})$.

1.2.3 Interpretations of problem (1.6):

Observe that when $n \to \infty$, while keeping p fixed we have $\lambda_n \to 0$, and we recover the estimator (MLE). But for finite n, (1.6) serves as an approximation to (MLE). Often for finite n, (MLE) may not be *identifiable*, while (1.6) always exists under mild conditions.

A Bayesian interpretation is that $\mathcal{R}(\theta)$ is the negative-log prior distribution, whereby the objective in (1.6) is negative-log posterior, i.e., $-\log \mathbb{P}(\theta | \{\boldsymbol{x}_i, y_i\}_{i=1}^n)$ and $\widehat{\theta}_{\mathsf{r-mle}}$ is the *maximum a-posteriori* estimator, which is the mode of the a-posteriori distribution of the unknown *random* parameter θ. The hyperparameter λ_n is interpreted as the inverse of the scale parameter of the prior distribution.

Another interpretation is that $\mathcal{L}$ is a data-fitting term that measures how well a proposed parameter θ fits the data, and the regularization $\mathcal{R}$ is a penalty term to enforce a structure so that the inverse problem is *well-posed* even when $p < n$. The hyperparameter λ_n is interpreted as the inverse of the signal-to-noise-ratio (SNR).

In statistical physics, the objective of (1.6) is often termed the Gibbs Free energy, with the first term referred to as the average potential energy of n particles, whereas the second term $\mathcal{R}$ being the entropy of the configuration θ, and the hyperparameter λ_n as the temperature of the system.

To contextualize in the vocabulary pertinent to machine learning theory, problems of the form (1.6) are also called Structural Risk Minimization (SRM) and are a generalization of ERM.

1.2.4 Appeal of M-estimators in high dimensions

Optimization as a numerical technique has advanced significantly over the past few decades. Arguably the most important innovation in optimization has been the free and open access to *automatic differentiation* libraries in high level programming languages, which has enabled the development of a large class of first and second order optimization algorithms. Example libraries in the Python programming language include Autograd, Caffe, Tensorflow, Pytorch, JAX, MNNet among others. This has led to developments such as stochastic, distributed algorithms for optimizing general non-convex differentiable/sub-differentiable functions. With affordable access to graphical processing units (GPU) via cloud computing services, large-scale optimization algorithms have also become efficient due to parallel computing.

There are two main advantages to regularized M-estimation, i.e., estimators specified as solutions to optimization problems of the type (1.6). First is *interpretability*; a solution of the form (MLE) allows specifying a concept or an estimator as a variational formulation, which makes the model extremely interpretable by choosing an appropriate loss function.

Second, in the Bayesian setup, under some mild regularity conditions in low dimensions, we have that the mode of a distribution is close to the mean of the distribution, whereby $\widehat{\theta}_{\mathsf{r-mle}}$ is somewhat close[2] to the posterior mean or the Bayes estimator:

$$\widehat{\theta}_{\mathsf{Bayes}} := \mathbb{E}[\theta | \{\boldsymbol{x}_i, y_i\}_{i=1}^n] = \frac{1}{Z(\{\boldsymbol{x}_i, y_i\}_{i=1}^n)} \int \theta \cdot \mathbb{P}_\theta(\{\boldsymbol{x}_i, y_i\}) \, \mathrm{d}\theta$$

[2]The closeness between the mode and mean may not always hold in high dimensions. However, computationally, finding the mode is much more tractable than numerically evaluating the integral, for which the computational complexity can be exponential in p. See Section 2.3 for a more detailed discussion.

1.2.5 Consistency in high dimensions

A large body of work on high-dimensional statistical analysis is now well-established for non-asymptotic error bounds on the mean-squared error when the regularizer $\mathcal{R}$ is decomposable, i.e., it is a sum of functions of individual coordinates of the input. A generic consistency result in this literature (see [105] for a detailed result, and its follow-up works) is of the following nature:

$$\left\|\widehat{\theta}_{\mathsf{r-mle}} - \theta^*\right\|^2 \lesssim \lambda_n^2 \Psi(\theta^*) + \lambda_n \Phi(\theta^*) \tag{1.7}$$

The first term $\Psi(\theta^*)$ is the *estimation error*. The second term $\Phi(\theta^*)$ is an *approximation error*, which is the error resulting from the regularization penalty $\mathcal{R}$ not accurately capturing the structure of θ^*. This vanishes if the model does not suffer from mis-specification.

In general $\lambda_n = \Omega\left(\sqrt{\frac{1}{n}}\right)$ yields the parametric rate n^{-1} when there is no model mis-specification. The optimal value of the quantity λ_n also depends on p, and is typically related to square-root of some notion of the size of the set to which θ^* belongs. For example if θ^* belongs to the space of $s-$ sparse vectors in $\mathbb{R}^p$, then $\lambda_n = \Omega(\sqrt{s\log(p)/n})$.

1.2.6 Brief History of Non-convex M-estimation

During 2000-2010, much the modelling aspect of machine learning algorithm was based on the dominant choice of the training procedure – convex optimization. This not only enabled providing rigorous guarantees for the performance of the estimators, but also came with off-the-shelf algorithms to solve the estimators.

Following the empirical success of non-convex optimization based estimators such as deep neural networks however, the focus of modelling has shifted to models which may need solving a non-convex optimization problem as its training procedure. This has also led to significant advances in our understanding of non-convex optimization formulations such as matrix factorization and matrix completion which were previously considered "hard" due to the non-convexity. Results such as those by [49] showed that any local optima is a global optimal for a large class of non-convex optimization problems of broad interest in machine learning.

During 2010-2020, the increased comfort in working with non-convex optimization formulations has led us back to *faithful modelling*, which incorporates application-specific information into the design of the model, without requiring convexity of the optimization problem. While this may make models more interpretable, unlike convex optimization however, the analysis frameworks are specialized and problem-specific and hence may not apply from one class of problems to another.

1.2.7 Challenges in analyzing high-dimensional M-estimators

While results of the type (1.7) have furthered our understanding of what type of models are tractable statistically as well as computationally in the high dimensional setting, the analysis framework has some limitations as described below.

First, the analysis framework in [105] relies heavily on the convex geometry of the objective function. Hence there are immediate challenges for non-convex optimization formulations even if the problems have unique solutions or no spurious local minima. The reliance on the convex analysis precludes our ability to understand more complex structures such as optimization problems involving deep neural networks.

Second, while the non-asymptotic results are extremely powerful, they are qualitative in nature. The unknown universal constants hidden in the inequality $\lesssim$ add a major barrier to directly applying these results from theory to practice.

Third, these results also depend on the structure of the $\|\cdot\|$ for which the bound is provided. Consequently, analyzing the distortion arbitrary metrics as in the low-dimensional case (1.5), is in general hard. This poses serious challenges from the practicality of the results since the distortion metrics often vary by not just the application but also often by business needs, regulation constraints, etc.

Fourth, the optimization problems – even for convex formulations – are never solved exactly till convergence, and often rely on an iterative algorithm. The analysis being purely based on KKT conditions of the solution is agnostic to the dynamics of the optimization algorithm used to find the solution. This precludes understanding the so-called *implicit bias* of M-estimation arising out of algorithmic choices made during the iterative optimization.

Finally, results of the type (1.7) suggest that the generalization error of an estimator has the classical U-shaped curve similar to the bias-variance tradeoff. However, recent empirical results, such as the so called *double-descent* phenomenon, in the context of overparameterized modelling suggests that the story behind the performance of M-estimators, even for convex formulations, may not be as straightforward. While such behaviour isn't contradictory to results by [105], the framework certainly does not aid the discovery of such phenomena due to the lack of precise characterizations of distortions.

1.3 Analysis Framework

Our analysis framework is based on a class of iterative decoding algorithms called Vector Approximate Message Passing (VAMP). The key property that these algorithms possess is a weak convergence of iterations to a scalar nonlinear dynamical system, called the State Evolution. Such a weak convergence result is similar in spirit to the asymptotic normality of the (MLE).

The weak convergence result enables exact analysis of the proposed solutions for a large class of metrics. We can provide a formula for the distortion between a model for the ground truth and the proposed solution. This formula is exact, i.e., without any large unknown universal constants, or without any inequalities or bounds. Such an exact analysis enables a more fine-grained analysis of estimators, with tight characterization of several phase-transition phenomena related to the success or failure of the estimators.

1.3.1 Contributions to high dimensional statistics:

Our framework tries to bridge the gap between theory and practice by focusing on a smaller regime, called the *proportional asymptotic regime*. In this regime we have

$$\lim_{n\to\infty} \frac{p}{n} = \beta \in (0,\infty) \tag{1.8}$$

for some constant β. Using the ML-VAMP framework, we can make precise predictions about a broad class of error metrics, similar to equation (1.7). A typical result using the ML-VAMP framework will apply to a sequence of problems, with $(\boldsymbol{x}^0(n), \widehat{\boldsymbol{x}}^t(n)) \in \mathbb{R}^{p_n}$ so that (1.8) holds, we have

$$\lim_{n\to\infty} \frac{1}{p_n} \sum_{i=1}^{p_n} \psi(x_i^0(n), \widehat{x}_i^t(n)) = \mathbb{E}\psi(X^0, \widehat{X}^t) \tag{1.9}$$

where X^0, X^t are low-dimensional random variables with known probability distributions, whereby the RHS can be calculated with relative ease. We also provide detailed assumptions under which such a result would hold. Here t is the number of iterations for which the VAMP iterative algorithm is run.

1.3.2 Relation to Replica Method from Statistical Physics

The replica method has a rich history in the field of statistical physics of spin glasses. However these results are derived from a heuristic derivation called the *replica trick*, and lack rigorous theoretical justifications. In several cases, the predictions made by the replica method match our predictions; however the assumptions under which an exact equivalence holds between the predictions made by these two methods, remains unclear. While the replica theoretic framework typically assumes random matrices are entrywise i.i.d. random, the ML-VAMP framework applies more broadly to the class of rotationally invariant random matrices. This class of random matrices includes matrices which can have arbitrary laws of singular values which enable modelling of more general problems in machine learning such as those considered in Chapter 5.

1.4 Organization of the dissertation

The rest of this dissertation is organized as follows: Chapter 3 states a general multi-layer inverse problem that the ML-VAMP framework is based on. Chapter 4 generalizes this framework to a broader class of problems involving matrix variables. These two chapters discuss the reconstruction of vector and matrix valued signals using deep generative models from noisy nonlinear measurements. Chapter 5 provides an application of the framework to analyzing some problems in machine learning, and provides a precise characterization of generalization error in learned models. All proofs are deferred to the corresponding appendices, and all chapters are self-contained. Results from Chapter 3 appeared in [43, 112, 115], those from Chapter 4 appeared in [116, 117] and were coauthored with Mojtaba Sahraee-Ardakan, Sundeep Rangan, Philip Schniter and Alyson Fletcher. Contents of Chapter 5 were published as [39] and were coauthored with Melikasadat Emami, Mojtaba Sahraee-Ardakan, Sundeep Rangan and Alyson Fletcher.

Chapter 2

Background and Preliminaries

In this chapter we review some related concepts necessary to develop our framework.Let us start by setting up notation for the rest

2.1 Notation

Matrices are denoted by boldfaced uppercase letters $\boldsymbol{A}$. $\boldsymbol{A}_{i*}$ denotes the i^{th} row and $\boldsymbol{A}_{*j}$ the j^{th} column. Vectors are denoted by boldfaced lowercase letters $\boldsymbol{a}$ and a_i is the i^{th} coordinate of $\boldsymbol{a}$, whereas a subvector of $\boldsymbol{a}$ indexed by $\beta \subseteq \{1, 2, \ldots, \dim(\boldsymbol{a})\}$ is denoted as $\boldsymbol{a}_\beta$. Norms without subscripts are 2-norms for vectors and Frobenius norms for matrices.

We assume that densities exist for the probability distributions being discussed. In case of discrete/hybrid distributions, with some abuse of notation, a point mass at points $\boldsymbol{x}^* \in \mathcal{X}$, is represented by the term $\delta_{\boldsymbol{x}^*}$. We omit the subscripts from probability densities $p_X(\boldsymbol{x})$, $p_{X|Y}(\boldsymbol{x}|\boldsymbol{y})$, $p_{X,Y}(\boldsymbol{x}, \boldsymbol{y})$ and denote them as $p(\boldsymbol{x})$, $p(\boldsymbol{x}|\boldsymbol{y})$ and $p(\boldsymbol{x}, \boldsymbol{y})$ respectively, unless disambiguation is needed. Unless specified otherwise, for subsets of $\mathbb{R}^N$, the base measure is the Lebesgue measure.

For a probability density function $q : \mathcal{X} \to \mathbb{R}$, and a function $f : \mathcal{X} \to \mathbb{R}$, we use the notation

$$\mathbb{E}[f|q] := \int_{\mathcal{X}} f(\boldsymbol{x})q(\boldsymbol{x})d\boldsymbol{x}$$

to denote the expectation w.r.t. density q, while $\mathbb{E}[\boldsymbol{x}|\boldsymbol{y}] := \mathbb{E}[\boldsymbol{x}|p_{X|Y}] =: \mathbb{E}[\boldsymbol{x}|p(\boldsymbol{x}|\boldsymbol{y})]$. Closely related is the notation for beliefs, or unnormalized densities b, where we use the notation $\mathbb{E}[\boldsymbol{x}|b] \equiv \mathbb{E}[\boldsymbol{x}|\frac{b(\boldsymbol{x})}{\int_{\mathcal{X}} b(\boldsymbol{x})d\boldsymbol{x}}]$. For the variance terms,

$$\mathbb{V}\{f|q\} = \int_{\mathcal{X}} (f(\boldsymbol{x}) - \mathbb{E}[f|q])^2 q(\boldsymbol{x})d\boldsymbol{x}$$

denotes the variance calculated under the density q, while $\mathbb{V}\{\boldsymbol{x}|\boldsymbol{y}\} := \mathbb{V}\{\boldsymbol{x}|p_{X|Y}\} =: \mathbb{V}\{\boldsymbol{x}|p(\boldsymbol{x}|\boldsymbol{y})\}$. When using the notation for $\mathbb{E}$ and $\mathbb{V}$, it is implicitly assumed that these quantities are well defined or that the necessary moments exist and are finite.

We use the standard "big-Oh" notation. For some universal constants C_i, N_i below:

$$\begin{aligned}
f_n \lesssim g_n \equiv f_n = O(g_n) &\equiv f_n \leq C_1 g_n \text{ for all } n \geq N_1, \\
f_n \gtrsim g_n \equiv f_n = \Omega(g_n) &\equiv f_n \geq C_2 g_n \text{ for all } n \geq N_2, \\
f_n = o(g_n) &\equiv \lim_{n\to\infty} \frac{f_n}{g_n} = 0, \\
f_n = \omega(g_n) &\equiv \lim_{n\to\infty} \frac{g_n}{f_n} = 0
\end{aligned}$$

2.2 Useful results from Probability theory

We will review concepts such as convergence in probability, almost sure convergence, and convergence in distribution, via the statements of the laws of large numbers. To that end, let $S_i = \sum_{i=1}^{n} X_i$.

Theorem 1 (Weak Law of Large Numbers). *If $\{X_i\}$ are centered i.i.d. random variables, with a law satisfying the tail condition $\mathbb{P}(|X_i| > x) = o(\frac{1}{x})$, then*

$$S_n/n \to 0, \quad \textit{in probability.}$$

which means $\lim_{n\to\infty} \mathbb{P}(|S_n/n| > \epsilon) = 0$ for all $\epsilon > 0$.

Theorem 2 (Strong Law of Large Numbers)**.** *If $\{X_i\}$ are centered pairwise independent integrable random variables,* i.e., $\mathbb{E}|X_i| < \infty$, then

$$S_n/n = 0, \quad \text{almost surely (a.s.)}$$

which means that $\mathbb{P}(\lim_{n\to} |S_n/n| > \epsilon) = 0$ for all $\epsilon > 0$. Notice this is a stronger statement than the weak law because of the order of the limit and the integral.

Theorem 3 (Central Limit Theorem)**.** *If $\{X_i\}$ are centered i.i.d. random variables with variance $\mathbb{E}|X_i|^2 = \sigma^2 < \infty$, then*

$$S_n/\sqrt{n} \implies \mathcal{N}(0, \sigma^2)$$

where the above convergence is in distribution. This means that for any bounded continuous function f, we have $\mathbb{E}f(S_n/\sqrt{n}) \to \mathbb{E}f(S)$, where $S \sim \mathcal{N}(0, \sigma^2)$. This is the weakest form of convergence. However, it holds even when S_n/n is not a continuous random variable, i.e., has a domain which is a strict subset of $\mathbb{R}$. Other stronger forms of convergence may not make sense under such a discrepancy in the domains of the random variables.

2.3 Approximate Bayesian Inference

Consider an unknown signal $\boldsymbol{x} \in \mathcal{X}$ with *prior* distribution having density $p(\boldsymbol{x})$, and let $\boldsymbol{x}$ be observed in a representation $\boldsymbol{y} \in \mathcal{Y}$ through a measurement channel which induces a conditional density $p(\boldsymbol{y}|\boldsymbol{x})$, also referred to as the *likelihood function* of $\boldsymbol{x}$. The density of the *posterior* distribution over $\boldsymbol{x}$ is given by the Bayes rule

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

$Z(\boldsymbol{y}) = \int p(\boldsymbol{x}')p(\boldsymbol{y}|\boldsymbol{x}')d\boldsymbol{x}'$ is called the evidence, or model likelihood, or partition function. It is a normalizing factor to make the quantity $p(\boldsymbol{x}|\boldsymbol{y})$ defined above a valid probability density function.

Bayesian inference deals with decision making based on statistics of the posterior distribution. The statistic of choice could differ based on the application. For example, point estimates such as the posterior mean (also called MMSE estimator) $\widehat{\boldsymbol{x}}_{\mathsf{mmse}} = \mathbb{E}[\boldsymbol{x}|\boldsymbol{y}]$, or the posterior mode (also called MAP estimator) $\widehat{\boldsymbol{x}}_{\mathsf{map}} = \underset{\boldsymbol{x}}{\text{argmax}} p(\boldsymbol{x}|\boldsymbol{y})$, or other quantities such as uncertainty reports in the form of conditional variance $\mathbb{V}\left\{\boldsymbol{x}|\boldsymbol{y}\right\}$. These are often of interest in signal processing and machine learning applications. On the the other hand, several applications in computer science, could in general demand marginal densities $\{p(\boldsymbol{x}_\alpha|\boldsymbol{y})\}_\alpha$ of the posterior distribution for some subvectors $\{\boldsymbol{x}_\alpha\}$.

2.3.1 The need for approximate inference

Exact inference of quantities related to the $p(\boldsymbol{x}|\boldsymbol{y})$ is in general difficult or computationally intractable, and several approximation strategies have been proposed in the last few decades. When $\mathcal{X} \subseteq \mathbb{R}^N$ is high dimensional, ie, $N \gtrsim 10^2$ which is typically the case in modern signal processing problems, a major computational challenge in dealing with the posterior distribution explicitly lies in the computation of the partition function $Z(\boldsymbol{y})$. The high dimensional integral is in general computationally intractable, and may require summing over the support of $\mathcal{X}$ which could be exponential in N. This is often colloquially called the *curse of dimensionality.* Hence estimators involving statistics of the posterior distribution often try to circumvent having to evaluate the partition function explicitly.

2.3.2 Approaches to Approximate Inference

One line of attack to approximating inference is via stochastic simulation, also called the Markov chain Monte Carlo (MCMC) methodology, which revolves around obtaining samples $\{\widetilde{\boldsymbol{x}}_i\}_{i=1}^N$ drawn from a *reliable* proxy density $\pi(\boldsymbol{x}) \approx p(\boldsymbol{x}|\boldsymbol{y})$ so as to approximate $\mathbb{E}[g(\boldsymbol{x})|y] \approx$

$\frac{1}{N}\sum_{i=1}^{N} g(\widetilde{\boldsymbol{x}}_i)$.

Other deterministic approaches revolve formulating an appropriate optimization problem the exact solution to which is the quantity to be estimated. Approximating the optimization problem by either approximating the objective function or the constraint set is strategy to perform approximate inference. This circumvent having to sample from a distribution, by approximating $p(\boldsymbol{x}|\boldsymbol{y})$ with an approximate density function.

Variational inference (VI) is an important framework of this type that encapsulates a large body of deterministic strategies for approximate inference. For example, the methods Variational Bayes, Mean Field Approximations, Free Energy minimization and Belief Propagation, Expectation Propagation, and Expectation Consistent Approximate Inference can all be understood as special instances of VI. We briefly review the setup for VI below.

2.3.3 Variational Inference

Much of what has been discussed here can be thought of a summarization of [158], [118] and [15], which are themselves definitive resources on historical developments on this topic.

Variational inference is a deterministic approach which provides a complementary alternative to MCMC based approximation methods for inference in large-scale statistical models. The key idea in variational inference is to generate an approximation to a target density function by considering an optimization problem in the space of density functions, referencing the paradigm of infinite dimensional optimization also called *calculus of variations.* Note that the computation most often happens on a cogent finite dimensional representation of these density spaces. Of particular interest in Bayessian inference is approximating the posterior density $p(\boldsymbol{x}|\boldsymbol{y})$ as discussed in the preamble but the ideas discussed here are more generally about approximating a target density function, say $p^*(\boldsymbol{x})$ which is factorizable as $\prod_\alpha f_\alpha(\boldsymbol{x}_{\partial\alpha})$, as is often the case in probabilistic graphical models.

Let $\mathcal{Q} \subseteq \{q : \mathcal{X} \to \mathbb{R}_+, \int q(\boldsymbol{x})d\boldsymbol{x} = 1\}$ be a subset of valid probability density functions.

Consider the optimization problem below for approximating a density function p^*,

$$q^* \in \operatorname*{argmin}_{q \in \mathcal{Q}} D_{\mathsf{KL}}\big(q \,\|\, p^*\big). \tag{2.1}$$

Owing to the fact that the KL-divergence is a metric, it is always non-negative, and vanishes only when both arguments are identical, (2.1) makes sense, and if $p^* \in \mathcal{Q}$ we have $q^* = p^*$. Otherwise, we get an approximation of p^*, in fact, q^* is a projection of p^* into $\mathcal{Q}$ in the KL-metric sense. We also remark that $q^*(\boldsymbol{x}) = 0$ whenever $p^*(\boldsymbol{x}) = 0$ due to the absolute continuity requirement of the KL-divergence to be finite. We note that the name *variational* inference would perhaps also hold if the metric of choice is one other than the KL-divergence, for example, the Wasserstein family of distances, or even other f-divergences, as considered in [3, 130]. However, the KL-divergence has perhaps received most of the attention from the community, potentially due to the decomposability of the KL-metric for product-factorizable densities, also known as the *chain rule of relative entropy.*

$$D_{\mathsf{KL}}\Big(g_1(\boldsymbol{x}_1)g_2(\boldsymbol{x}_2)\Big\|f_1(\boldsymbol{x}_1)f_2(\boldsymbol{x}_2)\Big) = D_{\mathsf{KL}}\Big(g_1(\boldsymbol{x}_1)\Big\|f_1(\boldsymbol{x}_1)\Big) + D_{\mathsf{KL}}\Big(g_2(\boldsymbol{x}_2)\Big\|f_2(\boldsymbol{x}_2)\Big)$$

This decomposability is amicable for optimization, and can also for analytical purposes. We refer the reader to [118] for a concise review of ideas in variational inference as applied to signal processing applications.

2.4 Denoising

Denoising refers to the problem of recovering $\boldsymbol{x}$ from observations $\boldsymbol{x} + \boldsymbol{w}$, where $\boldsymbol{w}$ is some noise with a distribution $p(\boldsymbol{w})$.

2.4.1 The Proximal Denoiser

For a function $f : \mathbb{R}^N \to \mathbb{R}$, the proximal operator with parameter $\tau \in \mathbb{R}_+$ is given by

$$\text{prox}_f(\boldsymbol{y};\tau) = \underset{\boldsymbol{x}}{\text{argmin}} f(\boldsymbol{x}) + \frac{1}{2\tau}\|\boldsymbol{x}-\boldsymbol{y}\|^2 \tag{2.2}$$

For a proper[1] convex f, $\text{prox}_f(\boldsymbol{y};\tau)$ is unique due to the strong convexity of the objective function.

The probabilistic interpretation for this operator is that it corresponds to the MAP estimator $\widehat{\boldsymbol{x}}_{\mathsf{map}}$, i.e., the mode of the posterior distribution $\boldsymbol{p}(\boldsymbol{x}|\boldsymbol{y})$ corresponding to a prior $\boldsymbol{p}(\boldsymbol{x}) \propto e^{-f(\boldsymbol{x})}$ and AWGN measurements $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{w}$ with $\boldsymbol{w} \sim \mathcal{N}(0, \tau I)$ is additive white gaussian noise. Hence it is often called a *denoiser*. For general f, the uniqueness of f is not guaranteed. However, the $\text{prox}_f(\cdot)$ posses certain computational advantages such as *decomposability*, for e.g.,

$$f(\boldsymbol{x}) = \sum_\alpha f_\alpha(\boldsymbol{x}_\alpha), \qquad \Longrightarrow \qquad [\text{prox}_f(\boldsymbol{y};\tau)]_\alpha = \text{prox}_{f_\alpha}(\boldsymbol{y}_\alpha;\tau).$$

for non-overlapping subvectors $\{\boldsymbol{x}_\alpha\}$ of $\boldsymbol{x}$, which follows readily from the definition of $\text{prox}(\cdot)$ above.

We also note generalizations of the proximal denoiser. For a function $f : \mathbb{R}^K \to \mathbb{R}$, and a symmetric positive semidefinite matrix $\boldsymbol{G} \in \mathbb{R}^{K\times K}_{\succeq 0}$, define

$$\text{prox}_f(\boldsymbol{r};\boldsymbol{G}) := \underset{\boldsymbol{x}}{\text{argmin}} f(\boldsymbol{x}) + \|\boldsymbol{x}-\boldsymbol{r}\|_{\boldsymbol{G}} \tag{2.3}$$

where the norm $\|\boldsymbol{x}-\boldsymbol{r}\|_{\boldsymbol{G}} := \sqrt{(\boldsymbol{x}-\boldsymbol{r})^\top \boldsymbol{G}(\boldsymbol{x}-\boldsymbol{r})}$.

For more general choice of $p(\boldsymbol{w})$, the proximal denoiser is often called the Bregman-proximal operator.

[1] $f(\boldsymbol{x}) > -\infty$ for all $\boldsymbol{x}$, and f has a non-empty effective domain, i.e., $\{\boldsymbol{x} : f(\boldsymbol{x}) < \infty\} \neq \phi$

2.4.2 The MMSE denoiser

Having considered the MAP denoising interpretation of the proximal operator, we look at the MMSE denoiser as well. Consider the mean squared error (MSE) loss, $\ell(\boldsymbol{x}, \widehat{\boldsymbol{x}}) = \|\boldsymbol{x} - \widehat{\boldsymbol{x}}\|^2$. Let $\widehat{\boldsymbol{x}}_{\mathsf{mmse}}$ denote the bayes optimal estimator with respect to the MSE loss, i.e.,

$$\widehat{\boldsymbol{x}}_{\mathsf{mmse}} = f^*, \qquad \text{where} \quad f^*(\boldsymbol{y}) = \underset{f(\boldsymbol{y}):f\in\mathcal{F}}{\operatorname{argmin}} \int_{\mathcal{X}} \|\boldsymbol{x} - f(\boldsymbol{y})\|^2 \boldsymbol{p}(\boldsymbol{x}|\boldsymbol{y}) d\boldsymbol{x}$$

where $\mathcal{F}$ is the set of all measurable functions $\mathcal{Y} \to \mathcal{X}$. Under certain mild assumptions such as existence of the second moment[2], this estimator has another form given by,

$$\widehat{\boldsymbol{x}}_{\mathsf{mmse}}(\boldsymbol{y}) = \mathbb{E}[\boldsymbol{x}|\boldsymbol{y}],$$

owing to the projection interpretation of the conditional expectation. Observe also that the minimum mean squared error is thus the posterior variance, i.e.,

$$\mathsf{MMSE}(\boldsymbol{y}) = \mathbb{V}\{\boldsymbol{x}|\boldsymbol{y}\} = \mathbb{E}[(\boldsymbol{x} - \widehat{\boldsymbol{x}}_{\mathsf{mmse}}(\boldsymbol{y}))^2|\boldsymbol{y}].$$

2.4.3 The LMMSE denoiser

Consider the same optimization problem as above with the constraint set of functions restricted to $\mathcal{F}_L$, the set of linear operators from $\mathcal{Y} \to \mathcal{X}$,

$$\begin{aligned}
\widehat{\boldsymbol{x}}_{\mathsf{lmmse}} &= f^*, &\quad \text{where} \quad && f^*(\boldsymbol{y}) &= \underset{f(\boldsymbol{y}):f\in\mathcal{F}_L}{\operatorname{argmin}} \int_{\mathcal{X}} \|\boldsymbol{x} - f(\boldsymbol{y})\|^2 \boldsymbol{p}(\boldsymbol{x}|\boldsymbol{y}) d\boldsymbol{x} \\
\widehat{\boldsymbol{x}}_{\mathsf{lmmse}} &\equiv \boldsymbol{W}^*, &\quad \text{where} \quad && \boldsymbol{W}^* &= \underset{\boldsymbol{W}\in\mathbb{R}^{N\times M}}{\operatorname{argmin}} \int_{\mathcal{X}} \|\boldsymbol{x} - \boldsymbol{W}\boldsymbol{y}\|^2 \boldsymbol{p}(\boldsymbol{x}|\boldsymbol{y}) d\boldsymbol{x},
\end{aligned}$$

[2]which are implicitly satisfied here since the MSE is meaningless otherwise

Setting the gradient of the above objective w.r.t. $\boldsymbol{W}$ to zero, we get the identity,

$$\boldsymbol{W}^{*}\boldsymbol{y}\boldsymbol{y}^{\top} = \mathbb{E}[\boldsymbol{x}|\boldsymbol{y}]\boldsymbol{y}^{\top}$$

2.4.4 Generalized Proximal Denoiser

For a matrix $\boldsymbol{M} \in \boldsymbol{\mathcal{S}}_{K}^{+}$ and we define the generalized proximal

$$\operatorname{prox}_{f}(\boldsymbol{r}; \boldsymbol{M}) = \underset{\boldsymbol{x}}{\operatorname{argmin}}\ f(\boldsymbol{x}) + \|\boldsymbol{x} - \boldsymbol{r}\|_{\boldsymbol{M}}^{2} \tag{2.4}$$

For the special case of $\boldsymbol{M} = \frac{1}{\tau}\boldsymbol{I}$ we get the standard proximal operator in (2.2)

2.4.5 Mean divergence

For an almost everywhere differentiable operator $\mathbf{g} : \mathbb{R}^{N} \to \mathbb{R}^{N}$, the Jacobian is a matrix valued function denoted $\boldsymbol{J}\mathbf{g}(\boldsymbol{x}) \in \mathbb{R}^{N \times N}$ and defined as $[\boldsymbol{J}\mathbf{g}(\boldsymbol{x})]_{ij} = \frac{\partial g_i}{\partial x_j}\Big|_{\boldsymbol{x}}$. We denote by $\langle \mathbf{g}'(\boldsymbol{x}) \rangle \in \mathbb{R}$ the mean divergence given by

$$\langle \mathbf{g}'(\boldsymbol{x}) \rangle := \frac{1}{N}\operatorname{trace}\left(\boldsymbol{J}\mathbf{g}(\boldsymbol{x})\right) = \frac{1}{N}\sum_{i=1}^{N} \frac{\partial g_i}{\partial x_i}\Big|_{\boldsymbol{x}}$$

Two important mean divergences worth noting are those of the prox_f and the $\mathbb{E}[\boldsymbol{x}|\boldsymbol{y}]$ operators. Note that one can show that the mean divergence for the function $\mathbf{g}(\boldsymbol{y}) := \mathbb{E}[\boldsymbol{x}|\boldsymbol{y}]$ is the average conditional covariance, $\frac{1}{N}\mathbb{V}\{\boldsymbol{x}|\boldsymbol{y}\}$.

2.5 Probabilistic Graphical Models

Graphical models combine two powerful tools of graph theory and probability theory for sophisticated statistical modeling in large-scale systems. The idea is to represent a probability distribution on a graph, such that the structure of the graph captures the essence of the structured randomness of the underlying variables. Let $x_s \in \mathcal{X}$ for all $s \in \{1, 2, \ldots, |V|\} =: V$.

2.5.1 Markov Random Fields (MRF) and Gibbs Fields

For an undirected graph $G = (V, E)$, with vertices $s \in V$ corresponding to variables x_s, and $(s,t) \notin E$ iff

$$x_s \perp\!\!\!\perp x_t \mid x_{V \setminus \{s,t\}}, \tag{2.5}$$

i.e., x_s and x_t are conditionally independent given the other variables in the graph. Any distribution that satisfies this conditional independence relationship is called a *Markov random field* (MRF) of that graph.

Conversely, for a graph G, a way to define a structured probability distribution is a joint density over the variables $\{x_s\}$ which can be written as

$$p(x_1, \ldots, x_{|V|}) \propto \prod_{C \in \mathcal{C}} \psi_C(\boldsymbol{x}_C), \tag{2.6}$$

where $\mathcal{C}$ is the set of all maximal cliques of the graph G. The resulting joint distribution is called a *Gibbs Field* or *Gibbs distribution*. The question regarding whether the two concepts of MRFs and Gibbs fields are equivalent was resolved by the Hammersley-Clifford theorem. They showed that every Gibbs field for a given graph always satisfies the conditional independence relationship (2.5); whereas, if an MRF exists with full support (all configurations on $\mathcal{X}^{|V|}$ are possible), then one can rewrite every MRF over this graph as the product of maximal clique potentials as in (2.6) thus defining a Gibbs field.

2.5.2 Directed Acyclic Graphs (DAG)

DAGs are another form of graphical models where edges $(s \to t) \in E$ are directed. Denote by $\pi(s)$ the parent set of vertex s, i.e., $\pi(s) := \{t \mid (t \to s) \in E\}$. A DAG corresponds to a probability distribution given by

$$p(x_1, \ldots, x_{|V|}) = \prod_{s \in V} p(x_s | \{x_t : t \in \pi(s)\})$$

2.5.3 Factor Graph

For large graphs, the factorization of the joint probability distribution is not easy to visualize from the usual depictions in MRFs or DAGs. Factor graph representations emphasize the factorization of the Gibbs distribution. For a density $p(\boldsymbol{x}) = \prod_\alpha \psi_\alpha(\boldsymbol{x}_{\partial\alpha})$, a factor graph is a representation for q as a bipartite graph $G = (V, F, E)$, with partition F consisting of factor nodes f_α, while the other partition V consists of variable nodes $\boldsymbol{x}_i$. These could be a single variable or a cluster of variables based, and the choice of grouping variables into clusters may result in simpler graphical structures for G, (see [63, Fig. 2]). Here $\partial\alpha = \{i \mid f_\alpha \text{ depends on } \boldsymbol{x}_i\}$. Similarly define $\partial i = \{\alpha \mid f_\alpha \text{ depends on } \boldsymbol{x}_i\}$.

2.6 Large System Limit Analysis: Proportional Asymptotics

We describe the setting for the high dimensional limit under which certain convergence properties related to the AMP algorithms hold. In general for each N, a sequence of problems

$$\boldsymbol{y}(N) = \boldsymbol{A}(N)\boldsymbol{x}(N) + \boldsymbol{\xi}(N)$$

is considered, where $y(N) \in \mathbb{R}^M$ and $\lim_{N\to\infty} \frac{M}{N} \to \delta \in (0, \infty)$. $A(N)$ is drawn from an ensemble of $M \times N$ matrices. We skip the dependence on N wherever it is unambiguous.

For a $p \geq 1$, a transform $f : \mathbb{R}^s \to \mathbb{R}^r$ is said to be *pseudo-Lipschitz continuous of order p* if,

$$\|f(\boldsymbol{a}) - f(\boldsymbol{b})\| \leq L \|\boldsymbol{a} - \boldsymbol{b}\| \left(1 + \|\boldsymbol{a}\|^{p-1} + \|\boldsymbol{b}\|^{p-1}\right), \qquad \text{for some } L > 0$$

We say f is *uniformly Lipschitz continuous* in $\boldsymbol{a}$ at $\bar{c}$ if for some $L_1, L_2, \epsilon_1, \epsilon_2 > 0$ we have,

1. $\|f(\boldsymbol{a}, c) - f(\boldsymbol{a}', c)\| \leq L_1 \|\boldsymbol{a} - \boldsymbol{a}'\|$, for $\boldsymbol{a}, \boldsymbol{a}' \in \operatorname{dom}(f)$ and $|c - \bar{c}| < \epsilon_1$
2. $\|f(\boldsymbol{a}, c_1) - f(\boldsymbol{a}, c_2)\| \leq L_2(1+\|\boldsymbol{a}\|)|c_1 - c_2|$, for $\boldsymbol{a} \in \operatorname{dom}(f)$ and $\max\{|c_1-\bar{c}|, |c_2-\bar{c}|\} < \epsilon_2$

A sequence of vectors $\{\boldsymbol{a}(N)\}$ with growing dimensions is said to *converge empirically with* p^{th} *order moments* to A, denoted in short as $\boldsymbol{a} \overset{PL(p)}{\to} A$, if the following conditions hold:

1. $\mathbb{E}|A|^p < \infty$
2. $\frac{1}{N}\sum_{i=1}^{N} f(a_i) = \mathbb{E}\, f(A)$ for all $f \in C_b \cup \{(\cdot)^p\}$,

where C_b denotes the set of bounded continuous functions. Note that due to the addition of the function $(\cdot)^p$, the above condition is stronger than convergence in distribution, i.e., it holds for a larger class of functions. A consequence of the above two conditions is that if $\boldsymbol{a} \overset{PL(p)}{\to} A$, then for any pseudo-Lipschitz function f of order p, we have $\frac{1}{N}\sum_{i=1}^{N} f(a_i) = \mathbb{E}\, f(A)$. If the components of $\boldsymbol{a}(N)$ are random, then we need additionally that the last equality hold almost surely, since the quantity on the left is random while the quantity on the right is deterministic.

2.6.1 Empirical Convergence of Vector Sequences

Definition 1 (Pseudo-Lipschitz continuity)**.** For a given $p \geq 1$, a function $\mathbf{f} : \mathbb{R}^d \to \mathbb{R}^m$ is called Pseudo-Lipschitz of order p if

$$\begin{aligned} &\|\mathbf{f}(\mathbf{x}_1) - \mathbf{f}(\mathbf{x}_2)\| \\ &\quad \leq C\|\mathbf{x}_1 - \mathbf{x}_2\| \left(1 + \|\mathbf{x}_1\|^{p-1} + \|\mathbf{x}_2\|^{p-1}\right) \end{aligned} \tag{2.7}$$

for some constant $C > 0$.

Observe that for $p = 1$, the pseudo-Lipschitz is equivalent to the standard definition of Lipschitz continuity.

Definition 2 (Uniform Lipschitz continuity)**.** Let $\phi(\mathbf{x}, \theta)$ be a function on $\mathbf{r} \in \boldsymbol{R}^d$ and $\theta \in \boldsymbol{R}^s$. We say that $\phi(\mathbf{x}, \theta)$ is *uniformly Lipschitz continuous* in $\mathbf{x}$ at $\theta = \overline{\theta}$ if there exists constants $L_1, L_2 \geq 0$ and an open neighborhood U of $\overline{\theta}$ such that

$$\|\phi(\mathbf{x}_1, \theta) - \phi(\mathbf{x}_2, \theta)\| \leq L_1 \|\mathbf{x}_1 - \mathbf{x}_2\| \tag{2.8}$$

for all $\mathbf{x}_1, \mathbf{x}_2 \in \boldsymbol{R}^d$ and $\theta \in U$; and

$$\|\phi(\mathbf{x}, \theta_1) - \phi(\mathbf{x}, \theta_2)\| \leq L_2 \left(1 + \|\mathbf{x}\|\right) \|\theta_1 - \theta_2\|, \tag{2.9}$$

for all $\mathbf{x} \in \boldsymbol{R}^d$ and $\theta_1, \theta_2 \in U$.

Definition 3 (Empirical convergence of a sequence)**.** Consider a sequence of vectors $\mathbf{x}(N) = \{\mathbf{x}_n(N)\}_{n=1}^N$ with $\mathbf{x}_n(N) \in \mathbb{R}^d$. So, each $\mathbf{x}(N)$ is a block vector with a total of Nd components. For a finite $p \geq 1$, we say that the vector sequence $\mathbf{x}(N)$ converges empirically with p-th order moments if there exists a random variable $X \in \mathbb{R}^d$ such that

(i) $\mathbb{E}\|X\|_p^p < \infty$; and

(ii) for any $f : \boldsymbol{R}^d \to \boldsymbol{R}$ that is pseudo-Lipschitz continuous of order p,

$$\lim_{N\to\infty} \tfrac{1}{N} \sum_{n=1}^{N} f(\mathbf{x}_n(N)) = \mathbb{E}\left[f(X)\right]. \tag{2.10}$$

In this case, with some abuse of notation, we will write

$$\lim_{N\to\infty} \{\mathbf{x}_n\} \overset{PL(p)}{=} X, \tag{2.11}$$

where we have omitted the dependence on N in $\mathbf{x}_n(N)$. We note that the sequence $\{\mathbf{x}(N)\}$

can be random or deterministic. If it is random, we will require that for every pseudo-Lipschitz function $f(\cdot)$, the limit (A.2) holds almost surely. In particular, if $\mathbf{x}_n \sim X$ are i.i.d. and $\mathbb{E}\|X\|_p^p < \infty$, then $\mathbf{x}$ empirically converges to X with p^{th} order moments.

$PL(p)$ convergence is equivalent to weak convergence plus convergence in p moment [10], and hence $PL(p)$ convergence is also equivalent to convergence in Wasserstein-p metric (See Chapter 6. [157]). We use this fact later in proving Theorem 10.

2.7 Belief propagation

Belief propagation is an algorithm [Pearl, Galager] for obtaining marginal densities of q^* for subvectors $\boldsymbol{x}_i$ without having to solve (2.1) directly. It proceeds by propagating messages on a factor graph.

Sum Product Algorithm

In its standard form, belief propagation appears as the sum-product algorithm,

$$\begin{aligned}\mathcal{M}_{\alpha\to i}(\boldsymbol{x}_i) &= \int f_\alpha(\boldsymbol{x}) \prod_{j\in\partial\alpha\setminus i} \mathcal{M}_{i\to\alpha}(\boldsymbol{x}_j) d\boldsymbol{x}_{\partial\alpha\setminus i} \\ \mathcal{M}_{i\to\alpha}(\boldsymbol{x}_i) &= \prod_{\beta\in\partial i\setminus\alpha} \mathcal{M}_{\beta\to i}(\boldsymbol{x}_i)\end{aligned} \tag{algo: Sum-Product}$$

From these messages, *beliefs* (or unnormalized densities) are computed as

$$\begin{aligned}q(\boldsymbol{x}_i) &= \prod_{\beta\in\partial i} \mathcal{M}_{\beta\to i}(\boldsymbol{x}_i) \\ q(\boldsymbol{x}_\alpha) &= f_\alpha(\boldsymbol{x}) \prod_{j\in\partial\alpha} \mathcal{M}_{i\to\alpha}(\boldsymbol{x}_j)\end{aligned}$$

The sum product algorithm was shown to be exact [120] for computing marginal densities for factor graphs which are forests, i.e., a disjoint union of trees, and converges in a single round of message passing, whereby it is also called sometimes referred to as the *forward-*

backward algorithm. In fact, in the tree case, BP is closely related to a dynamic programming algorithm.

More more general graphs, i.e., graph with loops, the BP algorithm can be applied for a few iterations and has demonstrated remarkable success in communication [48], [30], [146]. However, in general the resulting solution is sub-optimal, see [158, Ex. 4.3], and [101].

Max-Product (or Min-Sum) Algorithm

We note that while belief propagation was introduced for finding marginal densities over a factor graph, one can also consider finding the mode of a factorizable density using a similar message passing approach. Factor graphs also appear in the literature on optimization and distributed optimization. Where an objective function of the form

$$\sum_{\alpha} C_\alpha(\boldsymbol{x}_{\partial\alpha}) + \sum_i C_i(\boldsymbol{x}_i)$$

is to be minimized. The same factor graph can now be used to cut down computation, which again is interpretable as a form of variable elimination for forest graphs. In this case however, the product in the aggregation of messages is replaced with a sum while the integral (or sum) is replaced with a minimization step, which gives the min-sum algorithm.

$$\begin{aligned} \mathcal{J}_{\alpha\to i}(\boldsymbol{x}_i) &= \min_{\boldsymbol{x}_{\partial\alpha\setminus i}} f_\alpha(\boldsymbol{x}) + \sum_{j\in\partial\alpha\setminus i} \mathcal{J}_{i\to\alpha}(\boldsymbol{x}_j) d\boldsymbol{x}_{\partial\alpha\setminus i} \qquad \text{(algo: Min-Sum)} \\ \mathcal{J}_{i\to\alpha}(\boldsymbol{x}_i) &= \sum_{\beta\in\partial i\setminus\alpha} \mathcal{J}_{\beta\to i}(\boldsymbol{x}_i) \end{aligned}$$

From these messages, the costs are computed as

$$q(\boldsymbol{x}_i) = \sum_{\beta \in \partial i} \mathcal{J}_{\beta \to i}(\boldsymbol{x}_i)$$

$$q(\boldsymbol{x}_\alpha) = f_\alpha(\boldsymbol{x}) \sum_{j \in \partial \alpha} \mathcal{J}_{i \to \alpha}(\boldsymbol{x}_j)$$

Belief propagation has also been applied to cases where the factor graph is not necessarily tree structured. This was called generalized or loopy belief propagation (LBP). Originally the LBP applications were restricted to sparse graphs which may have loops in them. [98] talks in detail about the various approximations to understand LBP and the approximation gap. Interestingly, for discrete distributions, max-sum LBP corresponds to an integer programming problem.

LBP when applied to non-sparse, i.e., dense graphs is essentially the basis for the original derivation of the Approximate Message Passing algorithm [33]. See 2.1 for the factor graph used in deriving the AMP.

2.8 Review of Approximate Message Passing algorithms

We are interested in recovering an unknown signal $\boldsymbol{x}^* \in \mathcal{X} \subseteq \mathbb{R}^N$ to be reconstructed from observations $\boldsymbol{y} \in \mathcal{Y} \subseteq \mathbb{R}^M$. The signal is assumed to be drawn from a prior density $p(\boldsymbol{x})$, and the measurement model $\boldsymbol{y} = \mathcal{M}(\boldsymbol{x})$ induces a conditional distribution $p(\boldsymbol{y}|\boldsymbol{x})$, which is the likelihood function of $\boldsymbol{x}$. Consequently by the Bayes rule, the posterior $p(\boldsymbol{x}|\boldsymbol{y}) \propto p(\boldsymbol{x})p(\boldsymbol{y}|\boldsymbol{x})$. In many signal processing applications where Bayesian inference is applied, point estimates are of interest rather than marginal probability densities. We focus on two point estimators, namely, the "Bayes optimal estimator" or $\widehat{\boldsymbol{x}}_{\mathsf{mmse}}$, which correspond to the posterior mean, and the MAP estimator $\widehat{\boldsymbol{x}}_{\mathsf{map}}$ which is a mode of the posterior density.

In the generalized linear model (GLM) for measurements, we are interested in reconstructing an unknown signal $\boldsymbol{x}^* \in \mathcal{X} \subseteq \mathbb{R}^N$ from observations or measurements $\boldsymbol{y} \in \mathcal{Y}$ given by

$$\boldsymbol{y} = \boldsymbol{\phi}(\boldsymbol{h}, \boldsymbol{\xi}), \qquad \boldsymbol{h} = \boldsymbol{A}\boldsymbol{x}^*$$

where $\boldsymbol{\phi} : \mathbb{R}^M \to \mathcal{Y} \subseteq \mathbb{R}^M$ is an *elementwise* function which could in general be non-linear, i.e., $[\boldsymbol{\phi}(\boldsymbol{x})]_i = \phi(x_i)$ for $i \in [n]$; and the matrix operator $\boldsymbol{A} \in \mathbb{R}^{M \times N}$ is called the measurement matrix or design matrix; and $\boldsymbol{h} \in \mathbb{R}^M$ an intermediate variable. The quantity $\boldsymbol{\xi}$ is noise, but is assumed to be coordinatewise drawn from the same distribution iid. We remark that one can think of the GLM as a single layer neural network with bias vector $\mathbf{0}$, weight matrix $\boldsymbol{A}$, and activation ϕ. This can later be extended to multi-layer measurement models. The goal is to estimate $\boldsymbol{x}$ when $\{\boldsymbol{y}, \boldsymbol{A}, \phi\}$ are known. We consider a unified treatment for the $\widehat{\boldsymbol{x}}_{\mathsf{map}}$ and $\widehat{\boldsymbol{x}}_{\mathsf{mmse}}$ estimators.

The randomness in $\boldsymbol{\xi}$ induces a conditional density or likelihood function given by,

$$p(\boldsymbol{y}|\boldsymbol{h}) = \prod_{i=1}^{M} p(y_i|h_i).$$

For example, the randomness in $\boldsymbol{\phi}$ could be due to some quantization noise, or finite numerical precision errors. Here, we introduce methods for reconstruction of $\boldsymbol{x}$ with i.i.d. priors, or equivalently separable penalty functions, whereby

$$p(\boldsymbol{x}) = \prod_{i=1}^{N} p(x_i), \qquad p(u) \propto \exp(-f(u)).$$

However, we later comment on the extensions to non-iid priors.

2.8.1 The Standard Linear Model

The GLM has a special case called the Standard Linear Model (SLM), where $\boldsymbol{\phi}(\boldsymbol{h}, \boldsymbol{\xi}) = \boldsymbol{h} + \boldsymbol{\xi}$ where $\boldsymbol{\xi}$ is additive white gaussian noise. This is perhaps the simplest model that also captures the effect of the *mixing* due to $\boldsymbol{A}$ in the presence of noise. For the SLM $p(\boldsymbol{y}|\boldsymbol{x}) \propto \frac{\nu}{2} \left\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\right\|^2$, where ν is the scalar precision of the noise $\boldsymbol{\xi}$, i.e., $\boldsymbol{\xi} \sim \mathcal{N}(0, I/\nu)$. This model is easier to analyze due to its relation to least squares problems, a class of problems which has arguably received most of the attention in the theory on vector space methods [85, Chap. 4]. The SLM provides incredible insights into the success or failure of a general approximate inference procedure.

In what follows, we consider the likelihood function and prior given by

$$p(\boldsymbol{y}|\boldsymbol{x}) \propto \exp(-\frac{\nu}{2}\left\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\right\|^2), \qquad p(\boldsymbol{x}) \propto \prod_{i=1}^{N} \exp(-f(x_i)),$$

whereby the MAP estimator is the mode of the posterior, or the minimizer of the negative log-posterior given by

$$\widehat{\boldsymbol{x}}_{\mathsf{map}} = \underset{\boldsymbol{x}}{\operatorname{argmin}} \frac{\nu}{2}\left\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\right\|^2 + \sum_{i=1}^{N} f(x_i),$$

whereas the MMSE estimator does not have a closed-form variational expression in general, but is given by

$$\widehat{\boldsymbol{x}}_{\mathsf{mmse}} = \frac{1}{Z(\boldsymbol{y})} \int_{\mathbb{R}^N} \boldsymbol{x} \exp\left(-\frac{\nu}{2}\left\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{x}\right\|^2 - \sum_{i=1}^{N} f(x_i)\right) d\boldsymbol{x}$$

In the context of the SLM, the Approximate Message Passing (AMP) algorithm [33] was introduced to improve the accuracy of fast iterative algorithms in compressive sensing. The fast algorithms based on iterative thresholding offered worse sparsity-undersampling tradeoffs than convex formulations, whereas the latter were computationally expensive. The AMP algorithm demonstrated the ability to achieve the same accuracy in terms of achieving the sparsity-undersampling tradeoff which is achievable by the convex formulations for large iid random Gaussian matrices.

A remarkably fascinating property of the AMP algorithm is that in a certain high dimensional limit, a scalar-valued recursion called the *State Evolution* (SE) given below, can track bulk statistics such as the mean squared error of reconstruction of the iterates,

$$\mathcal{E}(\gamma_k, \boldsymbol{\tau}_k) = \mathbb{E}[g(X + \mathcal{N}(0, \boldsymbol{\tau}_k); \gamma_k)]^2, \qquad X \sim p(X) \propto e^{-f(X)} \tag{2.12a}$$

$$\boldsymbol{\tau}_{k+1} = \nu^{-1} + \frac{N}{M}\mathcal{E}(\gamma_k, \tau_k), \tag{2.12b}$$

and one can show rigourously that $\mathcal{E}(\gamma_k, \boldsymbol{\tau}_k) = \lim_{N\to\infty} \frac{1}{N} \|\widehat{\boldsymbol{x}}_k - \boldsymbol{x}^*\|^2$ a.s., for $\boldsymbol{A}$ drawn from certain random matrix ensembles. The SE is an incredible tool, since it can provide analytical expressions such as the formula [33, eqn. 5] for the boundary of sparse recovery, a quantity of significant interest for which convex formuations do not provide cogent analytical characterizations.

The AMP algorithm derives its name from a loopy belief propagation based derivation over a dense factor graph. We state the AMP algorithm and its briefly describe its derivation. Similar to AMP, based on belief propagation on another factor graph, albeit with vector-valued nodes, the vector approximate message passing (VAMP) algorithm was proposed by [126]. The VAMP algorithm also posses a SE of its own, which holds for a much larger class of random matrices than the i.i.d. Gaussian ensemble. This makes the VAMP attractive as an analytical tool just like the AMP, while being more robust to the failure modes of AMP. The VAMP algorithm is almost equivalent to several other algorithms proposed recently, such as Turbo-signal recovery [87, 138], and expectation consistent (EC) inference [72, 110] with mild differences. We discuss the VAMP algorithm, and its state evolution while contrasting it with the other algorithms similar to it.

2.8.2 Approximate Message Passing algorithm

The AMP algorithm was introduced in [33]. It can be summarized by the following sequence of iterations given in Algorithm 1

It can be derived as loopy belief propagation on the bipartite factor graph shown in Fig. 2.1. For non-sparse matrices $\boldsymbol{A}$, notice that the graph is dense. While performing loopy belief

Algorithm 1 Approximate Message Passing

```
Require: Data $\{\boldsymbol{y}, \boldsymbol{A}\}$, denoiser $\mathbf{g}(\cdot)$, number of iterations $K_{\text{it}}$, denoiser parameter $\{\gamma_k\}_{k=1}^{K_{\text{it}}} > 0$
1: Initialize $\boldsymbol{r}_0 = \mathbf{0}$
2: for $k = 0, 1, \ldots, K_{\text{it}} - 1$ do
3:     $\widehat{\boldsymbol{x}}_k = \mathbf{g}(\boldsymbol{r}_k; \gamma_k)$
4:     $\alpha_k = \langle \mathbf{g}'(\boldsymbol{r}_k; \gamma_k) \rangle$
5:     $\boldsymbol{v}_k = \boldsymbol{y} - \boldsymbol{A}\widehat{\boldsymbol{x}}_k + \frac{N}{M}\alpha_k \boldsymbol{v}_{k-1}$
6:     $\boldsymbol{r}_{k+1} = \widehat{\boldsymbol{x}}_k + \boldsymbol{A}^\top \boldsymbol{v}_k$
7: end for
```

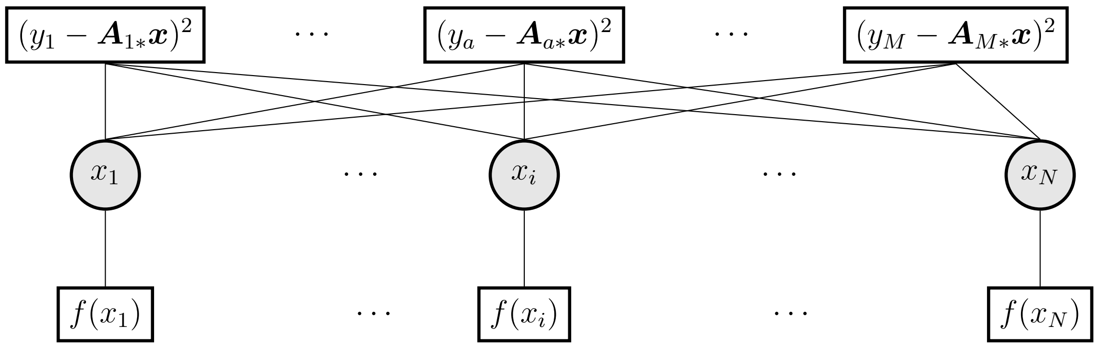

$(y_1 - \boldsymbol{A}_{1*}\boldsymbol{x})^2$
$\cdots$
$(y_a - \boldsymbol{A}_{a*}\boldsymbol{x})^2$
$\cdots$
$(y_M - \boldsymbol{A}_{M*}\boldsymbol{x})^2$
x_1
x_i
x_N
$f(x_1)$
$f(x_i)$
$f(x_N)$

Figure 2.1: Factor graph for the loopy belief propagation derivation of the AMP

propagation on this factor graph, the factor-to-variable messages $\mathcal{M}_{a \to i}$ approximated as Gaussian densities. Intuitively, one can think of the Gaussian approximation as the second order Taylor expansion of the negative log density.

A detailed derivation is provided in [98, Sec 5.2] for MAP inference, which corresponds to max-product (or min-sum) AMP, and in [35] for MMSE inference, which corresponds to sum-product AMP. The only difference lies in the denoisers $\mathbf{g}$ for the two estimators. For the MAP case, the denoiser is the proximal operator of f, whereas for the MMSE case, the denoiser $\mathbf{g}$ is a componentwise extension of the scalar denoiser g given by

$$g(a, c) = \frac{1}{\boldsymbol{Z}(a, c, f)} \int_{\mathbb{R}} x \cdot \exp\left(-f(x) - \frac{c}{2}(x - a)^2\right) dx,$$

where $\boldsymbol{Z}(c, a, f) = \int_{\mathbb{R}} \exp\left(-f(x) - \frac{c}{2}(x - a)^2\right) dx$ is the normalizing factor.

The term $\frac{N}{M}\alpha_k \boldsymbol{v}_{k-1}$ is called the *Onsager correction term.* The idea behind this term is that it causes $\boldsymbol{r}_k$ to be a *noisy* version of $\boldsymbol{x}^*$, i.e.,

$$\boldsymbol{r}_k \approx \boldsymbol{x}^* + \boldsymbol{\tau}_k \boldsymbol{w}, \tag{2.13}$$

where $\boldsymbol{w} \sim \mathcal{N}(0, I)$, (see [98, Fig. 6] for an empirical demonstration). This justifies naming the elementwise operation $\mathbf{g}(\cdot)$ to be a denoiser.

2.8.3 Relation to Iterative First Order Optimization methods

In the absence of the Onsager correction term, i.e., when $\alpha_k = 0$, the MAP version of the AMP algorithm is exactly the proximal gradient descent algorithm. For example, for the ℓ_1 penalized MAP problem, also called the LASSO problem [56] the proximal operator is the soft-thresholding operation [28], [21]. In this case, the AMP algorithm is exactly the iterative shrinkage and thresholding algorithm (ISTA).

An alternative way to understand the Onsager correction term is as a *momentum* term, often used to speed up first order optimization algorithms to achieve the fastest possible rate $O(k^{-2})$ for convex problems [106]. However, in understanding the AMP through this lens, the stepsize in the momentum α_k is *adaptive* and is strongly linked to the curvature of denoiser at its input $\boldsymbol{r}_k$. This adaptation significantly speeds up the convergence, (see the plot [98, Fig. 7]).

In fact, due to the regularized least squares interpretation of the MAP inference problem,

$$\min_x \tfrac{1}{2} \left\| \boldsymbol{y} - \boldsymbol{A}\boldsymbol{x} \right\|^2 + \Phi(\boldsymbol{x}),$$

the properties of the AMP algorithm in solving these problems have also been studied for non-separable convex penalty functions Φ, see [90]

2.8.4 Limitations of AMP

The AMP algorithm is not robust to general measurement matrices $\boldsymbol{A}$. For deviations from the entrywise i.i.d. ensemble, the AMP iterations are known to diverge. We refer the reader to [128] and [20] as well as comparative figures in [126]. The key failure modes of the AMP are that the algorithm does not seem to converge for non-zero mean of $\boldsymbol{A}_{ij}$ and ill-conditioning. Some issues have been addressed by a technique in optimization called damping, see [156].

2.8.5 Vector Approximate Message Passing algorithm

The VAMP algorithm [126], considered a simpler factor graph shown below with vector valued nodes, To describe the message passing derivation briefly, the approximate beliefs at variable

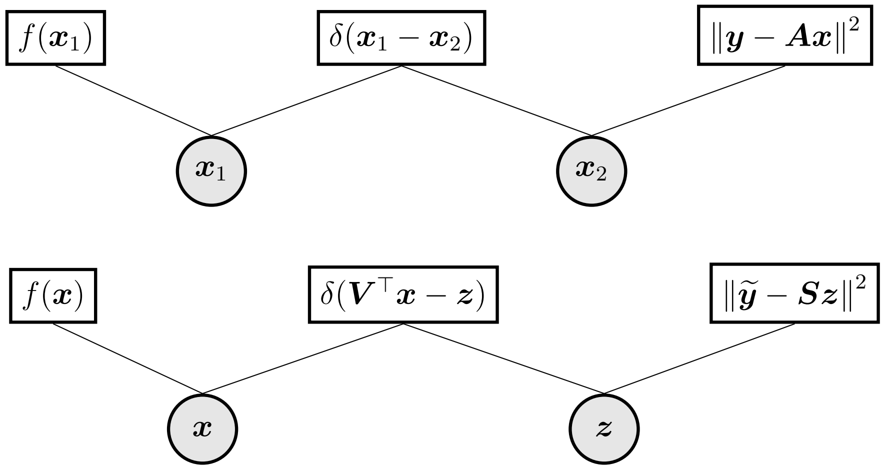

$f(\boldsymbol{x}_1)$
$\delta(\boldsymbol{x}_1 - \boldsymbol{x}_2)$
$\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|^2$
$\boldsymbol{x}_1$
$\boldsymbol{x}_2$
$f(\boldsymbol{x})$
$\delta(\boldsymbol{V}^\top \boldsymbol{x} - \boldsymbol{z})$
$\|\widetilde{\boldsymbol{y}} - \boldsymbol{S}\boldsymbol{z}\|^2$
$\boldsymbol{x}$
$\boldsymbol{z}$

Figure 2.2: (top) Factor graph used in the original derivation of VAMP given in Algorithm 2. (bottom) Alternative factor graph for VAMP. The output-side denoiser for message passing on this factor graph is $\mathbf{G}_2$, whereas it is $\mathbf{g}_2$ for the factor graph on top. This factor graph is implicitly used in analysis in [126]. It is more conducive to extension to GLMs and multi-layer models.

nodes $\boldsymbol{x}_i$ are assumed to be Gaussian with mean and precision parameters $(\widehat{\boldsymbol{x}}_i, \eta_i \boldsymbol{I})$; and the factor-to-variable messages $\mathcal{M}_{\delta \to \boldsymbol{x}_i}(\boldsymbol{x}_i)$ are assumed to be Gaussian with means and precision parameters $(\boldsymbol{r}_i, \gamma_i \boldsymbol{I})$, where η_i, γ_i are scalars. When message passing proceeds, the updates for the parameters for these Gaussian distributions are obtained by moment-matching. The iterations are given in Algorithm 2.

The denoiser $\mathbf{g}_2(\boldsymbol{r}_2;\gamma_2)$ corresponds to the LMMSE denoising. For both the MAP and MMSE case this denoiser is given by

$$\mathbf{g}_2(\boldsymbol{r}_2;\gamma_2) := \underset{\boldsymbol{x}}{\operatorname{argmin}} \frac{\nu}{2} \left\| y - \boldsymbol{A}\boldsymbol{x} \right\|^2 + \frac{\gamma_2}{2} \left\| \boldsymbol{x} - \boldsymbol{r}_2 \right\|^2 = (\nu \boldsymbol{A}^\top \boldsymbol{A} + \gamma_2 \boldsymbol{I})^{-1} (\nu \boldsymbol{A}^\top y + \gamma_2 \boldsymbol{r}_2) \tag{2.14}$$

Using the "full" SVD of $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^\top$ with square matrices $\boldsymbol{U}$ and $\boldsymbol{V}$, and a rectangular matrix $\boldsymbol{S}$, the denoiser $\mathbf{g}_2$ given in (2.15) can be rewritten as

$$\mathbf{g}_2(\boldsymbol{r}_2;\gamma_2) = \boldsymbol{V}\mathbf{G}_2(\boldsymbol{V}^\top \boldsymbol{r}_2;\gamma_2) \tag{2.15a}$$

$$\mathbf{G}_2(\boldsymbol{a};c) := (\nu \boldsymbol{S}^2 + c\boldsymbol{I})^{-1}(\nu \boldsymbol{S}\widetilde{\boldsymbol{y}} + c\boldsymbol{a}). \tag{2.15b}$$

Observe that $\mathbf{G}_2$ is an elementwise operator. This fact is used in the State Evolution analysis of the algorithm later. Since the AWGN is isotropic, we can rewrite $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{\xi}$ as

$$\widetilde{\boldsymbol{y}} = \boldsymbol{S}\boldsymbol{z} + \widetilde{\boldsymbol{\xi}},$$

where $\widetilde{\boldsymbol{y}}$ and $\widetilde{\boldsymbol{\xi}}$ are rotated vectors $\boldsymbol{U}^\top\boldsymbol{y}$ and $\boldsymbol{U}^\top\boldsymbol{\xi}$ respectively, and $\boldsymbol{z} := \boldsymbol{V}^\top\boldsymbol{x}$. Notice that $\boldsymbol{S}$ is an elementwise operator. The posterior for this problem can be given by the factor graph in the bottom panel of Fig 2.2. The resulting approximate message passing on this factor graph is used implicitly during the analysis of VAMP in [126].

2.8.6 Extensions to the Generalized Linear Model

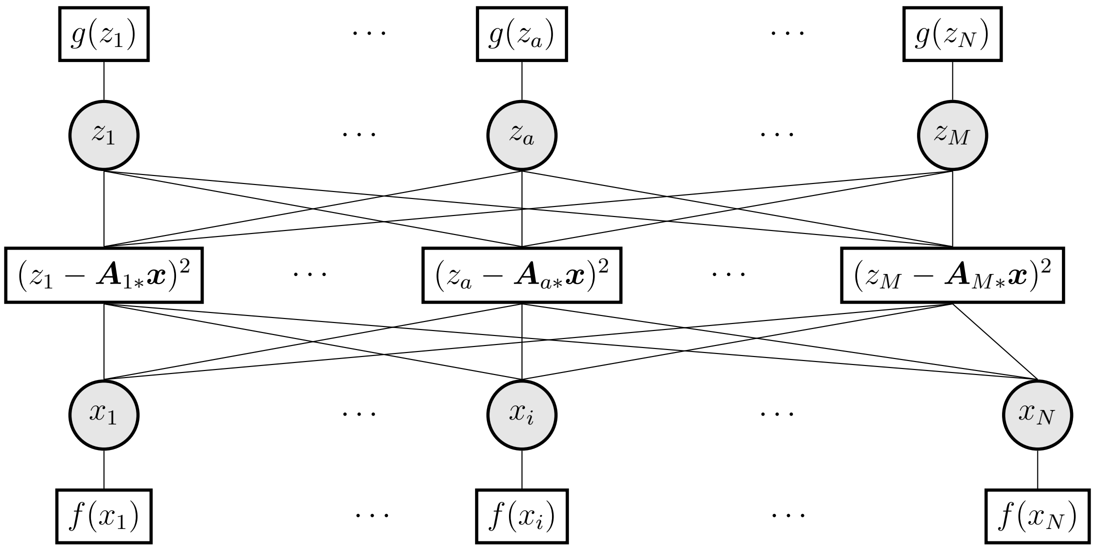
Note that in general, the AWGN may be restrictive since the domain $\mathcal{Y}$ of the observed variable is implicitly assumed to be $\mathbb{R}^M$, it may often not make sense due to the physical constraints of the system, or the interpretation of the observed variables, for e.g., $\mathcal{Y} = \{\text{Yes, No}\}$ indicating binary choice variables, or $\mathcal{Y} = [a, b]$, or $\mathcal{Y} = \{0, 1, 2, \ldots\}$ such as in count data, and so on. Fortunately, much of what applies for the AWGN channel can be extended to other

Algorithm 2 Vector Approximate Message Passing

```
Require: Denoiser $\mathbf{g}_1(\cdot)$, LMMSE denoiser $\mathbf{g}_2(\cdot)$ from (2.15), number of iterations $K_{\text{it}}$
 1: Initialize $\boldsymbol{r}_{10} = \mathbf{0}$, $\gamma_{10} > 0$
 2: for $t = 0, 1, \ldots, K_{\text{it}} - 1$ do
 3:     // Denoising
 4:     $\widehat{\boldsymbol{x}}_{1k} = \mathbf{g}_1(\boldsymbol{r}_{1k}; \gamma_{1k})$
 5:     $\alpha_{1k} = \langle \mathbf{g}_1'(\boldsymbol{r}_{1k}; \gamma_{1k}) \rangle$
 6:     $\eta_{1k} = \gamma_{1k} / \alpha_{1k}$
 7:     $\gamma_{2k} = \eta_{1k} - \gamma_{1k}$
 8:     $\boldsymbol{r}_{2k} = (\eta_{1k}\widehat{\boldsymbol{x}}_{1k} - \gamma_{1k}\boldsymbol{r}_{1k}) / \gamma_{2k}$
 9:
10:     // LMMSE
11:     $\widehat{\boldsymbol{x}}_{2k} = \mathbf{g}_2(\boldsymbol{r}_{2k}; \gamma_{2k})$
12:     $\alpha_{2k} = \langle \mathbf{g}_2'(\boldsymbol{r}_{2k}; \gamma_{2k}) \rangle$
13:     $\eta_{2k} = \gamma_{2k} / \alpha 2k$
14:     $\gamma_{1,k+1} = \eta_{2k} - \gamma_{2k}$
15:     $\boldsymbol{r}_{1,k+1} = (\eta_{2k}\widehat{\boldsymbol{x}}_{2k} - \gamma_{2k}\boldsymbol{r}_{2k}) / \gamma_{1k}$
16: end for
```

exponential family distributions. Keeping in fashion with the nomenclature for the GLM, the prefix *generalized* in this context is indicative of an extension of a signal recovery method to other exponential family distributions. The generalized versions of the previously mentioned iterative algorithms have been considered in GAMP [122], GVAMP [139], GTurbo [84] and GEC [44, 60].

The GAMP algorithm by [122] performs approximate message passing on the factor graph with scalar valued variable nodes shown in the top panel of Fig. 2.3. Note that this is also a bipartite graph. Similarly, the VAMP algorithm was extended to the GLM by [139]. One can show that their algorithm can be derived from the factor graph over vector valued nodes shown in the bottom panel of Fig 2.3. Although the algorithm stated in [139] is more compact, the factor graph stated in the bottom panel of Fig. 2.3 is more conducive to extension to multi-layer models.


$g(z_1)$
$\cdots$
$g(z_a)$
$\cdots$
$g(z_N)$
z_1
z_a
z_M
$(z_1 - \boldsymbol{A}_{1*}\boldsymbol{x})^2$
$(z_a - \boldsymbol{A}_{a*}\boldsymbol{x})^2$
$(z_M - \boldsymbol{A}_{M*}\boldsymbol{x})^2$
x_1
x_i
x_N
$f(x_1)$
$f(x_i)$
$f(x_N)$

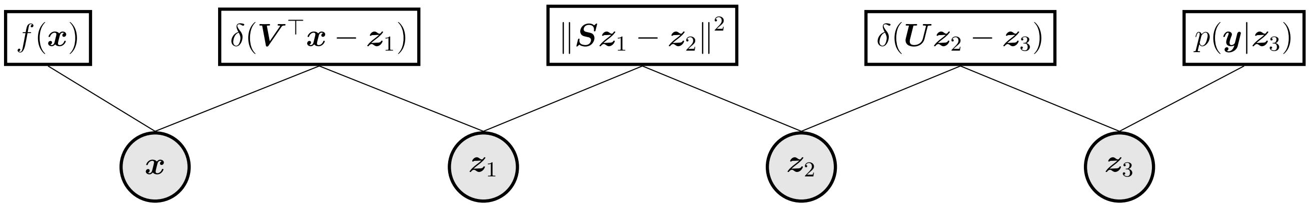

$f(\boldsymbol{x})$
$\delta(\boldsymbol{V}^\top \boldsymbol{x} - \boldsymbol{z}_1)$
$\|\boldsymbol{S}\boldsymbol{z}_1 - \boldsymbol{z}_2\|^2$
$\delta(\boldsymbol{U}\boldsymbol{z}_2 - \boldsymbol{z}_3)$
$p(\boldsymbol{y}|\boldsymbol{z}_3)$
$\boldsymbol{x}$
$\boldsymbol{z}_1$
$\boldsymbol{z}_2$
$\boldsymbol{z}_3$

Figure 2.3: Factor graphs for deriving (top): the GAMP algorithm [122], and (bottom): GVAMP [139] or the more general GEC algorithm [44], [60].

2.8.7 State Evolution of AMP Algorithms

One of the remarkable properties of the Approximate Message Passing algorithms is that certain bulk statistics such as mean squared error of its iterates $\{\widehat{\boldsymbol{x}}_k\}$ can be shown to evolve according to a set of scalar iterations in a certain high dimensional asymptotic setting described in Section 2.6. This scalar iteration is called the state evolution (SE) and is given by equations (2.12).

Specifically, one can show rigorously, that for an average of a pseudo-lipschitz function ψ of order p, the following convergence holds as the dimension $N \to \infty$,

$$\psi(\widehat{\boldsymbol{x}}_k, \boldsymbol{x}^*) = \frac{1}{N}\sum_{i=1}^{N} \psi(\widehat{x}_{ki}^{(t)}, x_i^*) \to \mathbb{E}\psi(\widehat{X}_k; X^*) \quad a.s.$$

For instance in the expression above, we could have $\psi(\widehat{\boldsymbol{x}}, \boldsymbol{x}^*) = \frac{1}{N}\left\|\widehat{\boldsymbol{x}} - \boldsymbol{x}^*\right\|^2$, i.e., the mean squared error or reconstruction, which is pseudo Lipschitz of order 2. Note here that the quantity to which the average converges is the expectation of scalar random variable, where

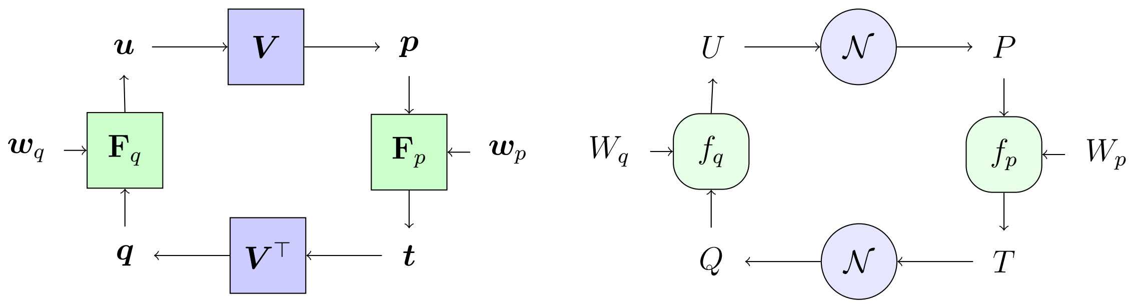
$$X^* \sim p(X) \propto e^{-f(X)}, \qquad \widehat{X}_k = g(X^* + \sqrt{\boldsymbol{\tau}_k}Z; \gamma_k), \quad Z \sim \mathcal{N}(0,1), \qquad Z \perp\!\!\!\perp X^*$$

and $\boldsymbol{\tau}_k$ is given by the scalar recursive equation (2.12).

A very nice intuitive understanding for the state evolution is provided in [33]. In essence, the effect of multiplication by a large i.i.d. Gaussian matrix $\boldsymbol{A}$ and $\boldsymbol{A}^\top$ cause the output to be componentwise i.i.d. Gaussian. Note that this does not necessarily hold if the input to the matrix is correlated with the matrix. The Onsager correction term however, decorrelates the input and allows the Gaussianizing property in equation (2.13) to hold.

We consider the state evolution of the VAMP algorithm due to the ease in exposition. The proof technique used in [126] generalizes the same idea of Bolthausen conditioning from i.i.d. Gaussian matrices $\boldsymbol{A}$ to orthogonally invariant matrices $\boldsymbol{A}$.

2.8.8 Sketch of the proof



$$\alpha_p = \left\langle \mathbf{h}_p'(\boldsymbol{p}, \boldsymbol{w}_p, \gamma_p) \right\rangle \tag{2.16a}$$
$$\gamma_p = \Gamma(\gamma_q, \alpha_p) \tag{2.16b}$$
$$\mathbf{F}_p \equiv C(\alpha_p)\left(\mathbf{h}_p(\boldsymbol{p}, \boldsymbol{w}_p, \gamma_p) - \alpha_p \boldsymbol{p}\right) \tag{2.16c}$$
$$\alpha_q = \left\langle \mathbf{h}_q'(\boldsymbol{q}, \boldsymbol{w}_q, \gamma_q) \right\rangle \tag{2.16d}$$
$$\gamma_q = \Gamma(\gamma_p, \alpha_p) \tag{2.16e}$$
$$\mathbf{F}_q \equiv C(\alpha_q)\left(\mathbf{h}_q(\boldsymbol{q}, \boldsymbol{w}_q, \gamma_q) - \alpha_q \boldsymbol{q}\right) \tag{2.16f}$$

$$\overline{\alpha}_p = \mathbb{E}\, h_p'(P, W_p, \overline{\gamma}_p) \tag{2.17a}$$
$$\overline{\gamma}_p = \Gamma(\overline{\gamma}_q, \overline{\alpha}_q) \tag{2.17b}$$
$$f_p \equiv C(\overline{\alpha}_p)\big(h_p(P, W_p, \overline{\gamma}_p) - \overline{\alpha}_p P\big) \tag{2.17c}$$
$$\overline{\alpha}_q = \mathbb{E}\, h_q'(Q, W_q, \overline{\gamma}_q) \tag{2.17d}$$
$$\overline{\gamma}_q = \Gamma(\overline{\gamma}_p, \overline{\alpha}_p) \tag{2.17e}$$
$$f_q \equiv C(\alpha_q)\big(h_q(Q, W_q, \overline{\gamma}_q) - \overline{\alpha}_q Q\big) \tag{2.17f}$$

Figure 2.4: High dimensional error system in $\mathbb{R}^N$ (left) with an equivalent univariate error system (right). The blocks $\boldsymbol{V}, \boldsymbol{V}^\top$ are pre-multiplications, where $\boldsymbol{V} \in \mathbb{R}^{N\times N}$. The block $\mathcal{N}$ outputs a sample from a Gaussian distribution with the same mean and variance as the input. The functions $\mathbf{F}_p, \mathbf{F}_q$ are componentwise functions based on $\mathbf{h}_p$ and $\mathbf{h}_q$. f_p and f_q are defined using h_p and h_q. $\mathbf{h}_p, \mathbf{h}_q$ are componentwise extensions of h_p, h_q. The quantities $\boldsymbol{w}_p, \boldsymbol{w}_q$ and W_p, W_q are extrinsic inputs.

Consider a dynamical system system shown in the left panel of Fig. 2.4. We skip the dependence on iteration number k, since it is unambiguous. One can show that the VAMP in Algorithm 2 is a special case of these iterations, with $\boldsymbol{p} = \boldsymbol{r}_1 - \boldsymbol{x}^*, \boldsymbol{t} = \boldsymbol{r}_2 - \boldsymbol{x}^*, \boldsymbol{q} = \boldsymbol{V}^\top \boldsymbol{t}$ and $\boldsymbol{u} = \boldsymbol{V}^\top \boldsymbol{p}$, extrinsic quantities $\boldsymbol{w}_p = \boldsymbol{x}^*$ and $\boldsymbol{w}_q = \{\widetilde{\boldsymbol{\xi}}, \boldsymbol{S}\}$, where $\widetilde{\boldsymbol{\xi}} := \boldsymbol{U}^\top \boldsymbol{\xi}$. Recall that $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^\top$ is the full singular value decomposition of $\boldsymbol{A}$, whereby the dynamical system is in $\mathbb{R}^N$. The componentwise functions

$$h_p(p, w_p, \gamma_p) := g_1(p + w_p, \gamma_p) - w_p \tag{2.18a}$$

$$h_q(q, w_q, \gamma_q) := \frac{\nu s \widetilde{\xi} + \gamma_q q}{\nu s^2 + \gamma_q} \tag{2.18b}$$

The parameters $(\alpha_1, \alpha_2, \gamma_1, \gamma_2)$ correspond to $(\alpha_p, \alpha_q, \gamma_p, \gamma_q)$. The following theorem shows that the bulk statistics of the dynamical system in $\mathbb{R}^N$ can be given by expectations of random variables in the scalar dynamical system in the right panel of Fig. 2.4.

Theorem 4. *Assume that*

1. $\boldsymbol{w}_p \overset{PL(2)}{\to} W_p$, $\boldsymbol{w}_q \overset{PL(2)}{\to} W_q$ *and initialization satisfies* $\boldsymbol{u}_0 \overset{PL(2)}{\to} U_0$
2. *Update functions* C, Γ_1, Γ_2 *are continuous.*
3. h_p, h_q, h'_p, h'_q *are uniformly Lipschitz continuous at* $\overline{\alpha}_p, \overline{\alpha}_q, \overline{\gamma}_p, \overline{\gamma}_q$

Then for any fixed k we have

1. $(\boldsymbol{w}_p, \boldsymbol{p}_0, \boldsymbol{p}_1, \ldots, \boldsymbol{p}_k) \overset{PL(2)}{\to} (W_p, P_0, P_1, \ldots, P_k)$, *where* $(P_0, P_1, \ldots, P_k)$ *is a zero mean Gaussian random vector independent of* W_p.
2. $(\boldsymbol{w}_q, \boldsymbol{q}_0, \boldsymbol{q}_1, \ldots, \boldsymbol{q}_k) \overset{PL(2)}{\to} (W_q, Q_0, Q_1, \ldots, Q_k)$, *where* $(Q_0, Q_1, \ldots, Q_k)$ *is a zero mean Gaussian random vector independent of* W_q.
3. $(\alpha_p, \gamma_p) \to (\overline{\alpha}_p, \overline{\gamma}_p)$ *and* $(\alpha_q, \gamma_q) \to (\overline{\alpha}_q, \overline{\gamma}_q)$

The full proof of Theorem 4 has been provided by [126] and [144]. The key idea in proving this result is a trick called Bolthausen conditioning described below.

Bolthausen Conditioning A challenging part in the proof is to evaluate the distribution of $\{p_i\}$ and $\{q_i\}$ in each iteration conditioned on values in preceding iterations which can be expressed as affine equation of $\boldsymbol{V}$, as $\mathcal{K}(\boldsymbol{V}) = \boldsymbol{k}$. It can be shown that the linear, noiseless, and compressed observation of $\boldsymbol{V}$ is equivalent to observing part of the columns in $\boldsymbol{V}$. Since any Haar matrix is unitarily invariant [64], the distribution of $\boldsymbol{V}$ is the same as the original one. Thus, evaluating the conditional distribution of $\boldsymbol{V}$ reduces to analyzing the conditional distribution of a Haar matrix given part of its columns.

Specifically, the effect of $\boldsymbol{V}$ on a vector after k iterations can be separated into two parts, one random and one deterministic. The deterministic part can be calculated exactly in the limit, and the random part can be simulated independently from another rotation matrix $\widetilde{\boldsymbol{V}}$ drawn uniformly from the set $\{\boldsymbol{V} \mid \mathcal{K}(\boldsymbol{V}) = \boldsymbol{k}\}$.

2.8.9 Fixed points of VAMP

In a remarkable works by [161] and [62], a variational principle was given to fixed points of Loopy Belief Propagation algorithms, i.e., it was shown that the fixed points of these iterative update rules correspond to stationary points of a certain constrained energy minimization problem, where the objective function is called the Bethe Free Energy (BFE). See [128], [124] and [20] for a discussion on the fixed points of AMP and GAMP and the discussion the Bethe Free energy.

[110] considered the MMSE problem for the GLM. [44] extended this to the MAP inference problem and gave several generalizations and convergence properties of EC approximate inference. The VAMP algorithm can also be considered as a special case of expectation consistent (EC) inference [110] and [44] with "uniform diagonalization". To be concise, the general EC algorithm allows for the precision matrices of the Gaussians to be arbitrary symmetric positive semidefinite matrices, however VAMP uses only ηI parameterizations. We briefly describe EC below.

Recall the variational optimization problem for exact recovery of the posterior density (2.1) which recovers the posterior density. However, due to the factorization $\boldsymbol{p}(\boldsymbol{x}|\boldsymbol{y}) \propto \boldsymbol{p}(\boldsymbol{x})\boldsymbol{p}(\boldsymbol{y}|\boldsymbol{x})$, the variational problem (2.1) is equivalent to

$$\boldsymbol{q}_i^* = \underset{\boldsymbol{q}_i \in \mathcal{Q}}{\operatorname{argmin}} D_{\mathsf{kl}}\big(\boldsymbol{q}_1 \,\|\, \boldsymbol{p}(\boldsymbol{x})\big) + D_{\mathsf{kl}}\big(\boldsymbol{q}_2 \,\|\, \boldsymbol{p}(\boldsymbol{y}|\boldsymbol{x})\big) + H(\boldsymbol{q}_3) \qquad \text{s.t. } \boldsymbol{q}_1 = \boldsymbol{q}_2 = \boldsymbol{q}_3,$$

using the chain rule for relative entropy. Relaxing the constraints of this optimization problem gives a strategy to approximate inference

$$\boldsymbol{q}_i^* = \underset{\boldsymbol{q}_i \in \mathcal{Q}}{\operatorname{argmin}} D_{\mathsf{kl}}\big(\boldsymbol{q}_1 \,\|\, \boldsymbol{p}(\boldsymbol{x})\big) + D_{\mathsf{kl}}\big(\boldsymbol{q}_2 \,\|\, \boldsymbol{p}(\boldsymbol{y}|\boldsymbol{x})\big) + H(\boldsymbol{q}_3) \tag{EC}$$
$$\text{s.t.} \quad \mathbb{E}_{\boldsymbol{q}_i}\boldsymbol{x} = \widehat{\boldsymbol{x}},\ \mathbb{E}_{\boldsymbol{q}_i}\|\boldsymbol{x}\|^2 = \tau$$

The solution corresponding to the above approximation is called Expectation Consistent (EC) inference, since the density matching conditions are relaxed to moment-matching conditions.

Several other approximate inference algorithms [72], [60], [84] with different intuitions for their derivation have been proposed in the last few years. However, they can all be rewritten as special instances of the GEC algorithm. The derivation provides a variational interpretation to the quantities in the algorithm. Specifically, it enables the interpretation of the quantities $\{\boldsymbol{r}_i, \gamma_i\}$ as scaled versions of Lagrange multipliers of an expectation consistent optimization problem.

Assuming $\gamma_1\boldsymbol{r}_1$ and $\gamma_2\boldsymbol{r}_2$ are Lagrange multipliers for the constraints $\mathbb{E}_{\boldsymbol{q}_1}\boldsymbol{x} = \mathbb{E}_{\boldsymbol{q}_3}\boldsymbol{x}$ and $\mathbb{E}_{\boldsymbol{q}_2}\boldsymbol{x} = \mathbb{E}_{\boldsymbol{q}_3}\boldsymbol{x}$ respectively, whereas γ_1 and γ_2 are multipliers for the constraints $\mathbb{E}_{\boldsymbol{q}_1}\|\boldsymbol{x}\|^2 = \mathbb{E}_{\boldsymbol{q}_3}\|\boldsymbol{x}\|^2$ and $\mathbb{E}_{\boldsymbol{q}_2}\|\boldsymbol{x}\|^2 = \mathbb{E}_{\boldsymbol{q}_3}\|\boldsymbol{x}\|^2$ respectively, the KKT equations for the problem EC give the following conditions

$$\begin{aligned}
\boldsymbol{q}_1^*(\boldsymbol{x}) &\propto \boldsymbol{p}(\boldsymbol{x}) \exp(-\frac{\gamma_1}{2} \|\boldsymbol{x} - \boldsymbol{r}_1\|^2), \\
\boldsymbol{q}_2^*(\boldsymbol{x}) &\propto \boldsymbol{p}(\boldsymbol{y}|\boldsymbol{x}) \exp(-\frac{\gamma_2}{2} \|\boldsymbol{x} - \boldsymbol{r}_2\|^2), \\
\boldsymbol{q}_3^*(\boldsymbol{x}) &\propto \exp(-\frac{\eta}{2} \|\boldsymbol{x} - \widehat{\boldsymbol{x}}\|^2), \\
\mathbb{E}_{\boldsymbol{q}_1^*}\boldsymbol{x} &= \mathbb{E}_{\boldsymbol{q}_2^*}\boldsymbol{x} = \widehat{\boldsymbol{x}}, \\
\mathbb{E}_{\boldsymbol{q}_1^*} \|\boldsymbol{x}\|^2 &= \mathbb{E}_{\boldsymbol{q}_2^*} \|\boldsymbol{x}\|^2 = N\eta^{-1}.
\end{aligned}$$

One can easily show that these are satisfied for any fixed points of the VAMP algorithm with $\widehat{\boldsymbol{x}}_1 = \widehat{\boldsymbol{x}}_2 =: \widehat{\boldsymbol{x}}$ and $\eta_1 = \eta_2 =: \eta$. Thus the VAMP algorithm is a Lagrangian method for reaching first order stationary points of the expectation consistent optimization problem (EC).

2.9 Marchenko-Pastur distribution

The Marchenko-Pastur law appears often in the literature on Approximate Message Passing. This is the distribution of the square of singular values of a random rectangular matrix with i.i.d. entries. Let $\boldsymbol{U} = \boldsymbol{V}_1\boldsymbol{S}\boldsymbol{V}_2$ be such a random rectangular matrix with i.i.d. Gaussian entries, and its singular value decomposition.

We describe the random variable S_{mp} whereby S_{mp}^2 has a rescaled Marchenko-Pastur distribution. Notice that the positive entries of $\mathbf{s}_{\text{mp}}$ are the positive eigenvalues of $\mathbf{U}^\mathsf{T}\mathbf{U}$ (or $\mathbf{U}\mathbf{U}^\mathsf{T}$).

Observe that $U_{ij} \sim N(0, \frac{1}{p})$, whereas, the standard scaling while studying the Marchenko-Pastur distribution is for matrices $\mathbf{H}$ such that $H_{ij} \sim \mathcal{N}(0, \frac{1}{N})$ (for e.g. see equation (1.10) from [151] and the discussion preceding it). Also notice that $\sqrt{\beta}\mathbf{U}$ has the same distribution as $\mathbf{H}$. Thus the results from [151] apply directly to the distributions of eigenvalues of $\beta\mathbf{U}^\mathsf{T}\mathbf{U}$ and $\beta\mathbf{U}\mathbf{U}^\mathsf{T}$. We state their result below taking into account this disparity in scaling.

The positive eigenvalues of $\beta \mathbf{U}^\mathsf{T}\mathbf{U}$ have an empirical distribution which converges to the following density:

$$\mu_\beta(x) = \frac{\sqrt{(b_\beta - x)_+(x-a_\beta)_+}}{2\pi\beta x} \tag{2.19a}$$

$$a_\beta = (1-\sqrt{\beta})^2 \qquad b_\beta := (1+\sqrt{\beta})^2. \tag{2.19b}$$

Similarly the positive eigenvalues of $\beta \mathbf{U}\mathbf{U}^\mathsf{T}$ have an empirical distribution converging to the density $\beta\mu_\beta$. We note the following integral which is useful in our analysis:

$$\begin{aligned} G_0 :&= \lim_{z\to 0^-} \mathbb{E}\frac{1}{S_{\text{mp}}^2 - z}\mathbb{1}_{\{S_{\text{mp}}>0\}} \\ &= \lim_{z\to 0^-}\int_{a_\beta}^{b_\beta} \frac{1}{x/\beta - z}\mu_\beta(x)dx = \frac{\beta}{|\beta-1|}. \end{aligned} \tag{2.20}$$

More generally, the Stieltjes transform of the density is given by:

$$G_{\text{mp}}(z) = \mathbb{E}\frac{1}{S_{\text{mp}}^2 - z}\mathbb{1}_{\{S_{\text{mp}}>0\}} = \int_{a_\beta}^{b_\beta}\frac{1}{x/\beta - z}\mu_\beta(x)dx \tag{2.21}$$

2.9.1 Properties of Marchenko-Pastur Law

If $\boldsymbol{A} \in \mathbb{R}^{M\times N}$ and $a_{ij} \sim \mathcal{N}(0, \frac{1}{N})$ i.i.d., and $\boldsymbol{A} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^\top$ with $\beta = \lim_{N\to\infty}\frac{N}{M}$. If $\{s_i\}_{i=1}^N$ converge in the PL(2) sense to S then $\beta \cdot S^2$ obeys the Marchenko pastur law with density function:

$$\mathbb{P}_{\beta S^2}(t) = \left(1 - \frac{1}{\beta}\right)_+ \delta_0(t) + \frac{\sqrt{(t-a_\beta)_+(b_\beta - t)_+}}{2\pi\beta t} \tag{2.22a}$$

$$\text{where} \qquad a_\beta = (1-\sqrt{\beta})^2 \qquad \text{and} \qquad b_\beta = (1+\sqrt{\beta})^2 \tag{2.22b}$$

Lemma 1.

$$\mathcal{S}(u) = \mathbb{E}\frac{u}{S^2+u} = 1 - \tfrac{1}{4\beta}\left(\sqrt{b_\beta+\beta u} - \sqrt{a_\beta+\beta u}\right)^2 \tag{2.23}$$

$$\mathcal{S}'(u) = \mathbb{E}\frac{S^2}{(S^2+u)^2} = \frac{1}{4}\left(\sqrt{b_\beta+\beta u} - \sqrt{a_\beta+\beta u}\right)\left(\frac{1}{\sqrt{a_\beta+\beta u}} - \frac{1}{\sqrt{b_\beta+\beta u}}\right) \tag{2.24}$$

Proof.

$$\mathcal{S}(u) = \mathbb{E}\frac{1}{\frac{1}{\beta u}\beta S^2+1} \overset{\text{(a)}}{=} 1 - \frac{\mathcal{F}(\frac{1}{\beta u},\beta)}{4\beta\frac{1}{\beta u}} = 1 - \tfrac{u}{4}\mathcal{F}(\tfrac{1}{\beta u},\beta) \tag{2.25}$$

where the LHS of (a) is the $\eta-$transform (see [151, Sec. 2.2.2]). The RHS of (a) is given in [155, pg. 303]:

$$\mathcal{F}(x,z) := \left(\sqrt{x(1+\sqrt{z})^2+1} - \sqrt{x(1-\sqrt{z})^2+1}\right)^2 \tag{2.26}$$

□

Chapter 3

Inference with Deep Generative Models

Deep generative priors offer powerful models for complex-structured data, such as images, audio, and text. Using these priors in inverse problems typically requires estimating the input and/or hidden signals in a multi-layer deep neural network from observation of its output. While these approaches have been successful in practice, rigorous performance analysis is complicated by the non-convex nature of the underlying optimization problems. This paper presents a novel algorithm, Multi-Layer Vector Approximate Message Passing (ML-VAMP), for inference in multi-layer stochastic neural networks. ML-VAMP can be configured to compute maximum a priori (MAP) or approximate minimum mean-squared error (MMSE) estimates for these networks. We show that the performance of ML-VAMP can be exactly predicted in a certain high-dimensional random limit. Furthermore, under certain conditions, ML-VAMP yields estimates that achieve the minimum (i.e., Bayes-optimal) MSE as predicted by the replica method. In this way, ML-VAMP provides a computationally efficient method for multi-layer inference with an exact performance characterization and testable conditions for optimality in the large-system limit.

This work was published in [113, 115] and was coauthoer with Mojtaba Sahraee-Ardakan, Sundeep Rangan, Philip Schniter, and Alyson K. Fletcher

3.1 Introduction

3.1.1 Inference with Deep Generative Priors

We consider inference in an L-layer stochastic neural network of the form

$$\boldsymbol{z}_\ell^0 = \mathbf{W}_\ell \boldsymbol{z}_{\ell-1}^0 + \mathbf{b}_\ell + \boldsymbol{\xi}_\ell, \qquad \ell = 1, 3, \ldots, L-1, \tag{3.1a}$$

$$\boldsymbol{z}_\ell^0 = \phi_\ell(\boldsymbol{z}_{\ell-1}^0, \boldsymbol{\xi}_\ell), \qquad \ell = 2, 4, \ldots, L, \tag{3.1b}$$

where $\boldsymbol{z}_0^0$ is the network input, $\{\boldsymbol{z}_\ell^0\}_{\ell=1}^{L-1}$ are hidden-layer signals, and $\boldsymbol{y} := \boldsymbol{z}_L^0$ is the network output. The odd-indexed layers (4.1a) are (fully connected) affine linear layers with weights $\mathbf{W}_\ell$, biases $\mathbf{b}_\ell$, and additive noise vectors $\boldsymbol{\xi}_\ell$. The even-indexed layers (4.1b) involve separable and possibly nonlinear functions ϕ_ℓ that are randomized[1] by the noise vectors $\boldsymbol{\xi}_\ell$. By "separable," we mean that $[\phi_\ell(\boldsymbol{z}, \boldsymbol{\xi})]_i = \phi_\ell(z_i, \xi_i) \; \forall i$, where ϕ_ℓ is some scalar-valued function, such as a sigmoid or ReLU, and where z_i and ξ_i represent the ith component of $\boldsymbol{z}$ and $\boldsymbol{\xi}$. We assume that the input $\boldsymbol{z}_0^0$ and noise vectors $\boldsymbol{\xi}_\ell$ are mutually independent, that each contains i.i.d. entries, and that the number of layers, L, is even. A block diagram of the network is shown in the top panel of Fig. 3.2. The *inference problem* is to estimate the input and hidden signals $\{\boldsymbol{z}_\ell\}_{\ell=0}^{L-1}$ from an observation of the network output $\boldsymbol{y}$. That is,

$$\text{Estimate } \{\boldsymbol{z}_\ell\}_{\ell=0}^{L-1} \text{ given } \boldsymbol{y},\ \{\mathbf{W}_{2k-1}, \mathbf{b}_{2k-1}, \phi_{2k}\}_{k=1}^{L/2}. \tag{3.2}$$

For inference, we will assume that network parameters (i.e., the weights $\mathbf{W}_\ell$, biases $\mathbf{b}_\ell$, and activation functions ϕ_ℓ) are all known, as are the distributions of the input $\boldsymbol{z}_0^0$ and the noise terms $\boldsymbol{\xi}_\ell$. Hence, we do *not* consider the network learning problem. The superscript "0" on $\boldsymbol{z}_\ell^0$ indicates that this is the "true" value of $\boldsymbol{z}_\ell$, to be distinguished from the estimates of $\boldsymbol{z}_\ell$ produced during inference denoted by $\widehat{\boldsymbol{z}}_\ell$.

[1]The role of the noise $\xi_{\ell,i}$ in ϕ_ℓ is allowed to be generic (e.g., additive, multiplicative, etc.). The relationship between $z_{\ell,i}^0$ and $z_{\ell-1,i}^0$ will be modeled using the conditional density $p(z_{\ell,i}^0 | z_{\ell-1,i}^0) = \int \delta\big(z_{\ell,i}^0 - \phi_\ell(z_{\ell-1,i}^0, \xi_{\ell,i})\big) p(\xi_{\ell,i}) \, d\xi_{\ell,i}$.

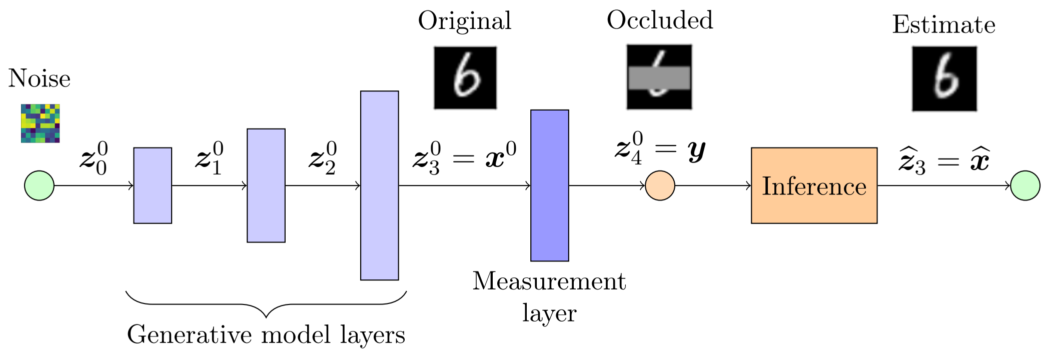


Figure 3.1: Motivating example: Inference for inpainting [16, 163]. An image $\boldsymbol{x}^0$ is modeled as the output of a generative model driven by white noise $\boldsymbol{z}_0^0$, and an occluded measurement $\boldsymbol{y}$ is generated by one additional layer. Inference is then used to recover the image $\boldsymbol{x}$ from the measurement $\boldsymbol{y}$.

The inference problem (3.2) arises in the following state-of-the-art approach to inverse problems. In general, solving an "inverse problem" means recovering some signal $\boldsymbol{x}$ from a measurement $\boldsymbol{y}$ that depends on $\boldsymbol{x}$. For example, in compressed sensing (CS) [38], the measurements are often modeled as $\boldsymbol{y} = \mathbf{A}\boldsymbol{x} + \boldsymbol{\xi}$ with known $\mathbf{A}$ and additive white Gaussian noise (AWGN) $\boldsymbol{\xi}$, and the signal is often modeled as a sparse linear combination of elements from a known dictionary, i.e., $\boldsymbol{x} = \boldsymbol{\Psi}\mathbf{z}$ for some sparse coefficient vector $\mathbf{z}$. To recover $\boldsymbol{x}$, one usually computes a sparse coefficient estimate $\widehat{\mathbf{z}}$ using a LASSO-type convex optimization [148] and then uses it to form a signal estimate $\widehat{\boldsymbol{x}}$, as in

$$\widehat{\boldsymbol{x}} = \boldsymbol{\Psi}\widehat{\mathbf{z}} \quad \text{for} \quad \widehat{\mathbf{z}} = \underset{\mathbf{z}}{\operatorname{argmin}} \tfrac{1}{2}\|\boldsymbol{y} - \mathbf{A}\boldsymbol{\Psi}\mathbf{z}\|^2 + \lambda\|\mathbf{z}\|_1, \tag{3.3}$$

where $\lambda > 0$ is a tunable parameter. The CS recovery approach (3.3) can be interpreted as a *two-layer* version of the inference problem: the first layer implements signal generation via $\boldsymbol{x} = \boldsymbol{\Psi}\mathbf{z}$, while the second layer implements the measurement process $\boldsymbol{y} = \mathbf{A}\mathbf{z} + \boldsymbol{\xi}$. Equation (3.3) then performs maximum a posteriori inference (see the discussion around equation (3.6)) to recover estimates of $\mathbf{z}$ and $\boldsymbol{x}$.

Although CS has met with some success, it has a limited ability to exploit the complex structure of natural signals, such as images, audio, and video. This is because the model

"$\boldsymbol{x} = \boldsymbol{\Psi}\mathbf{z}$ with sparse $\mathbf{z}$" is overly simplistic; it is a *one-layer* generative model. Much more sophisticated modeling is possible with multi-layer priors, as demonstrated in recent works on variational autoencoders (VAEs) [77, 133], generative adversarial networks (GANs) [121, 135], and deep image priors (DIP) [153, 154]. These models have had tremendous success in modeling richly structured data, such as images and text.

A typical application of solving an inverse problem using a deep generative model is shown in Fig. 3.1. This figure considers the classic problem of *inpainting* [14], for which reconstruction with DIP has been particularly successful [16, 163]. Here, a noise-like *innovation* signal $\mathbf{z}_0^0$ drives a three-layer generative network to produce an image $\boldsymbol{x}^0$. The generative network would have been trained on an ensemble of images similar to the one being estimated using, e.g., VAE or GAN techniques. The measurement process, which manifests as occlusion in the inpainting problem, is modeled using one additional layer of the network, which produces the measurement $\boldsymbol{y}$. Inference is then used to recover the image $\boldsymbol{x}^0$ (i.e., the hidden-layer signal $\mathbf{z}_3^0$) from $\boldsymbol{y}$. In addition to inpainting, this deep-reconstruction approach can be applied to other *linear* inverse problems (e.g., CS, de-blurring, and super-resolution) as well as *generalized-linear* [92] inverse problems (e.g., classification, phase retrieval, and estimation from quantized outputs). We note that the inference approach provides an alternative to designing and training a separate reconstruction network, such as in [17, 95, 100].

When using deterministic deep generative models, the unknown signal $\boldsymbol{x}^0$ can be modeled as $\boldsymbol{x}^0 = \mathcal{G}(\mathbf{z}_0^0)$, where $\mathcal{G}$ is a trained deep neural network and $\boldsymbol{z}_0^0$ is a realization of an i.i.d. random vector, typically with a Gaussian distribution. Consequently, to recover $\boldsymbol{x}^0$ from a linear-AWGN measurement of the form $\boldsymbol{y} = \mathbf{A}\boldsymbol{x}^0 + \boldsymbol{\xi}$, the compressed-sensing approach in (3.3) can be extended to a regularized least-squares problem [22] of the form

$$\widehat{\boldsymbol{x}} = \mathcal{G}(\widehat{\boldsymbol{z}}_0), \quad \widehat{\boldsymbol{z}}_0 := \underset{\boldsymbol{z}}{\operatorname{argmin}} \, \tfrac{1}{2} \left\| \boldsymbol{y} - \boldsymbol{A}\mathcal{G}(\boldsymbol{z}) \right\|^2 + \lambda \left\| \boldsymbol{z} \right\|^2 . \tag{3.4}$$

In practice, the optimization in (3.4) is solved using a gradient-based method. This approach can be straightforwardly implemented with deep-learning software packages and has been used, with excellent results, in [16, 53, 73, 97, 142, 150, 163]. The minimization (3.4) has also been useful in interpreting the semantic meaning of hidden signals in deep networks [88, 165]. VAEs [77, 133] and certain GANs [37] can also produce decoding networks that sample from the posterior density, and sampling methods such as Markov-chain Monte Carlo (MCMC) algorithms and Langevin diffusion [23, 160] can also be employed. We note that while the weight matrices in the motivating example in Fig. 3.1 are constant, during analysis we assume that they are instances of random matrices drawn from a general distribution of random matrices.

3.1.2 Analysis via Approximate Message Passing (AMP)

While reconstruction with deep generative priors has seen tremendous practical success, its performance is not fully understood. Optimization approaches such as (3.4) are typically non-convex and difficult to analyze. As we discuss below, most results available today only provide bounds, and these bounds are often be overly conservative (see Section 3.1.4).

Given a network architecture and statistics on the unknown signals, fundamental information-theoretic questions include: What are the precise limits on the accuracy of estimating the hidden signals $\{\mathbf{z}_\ell^0\}_{\ell=0}^{L-1}$ from the measurements $\boldsymbol{y}$? How well do current estimation methods perform relative to these limits? Is it possible to design computationally efficient yet optimal methods?

To answer these questions, this paper considers deep inference via approximate message passing (AMP), a powerful approach for analyzing estimation problems in certain high-dimensional random settings. Since its origins in understanding linear inverse problems in compressed sensing [31, 32], AMP has been extended to an impressive range of estimation and learning tasks, including generalized linear models [123], models with parametric uncertainty [46], structured priors [40], and bilinear problems [137]. For these problems, AMP-based methods have been able to provide computationally efficient algorithms with precise high-dimensional analyses. Often, AMP approaches yield optimality guarantees in cases where all other known approaches do not. See [8] for a detailed discussion on the optimality of AMP.

3.1.3 Main Contributions

In this work, we develop a multi-layer version of AMP for inference in deep networks. The proposed approach builds on the recent vector AMP (VAMP) method of [127], which is itself closely related to expectation propagation (EP) [96, 145], expectation-consistent approximate inference (EC) [45, 111], S-AMP [19], and orthogonal AMP [86]. The proposed method is called *multi-layer VAMP*, or ML-VAMP. As will be described in detail below, ML-VAMP estimates the hidden signals in a deep network by cycling through a set of relatively simple *estimation functions* $\{\mathbf{g}_\ell^\pm\}_{\ell=0}^L$. The information flow in ML-VAMP is shown in the bottom panel of Fig. 3.2. The ML-VAMP method is similar to the multi-layer AMP method of [89] but can handle a more general class of matrices in the linear layers. In addition, as we will describe below, the proposed ML-VAMP algorithm can be configured for either MAP or MMSE estimation. We will call these approaches MAP-ML-VAMP and MMSE-ML-VAMP.

We establish several key results on the ML-VAMP algorithm:

- We show that, for both MAP and MMSE inference, the fixed points of the ML-VAMP algorithm correspond to stationary points of variational formulations of these estimators. This allows the interpretation of ML-VAMP as a Lagrangian algorithm with adaptive step-sizes in both cases. These findings are given in Theorems 5 and 6 and are similar to previous results for AMP [78, 129]. Section 3.3 describes these results.

- We prove that, in a certain large system limit (LSL), the behavior of ML-VAMP is exactly described by a deterministic recursion called the *state evolution* (SE). This SE analysis is a multi-layer extension of similar results [10, 70, 127] for AMP and VAMP. The SE equations enable *asymptotically exact* predictions of macroscopic behaviors of the hidden-layer estimates for *each iteration* of the ML-VAMP algorithm. This allows us to obtain error bounds even if the algorithm is run for a finite number of iterations. The SE analysis, given in Theorem 9, is the main contribution of the paper, and is discussed in Section 3.4.

- Since the original conference versions of this paper [42, 112], formulae for the minimum mean-squared error (MMSE) for inference in deep networks have been conjectured in [7, 47, 131]. As discussed in Section 3.4.3, these formulae are based on heuristic techniques, such as the replica method from statistical physics, and have been rigorously proven in special cases [6, 132]. Remarkably, we show that the mean-squared-error (MSE) of ML-VAMP exactly matches the predicted MMSE in certain cases.

- Using numerical simulations, we verify the predictions of the main result from Theorem 9. In particular, we show that the SE accurately predicts the MSE even for networks that are not considered large by today's standards. We also perform experiments with the MNIST handwritten digit dataset. Here we consider the inference problem using learned networks, for which the weights do not satisfy the randomness assumptions required in our analysis.

In summary, ML-VAMP provides a computationally efficient method for inference in deep networks whose performance can be exactly predicted in certain high-dimensional random settings. Moreover, in these settings, the MSE performance of ML-VAMP can match the existing predictions of the MMSE.

3.1.4 Prior Work

There has been growing interest in studying learning and inference problems in high-dimensional, random settings. One common model is the so-called *wide network*, where the dimensions of the input, hidden layers, and output are assumed to grow with a fixed linear scaling, and the weight matrices are modeled as realizations of random matrices. This viewpoint has been taken in [24, 51, 54, 103], in several works that explicitly use AMP methods [47, 82, 89, 131], and in several works that use closely related random-matrix techniques [108, 141].

The existing work most closely related to ours is that by Manoel et al. [89], which developed a multi-layer version of the original AMP algorithm [31]. The work [89] provides a state-evolution analysis of multi-layer inference in networks with entrywise i.i.d. Gaussian weight matrices. In contrast, our results apply to the larger class of rotationally invariant matrices (see Section 3.4 for details), which includes i.i.d. Gaussian matrices case as a special case.

Several other recent works have also attempted to characterize the performance of reconstruction using deep priors in random settings. For example, when $\boldsymbol{z}_0^0 \in \mathbb{R}^k$ and $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ is a realization of an i.i.d. Gaussian matrix with $m = \Omega(kL \log n)$, Bora et al. [16] showed that an L-layer network $\mathcal{G}$ with ReLU activations can provide provably good reconstruction of $\boldsymbol{x}^0 \in \text{Range}(\mathcal{G})$ from measurements $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}^0 + \boldsymbol{\xi}$. For the same problem, [53] and [65] show that, for $\mathbf{W}_\ell \in \mathbb{R}^{N_\ell \times N_{\ell-1}}$ generated entrywise i.i.d. Gaussian and $N_\ell = \Omega(N_{\ell-1} \log N_{\ell-1})$, one can derive bounds on reconstruction error that hold with high probability under similar conditions on m. Furthermore, they also show that the cost function of (3.4) has stationary points in only two disjoint regions of the $\boldsymbol{z}_0$ space, and both are closely related to the true solution $\boldsymbol{z}_0^0$. In [81], the authors use a layer-wise reconstruction scheme to prove reconstruction error bounds when $N_\ell = \Omega(N_{\ell-1})$, i.e., the network is expansive, but with a constant factor as opposed to the logarithmic factor in [65].

Our results, in comparison, provide an asymptotically exact characterization of the reconstruction error—not just bounds. Moreover, our results hold for arbitrary hidden-dimension ratios $N_\ell/N_{\ell-1}$, which can be less than, equal to, or greater than one. On the other hand, our results hold only in the large-system limit, whereas the other results above hold in the finite-dimensional regime. Nevertheless, we think that it should be possible to derive a finite-dimensional version of our analysis (in the spirit of [134]) that holds with high probability. Also, our experimental results suggest that our large-system-limit analysis is a

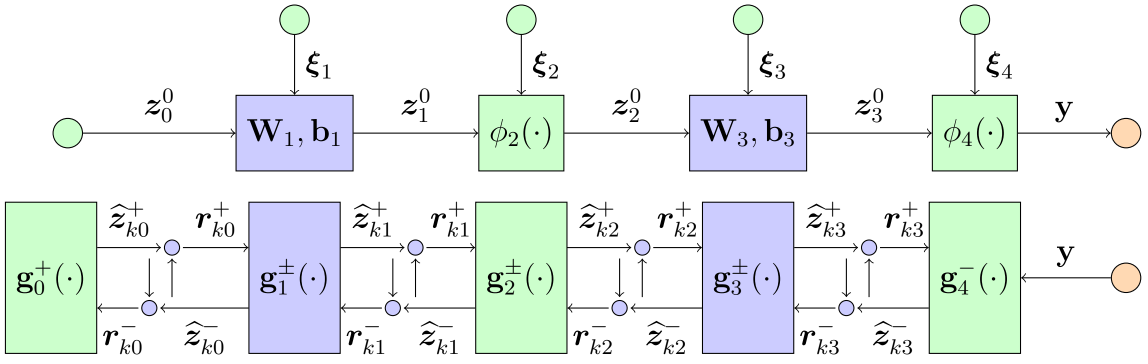


Figure 3.2: Top panel: Feedfoward neural network mapping an input $\boldsymbol{z}_0$ to output $\boldsymbol{y} = \boldsymbol{z}_4^0$ in the case of $L = 4$ layers. Bottom panel: ML-VAMP estimation functions $\mathbf{g}_\ell^\pm(\cdot)$ and estimation quantities $\boldsymbol{r}_{k\ell}^\pm$ and $\widehat{\boldsymbol{z}}_{k\ell}^\pm$ at iteration k.

good approximation of behavior at moderate dimensions.

Some of the material in this paper appeared in conference versions [42, 112], Theorems 5 and 9 were stated in [112], whereas Theorem 8 was stated in [42]. The current paper includes all the proofs, simulation details, and provides a unified treatment of both MAP and MMSE estimation. Additionally, Theorem 6 and its proof are new results.

3.2 Multi-layer Vector Approximate Message Passing

3.2.1 Problem Formulation

We consider inference in a probabilistic setting where, in (4.1), $\mathbf{z}_0^0$ and $\{\boldsymbol{\xi}_\ell\}_{\ell=1}^L$ are modeled as random vectors with known densities. Due to the Markovian structure of $\{\boldsymbol{z}_\ell\}$ in (4.1), the posterior distribution $p(\boldsymbol{z}|\boldsymbol{y})$, where $\boldsymbol{z} := \{\boldsymbol{z}_0\}_{\ell=0}^{L-1}$, factorizes as

$$p(\boldsymbol{z}|\boldsymbol{y}) \propto p(\boldsymbol{z}, \boldsymbol{y}) = p(\boldsymbol{z}, \boldsymbol{z}_L) = p(\boldsymbol{z}_0) \prod_{\ell=1}^{L} p(\boldsymbol{z}_\ell|\boldsymbol{z}_{\ell-1}), \tag{3.5}$$

where the form of $p(\boldsymbol{z}_\ell|\boldsymbol{z}_{\ell-1})$ is determined by $\mathbf{W}_\ell$, $\mathbf{b}_\ell$, and the distribution of $\boldsymbol{\xi}_\ell$ for odd ℓ; and by ϕ_ℓ and the distribution of $\boldsymbol{\xi}_\ell$ for even ℓ. We will assume that $\boldsymbol{z}_\ell \in \mathbb{R}^{N_\ell}$, where N_ℓ can vary across the layers ℓ.

Similar to other graphical-model methods [159], we consider two forms of estimation: MAP estimation and MMSE estimation. The *maximum a priori*, or **MAP**, estimate is defined as

$$\widehat{\boldsymbol{z}}_{\mathsf{map}} := \underset{\boldsymbol{z}}{\operatorname{argmax}}\; p(\boldsymbol{z}|\boldsymbol{y}). \tag{3.6}$$

Although we will focus on MAP estimation, most of our results will apply to general M-estimators [68] of the form,

$$\widehat{\boldsymbol{z}}_{\mathsf{m\text{-}est}} := \underset{\boldsymbol{z}}{\operatorname{argmin}} \left\{ \mathscr{L}_0(\boldsymbol{z}_0) + \sum_{\ell=1}^{L} \mathscr{L}_\ell(\boldsymbol{z}_\ell, \boldsymbol{z}_{\ell-1}) \right\}$$

for loss functions $\mathscr{L}_\ell$. The MAP estimator then corresponds to loss functions $\mathscr{L}_\ell = -\ln p(\boldsymbol{z}_\ell|\boldsymbol{z}_{\ell-1})$ and $\mathscr{L}_0 = -\ln p(\boldsymbol{z}_0)$.

We will also consider the minimum mean-squared error, or **MMSE**, estimate, defined as

$$\widehat{\boldsymbol{z}}_{\mathsf{mmse}} := \mathbb{E}[\boldsymbol{z}|\boldsymbol{y}] = \int \boldsymbol{z}\, p(\boldsymbol{z}|\boldsymbol{y})\, \mathrm{d}\boldsymbol{z}. \tag{3.7}$$

To compute the MMSE estimate, we first compute the posterior marginals $p(\boldsymbol{z}_\ell|\mathbf{y})$. We will also be interested in estimating the posterior marginals $p(\boldsymbol{z}_\ell|\mathbf{y})$. From estimates of the posterior marginals, one can also compute other estimates, such as the mininum mean-absolute error (MMAE) estimate, i.e., the median of the posterior marginal.

3.2.2 The ML-VAMP Algorithm

Similar to the generalized EC (GEC) [45] and generalized VAMP [140] algorithms, the ML-VAMP algorithm attempts to compute MAP or MMSE estimates using a sequence of forward-pass and backward-pass updates. The updates of the algorithm are specified in Algorithm 3. The quantities updated in the forward pass are denoted by superscript $+$, and those updated in the backward pass are denoted by superscript $-$. The notation on lines 11 and 23 means $\langle \partial \boldsymbol{f}(\boldsymbol{x}^*)/\partial \boldsymbol{x} \rangle := \frac{1}{n}\sum_{i=1}^{n} \partial f_i(x_i)/\partial x_i$ evaluated at $\boldsymbol{x} = \boldsymbol{x}^*$, where $\boldsymbol{x} \in \mathbb{R}^n$

Algorithm 3 Multi-layer Vector Approximate Message Passing (ML-VAMP)

```
Require: Estimation functions $\mathbf{g}_0^+$, $\mathbf{g}_L^-$, and $\{\mathbf{g}_\ell^\pm\}_{\ell=1}^{L-1}$.
 1: Set $\boldsymbol{r}_{0\ell}^- = \mathbf{0}$ and initialize $\theta_{0\ell}^-$ for $\ell = 0, 1, \ldots, L-1$.
 2: for $k = 0, 1, \ldots, N_{\text{it}} - 1$ do
 3:     // Forward Pass
 4:     $\widehat{\boldsymbol{z}}_{k0}^+ = \mathbf{g}_0^+(\boldsymbol{r}_{k0}^-, \theta_{k0}^+)$
 5:     $\alpha_{k0}^+ = \langle \partial \mathbf{g}_0^+(\boldsymbol{r}_{k0}^-, \theta_{k0}^+)/\partial \boldsymbol{r}_0^- \rangle$
 6:     $\boldsymbol{r}_{k0}^+ = (\widehat{\boldsymbol{z}}_{k0}^+ - \alpha_{k0}^+ \boldsymbol{r}_{k0}^-)/(1 - \alpha_{k0}^+)$
 7:     for $\ell = 1, \ldots, L-1$ do
 8:         $\widehat{\boldsymbol{z}}_{k\ell}^+ = \mathbf{g}_\ell^+(\boldsymbol{r}_{k\ell}^-, \boldsymbol{r}_{k,\ell-1}^+, \theta_{k\ell}^+)$
 9:         $\alpha_{k\ell}^+ = \langle \partial \mathbf{g}_\ell^+(\boldsymbol{r}_{k\ell}^-, \boldsymbol{r}_{k,\ell-1}^+, \theta_{k\ell}^+)/\partial \boldsymbol{r}_\ell^- \rangle$
10:         $\boldsymbol{r}_{k\ell}^+ = (\widehat{\boldsymbol{z}}_{k\ell}^+ - \alpha_{k\ell}^+ \boldsymbol{r}_{k\ell}^-)/(1 - \alpha_{k\ell}^+)$
11:     end for
12:
13:     // Backward Pass
14:     $\widehat{\boldsymbol{z}}_{k,L-1}^- = \mathbf{g}_L^-(\boldsymbol{r}_{k,L-1}^+, \theta_{kL}^-)$
15:     $\alpha_{k+1,L-1}^- = \langle \partial \mathbf{g}_L^-(\boldsymbol{r}_{k,L-1}^+, \theta_{kL}^-)/\partial \boldsymbol{r}_{k,L-1}^+ \rangle$
16:     $\boldsymbol{r}_{k+1,L-1}^- = (\widehat{\boldsymbol{z}}_{k,L-1}^- - \alpha_{k,L-1}^- \boldsymbol{r}_{k,L-1}^+)/(1 - \alpha_{k,L-1}^-)$
17:     for $\ell = L-1, \ldots, 1$ do
18:         $\widehat{\boldsymbol{z}}_{k,\ell-1}^- = \mathbf{g}_\ell^-(\boldsymbol{r}_{k+1,\ell}^-, \boldsymbol{r}_{k,\ell-1}^+, \theta_{k\ell}^-)$
19:         $\alpha_{k+1,\ell-1}^- = \langle \partial \mathbf{g}_\ell^-(\boldsymbol{r}_{k+1,\ell}^-, \boldsymbol{r}_{k,\ell-1}^+, \theta_{k\ell}^-)/\partial \boldsymbol{r}_{\ell-1}^+ \rangle$
20:         $\boldsymbol{r}_{k+1,\ell-1}^- = (\widehat{\boldsymbol{z}}_{k,\ell-1}^- - \alpha_{k,\ell-1}^- \boldsymbol{r}_{k,\ell-1}^+)/(1 - \alpha_{k,\ell-1}^-)$
21:     end for
22: end for
```

and $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^n$ acts componentwise. The update formulae can be derived similar to those for the GEC algorithm [45], using expectation-consistent approximations of the Gibbs free energy inspired by [111].

The ML-VAMP algorithm splits the estimation of $\boldsymbol{z} = \{\boldsymbol{z}_\ell\}_{\ell=0}^{L-1}$ into smaller problems that are solved by the *estimation functions* $\{\mathbf{g}_\ell^\pm\}_{\ell=1}^{L-1}$, $\mathbf{g}_0^+$ and $\mathbf{g}_L^-$. (See Figure 3.2, bottom panel.) As described below, the form of $\mathbf{g}_\ell^\pm$ depends on whether the goal is MAP or MMSE estimation. During the forward pass, the estimators $\mathbf{g}_\ell^+$ are invoked, whereas in the backward pass, $\mathbf{g}_\ell^-$ are invoked. Similarly, the ML-VAMP algorithm maintains two copies, $\widehat{\boldsymbol{z}}^+$ and $\widehat{\boldsymbol{z}}^-$, of the estimate of $\boldsymbol{z}$. For $\ell = 1, 2, \ldots, L-1$, each pair of estimators $(\mathbf{g}_\ell^+, \mathbf{g}_\ell^-)$ takes as input $\boldsymbol{r}_{\ell-1}^+$ and $\boldsymbol{r}_\ell^-$ to update the estimates $\widehat{\boldsymbol{z}}_\ell^+$ and $\widehat{\boldsymbol{z}}_{\ell-1}^-$, respectively. Similarly, $\mathbf{g}_0^+$ and $\mathbf{g}_L^-$ take inputs $\boldsymbol{r}_0^-$ and $\boldsymbol{r}_{L-1}^+$ to update $\widehat{\boldsymbol{z}}^0$ and $\widehat{\boldsymbol{z}}_{L-1}^-$, respectively. The estimation functions also take parameters $\theta_\ell^\pm$.

3.2.3 MAP and MMSE Estimation Functions: $\{\mathbf{g}_\ell^+\}$

The ML-VAMP algorithm is an iterative application of estimation functions $\mathbf{g}_\ell^\pm$ which take as input $(\mathbf{r}_\ell^-, \boldsymbol{r}_{\ell-1}^+)$ and output $(\widehat{\mathbf{z}}_\ell^+, \widehat{\mathbf{z}}_{\ell-1}^-)$. During the forward pass the output $\widehat{\mathbf{z}}_{\ell-1}^-$ is dropped whereas in the backward pass $\widehat{\mathbf{z}}_\ell^+$ is dropped. These estimation functions can take arbitrary parametric forms.

The form of the estimation functions $\{\mathbf{g}_\ell^\pm\}_{\ell=0}^{L-1}$ depends on whether the goal is to perform MAP or MMSE estimation. In either case, we restrict ourselves to the following parameterization

$$\begin{aligned} &\theta_{k0}^+ = \gamma_{k0}^-, \qquad \theta_{k\ell}^+ = (\gamma_{k\ell}^-, \gamma_{k,\ell-1}^+), \\ &\theta_{kL}^- = \gamma_{k,L-1}^+ \quad \theta_{k\ell}^- = (\gamma_{k+1,\ell}^-, \gamma_{k,\ell-1}^+), \end{aligned} \tag{3.8}$$

where $\gamma_{k\ell}^\pm$ and $\eta_{k\ell}^\pm$ are scalars updated at iteration $k \geq 0$ and all $\ell = 0, 1, \ldots, L-1$ as follows:

$$\begin{aligned} &\gamma_{k\ell}^+ = \eta_{k\ell}^+ - \gamma_{k\ell}^-, \qquad \gamma_{k+1,\ell}^- = \eta_{k+1,\ell}^- - \gamma_{k\ell}^+, \\ &\eta_{k\ell}^+ = \gamma_{k\ell}^-/\alpha_{k\ell}^+ \qquad \eta_{k+1,\ell}^- = \gamma_{k\ell}^+/\alpha_{k+1,\ell}^-, \end{aligned} \tag{3.9}$$

while the updates of $\alpha_{k\ell}^\pm$ are explicitly given in lines 11 and 23 of Algorithm 3. The parameters $\gamma_{k\ell}^\pm$ and $\eta_{k\ell}^\pm$ respectively, represent estimates for precision (inverse variance) of the input $\mathbf{r}_{k\ell}^\pm$ and output $\widehat{\mathbf{z}}_{k\ell}^\pm$ to the estimation functions $\mathbf{g}_\ell^\pm$. They can also be interpreted as surrogates for curvature information (or second-order information) of the loss function. The quantities $\alpha_{k\ell}^\pm \in (0, 1)$ couple the forward and backward iterations via the so-called *Onsager correction* terms in line 12 and 24.

Given these parameters, both the MAP and MMSE estimation functions are defined from the *belief* function over $(\boldsymbol{z}_\ell, \boldsymbol{z}_{\ell-1})$:

$$\begin{aligned} b_\ell(\boldsymbol{z}_\ell, \boldsymbol{z}_{\ell-1}|\boldsymbol{r}_\ell^-, \boldsymbol{r}_{\ell-1}^+, \gamma_\ell^-, \gamma_{\ell-1}^+) &\propto p(\boldsymbol{z}_\ell|\boldsymbol{z}_{\ell-1})\times \\ &\exp\left(-\tfrac{\gamma_\ell^-}{2}\left\|\boldsymbol{z}_\ell - \boldsymbol{r}_\ell^-\right\|^2 - \tfrac{\gamma_{\ell-1}^+}{2}\left\|\boldsymbol{z}_{\ell-1} - \boldsymbol{r}_{\ell-1}^+\right\|^2\right) \end{aligned} \tag{3.10}$$

for $\ell = 1, 2, \ldots, L-1$. Similarly, $b_L(\boldsymbol{z}_L, \boldsymbol{z}_{L-1}) \propto p(\boldsymbol{y}|\boldsymbol{z}_{L-1})\exp(-\frac{\gamma_{L-1}^+}{2}\|\boldsymbol{z}_{L-1} - \boldsymbol{r}_{L-1}^+\|^2)$, and $b_0(\boldsymbol{z}_0, \boldsymbol{z}_{-1}) \propto p(\boldsymbol{z}_0)\exp(-\frac{\gamma_0^-}{2}\|\boldsymbol{z}_0 - \boldsymbol{r}_0^-\|^2)$. When performing MMSE inference, we use

$$\begin{aligned} (\widehat{\boldsymbol{z}}_\ell^+, \widehat{\boldsymbol{z}}_{\ell-1}^-)_{\mathsf{mmse}} &= \mathbf{g}_{\ell,\mathsf{mmse}}^\pm(\boldsymbol{r}_\ell^-, \boldsymbol{r}_{\ell-1}^+; \gamma_\ell^-, \gamma_{\ell-1}^+) \\ &= \mathbb{E}[(\boldsymbol{z}_\ell, \boldsymbol{z}_{\ell-1})|b_\ell], \end{aligned} \tag{3.11}$$

where $\mathbb{E}[\cdot|b_\ell]$ denotes expectation with respect to the (normalized) distribution b_ℓ. Similarly, for MAP inference, we use

$$\begin{aligned} (\widehat{\boldsymbol{z}}_\ell^+, \widehat{\boldsymbol{z}}_{\ell-1}^-)_{\mathsf{map}} &= \mathbf{g}_{\ell,\mathsf{map}}^\pm(\boldsymbol{r}_\ell^-, \boldsymbol{r}_{\ell-1}^+; \gamma_\ell^-, \gamma_{\ell-1}^+) \\ &= \underset{\boldsymbol{z}_\ell, \boldsymbol{z}_{\ell-1}}{\operatorname{argmax}}\ b_\ell(\boldsymbol{z}_\ell, \boldsymbol{z}_{\ell-1}). \end{aligned} \tag{3.12}$$

Notice that (3.12) corresponds to the proximal operator of $-\ln p(\boldsymbol{z}_\ell|\boldsymbol{z}_{\ell-1})$. We will use "MMSE-ML-VAMP" to refer to ML-VAMP with the MMSE estimation functions (3.11), and "MAP-ML-VAMP" to refer to ML-VAMP with the MAP estimation functions (3.12).

3.2.4 Computational Complexity

A key feature of the ML-VAMP algorithm is that, for the neural network (4.1), the MMSE and MAP estimation functions (3.11) and (3.12) are computationally easy to compute. To see why, first recall that, for the even layers $\ell = 2, 4, \ldots L$, the map ϕ_ℓ in (4.1b) is assumed separable and the noise $\boldsymbol{\xi}_\ell$ is assumed i.i.d. As a result, $\boldsymbol{z}_\ell$ is conditionally independent given $\boldsymbol{z}_{\ell-1}$, i.e., $p(\boldsymbol{z}_\ell|\boldsymbol{z}_{\ell-1}) = \prod_i p(z_{\ell,i}|z_{\ell-1,i})$. Thus, for even ℓ, the belief function b_ℓ in (4.12) also factors into a product of the form $b_\ell(\boldsymbol{z}_\ell, \boldsymbol{z}_{\ell-1}) = \prod_i b_\ell(z_{\ell,i}, z_{\ell-1,i})$, implying that the MAP and MMSE versions of $\mathbf{g}_\ell^\pm$ are both coordinate-wise separable. In other words, the MAP and MMSE estimation functions can be computed using N_ℓ scalar MAP or MMSE estimators.

Next consider (4.1a) for $\ell = 1, 3, \ldots, L-1$, i.e., the linear layers. Assume that $\boldsymbol{\xi}_\ell \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}\nu_\ell^{-1})$ for some precision (i.e., inverse variance) $\nu_\ell > 0$. Then $p(\boldsymbol{z}_\ell|\boldsymbol{z}_{\ell-1}) \propto \frac{\nu_\ell}{2}\left\|\boldsymbol{z}_\ell - \boldsymbol{W}_\ell \boldsymbol{z}_{\ell-1} - \mathbf{b}_\ell\right\|^2$. In this case, the MMSE and MAP estimation functions (3.11) and (3.12) are identical, and both take the form of a standard least-squares problem. Similar to the VAMP algorithm [127], the least-squares solution—which must be recomputed at each iteration k—is can be efficiently computed using a single singular value decomposition (SVD) that is computed once, before the iterations begin. In particular, we compute the SVD

$$\boldsymbol{W}_\ell = \boldsymbol{V}_\ell \operatorname{Diag}(\boldsymbol{s}_\ell)\boldsymbol{V}_{\ell-1}, \tag{3.13}$$

where $\boldsymbol{V}_\ell \in \mathbb{R}^{N_\ell \times N_\ell}$ and $\boldsymbol{V}_{\ell-1} \in \mathbb{R}^{N_{\ell-1}\times N_{\ell-1}}$ are orthogonal and $\operatorname{Diag}(\boldsymbol{s}_\ell) \in \mathbb{R}^{N_\ell \times N_{\ell-1}}$ is a diagonal matrix that contains the singular values of $\boldsymbol{W}_\ell$. Let $\overline{\mathbf{b}}_\ell := \boldsymbol{V}_\ell^\top \mathbf{b}_\ell$. Then for odd ℓ, the updates (3.11) and (3.12) both correspond to quadratic problems, which can be simplified by exploiting the rotational invariance of the ℓ_2 norm. Specifically, one can derive that

$$\begin{aligned}
\widehat{\boldsymbol{z}}_\ell^+ &= \mathbf{g}_\ell^+(\boldsymbol{r}_\ell^-, \boldsymbol{r}_{\ell-1}^+, \gamma_\ell^-, \gamma_{\ell-1}^+) \\
&= \mathbf{V}_\ell \mathbf{G}_\ell^+(\mathbf{V}_\ell^\top \boldsymbol{r}_\ell^-, \mathbf{V}_{\ell-1}\boldsymbol{r}_{\ell-1}^+, \overline{\mathbf{s}}_\ell, \overline{\mathbf{b}}_\ell, \gamma_\ell^-, \gamma_{\ell-1}^+), \qquad (3.14a) \\
\widehat{\boldsymbol{z}}_{\ell-1}^- &= \mathbf{g}_\ell^-(\boldsymbol{r}_\ell^-, \boldsymbol{r}_{\ell-1}^+, \gamma_\ell^-, \gamma_{\ell-1}^+) \\
&= \mathbf{V}_{\ell-1}^\top \mathbf{G}_\ell^-(\mathbf{V}_\ell^\top \boldsymbol{r}_\ell^-, \mathbf{V}_{\ell-1}\boldsymbol{r}_{\ell-1}^+, \overline{\mathbf{s}}_\ell, \overline{\mathbf{b}}_\ell, \gamma_\ell^-, \gamma_{\ell-1}^+), \qquad (3.14b)
\end{aligned}$$

where *transformed denoising functions* $\mathbf{G}_\ell^\pm(\cdot)$ are componentwise extensions of $G_\ell^\pm(\cdot)$, defined as

$$\begin{bmatrix} G_\ell^+ \\ G_\ell^- \end{bmatrix} = \begin{bmatrix} -\nu_\ell s_\ell & \gamma_\ell^- + \nu_\ell \\ \gamma_{\ell-1}^+ + \nu_\ell s_\ell^2 & -\nu_\ell s_\ell \end{bmatrix}^{-1} \begin{bmatrix} \gamma_\ell^- u_\ell + \nu_\ell \bar{b}_\ell \\ \gamma_{\ell-1}^+ u_{\ell-1} - \nu_\ell s_\ell \bar{b}_\ell \end{bmatrix} \tag{3.15}$$

Note that G_ℓ^+ and G_ℓ^- are functions which take inputs $(u_\ell, u_{\ell-1}, s_\ell, \bar{b}_\ell, \gamma_\ell^-, \gamma_{\ell-1}^+)$ and output the expressions on the RHS. A detailed derivation of equations (3.14) and (3.15) is given in [41, Appendix B]. Note that the argument $\overline{\boldsymbol{s}}_\ell$ in (3.14a) is N_ℓ dimensional, whereas in (3.14b) it is $N_{\ell-1}$ dimensional, i.e., appropriate zero-padding is applied. Keeping this subtlety in mind, we use $\overline{\boldsymbol{s}}_\ell$ to keep the notation simple.

From Algorithm 3, we see that each pass of the MAP-ML-VAMP or MMSE-ML-VAMP algorithm requires solving (a) scalar MAP or MMSE estimation problems for the non-linear, separable layers; and (b) least-squares problems for the linear layers. In particular, no high-dimensional integrals or high-dimensional optimizations are involved.

3.3 Fixed Points of ML-VAMP

Our first goal is to characterize the fixed points of Algorithm 3. To this end, let $\boldsymbol{r}_\ell^+, \boldsymbol{r}_\ell^-, \widehat{\boldsymbol{z}}_\ell$ with parameters $\alpha_\ell^+, \alpha_\ell^-, \gamma_\ell^+, \gamma_\ell^-, \eta_\ell$ be a fixed point of the ML-VAMP algorithm, where we have dropped the iteration subscript k. At a fixed point, we do not need to distinguish between $\widehat{\boldsymbol{z}}_\ell^+$ and $\widehat{\boldsymbol{z}}_\ell^-$, nor between η_ℓ^+ and η_ℓ^-, since the updates in (3.9) imply that

$$\begin{aligned} &\eta_\ell^+ = \eta_\ell^- = \gamma_\ell^+ + \gamma_\ell^- =: \eta_\ell, \\ &\alpha_\ell^+ = \tfrac{\gamma_\ell^-}{\eta_\ell}, \quad \alpha_\ell^- = \tfrac{\gamma_\ell^+}{\eta_\ell}, \quad \text{and} \quad \alpha_\ell^+ + \alpha_\ell^- = 1. \end{aligned} \tag{3.16}$$

Applying these relationships to lines 12 and 24 of Algorithm 3 gives

$$\widehat{\boldsymbol{z}}_\ell^+ = \widehat{\boldsymbol{z}}_\ell^- = \frac{\gamma_\ell^+ \boldsymbol{r}_\ell^+ + \gamma_\ell^- \boldsymbol{r}_\ell^-}{\gamma_\ell^+ + \gamma_\ell^-} =: \widehat{\boldsymbol{z}}_\ell. \tag{3.17}$$

3.3.1 Fixed points of MAP-ML-VAMP and connections to ADMM

Our first results relates the MAP-ML-VAMP updates to an ADMM-type minimization of the MAP objective (3.6). For this we use *variable splitting*, where we replace each variable $\mathbf{z}_\ell$ with two copies, $\mathbf{z}_\ell^+$ and $\mathbf{z}_\ell^-$. Then, we define the objective function

$$F(\mathbf{z}^+, \mathbf{z}^-) := -\ln p(\mathbf{z}_0^+) - \sum_{\ell=1}^{L-1} \ln p(\mathbf{z}_\ell^+ | \mathbf{z}_{\ell-1}^-) - \ln p(\mathbf{y} | \mathbf{z}_{L-1}^-)$$

over the variable groups $\mathbf{z}^+ := \{\mathbf{z}_\ell^+\}_{\ell=1}^{L-1}$ and $\mathbf{z}^- := \{\mathbf{z}_\ell^-\}_{\ell=1}^{L-1}$. The optimization (3.6) is then equivalent to

$$\min_{\mathbf{z}^+, \mathbf{z}^-} F(\mathbf{z}^+, \mathbf{z}^-) \quad \text{s.t.} \quad \mathbf{z}_\ell^+ = \mathbf{z}_\ell^-, \quad \forall\, \ell = 0, \dots, L-1. \tag{3.18}$$

Corresponding to this constrained optimization, we define the augmented Lagrangian

$$\begin{aligned}\mathcal{L}(\mathbf{z}^+, \mathbf{z}^-, \mathbf{s}) &= F(\mathbf{z}^+, \mathbf{z}^-) \\ &\quad + \sum_{\ell=0}^{L-1} \eta_\ell \mathbf{s}_\ell^\mathsf{T}(\mathbf{z}_\ell^+ - \mathbf{z}_\ell^-) + \frac{\eta_\ell}{2}\|\mathbf{z}_\ell^+ - \mathbf{z}_\ell^-\|^2, \end{aligned} \tag{3.19}$$

where $\mathbf{s} := \{\mathbf{s}_\ell\}$ is a set of dual parameters, $\gamma_\ell^\pm > 0$ are weights, and $\eta_\ell = \gamma_\ell^+ + \gamma_\ell^-$. Now, for $\ell = 1, \ldots, L-2$, define

$$\begin{aligned}\mathcal{L}_\ell(\mathbf{z}_{\ell-1}^-, \mathbf{z}_\ell^+; \mathbf{z}_{\ell-1}^+, \mathbf{z}_\ell^-, \mathbf{s}_{\ell-1}, \mathbf{s}_\ell) &:= -\ln p(\mathbf{z}_\ell^+|\mathbf{z}_{\ell-1}^-) + \eta_\ell \mathbf{s}_\ell^\mathsf{T} \mathbf{z}_\ell^+ \\ - \eta_{\ell-1}\mathbf{s}_{\ell-1}^\mathsf{T}\mathbf{z}_{\ell-1}^- &+ \tfrac{\gamma_{\ell-1}^+}{2}\|\mathbf{z}_{\ell-1}^- - \mathbf{z}_{\ell-1}^+\|^2 + \tfrac{\gamma_\ell^-}{2}\|\mathbf{z}_\ell^+ - \mathbf{z}_\ell^-\|^2,\end{aligned}$$

which represents the terms in the Lagrangian $\mathcal{L}(\cdot)$ in (3.19) that contain $\mathbf{z}_{\ell-1}^-$ and $\mathbf{z}_\ell^+$. Similarly, define $\mathcal{L}_0(\cdot)$ and $\mathcal{L}_{L-1}(\cdot)$ using $p(\mathbf{z}_0^+)$ and $p(\mathbf{y}|\mathbf{z}_{L-1}^+)$, respectively. One can then verify that

$$\mathcal{L}(\mathbf{z}^+, \mathbf{z}^-, \mathbf{s}) = \sum_{\ell=0}^{L-1} \mathcal{L}_\ell(\mathbf{z}_{\ell-1}^-, \mathbf{z}_\ell^+; \mathbf{z}_{\ell-1}^+, \mathbf{z}_\ell^-, \mathbf{s}_{\ell-1}, \mathbf{s}_\ell).$$

Theorem 5 (MAP-ML-VAMP)**.** *Consider the iterates of Algorithm 3 with MAP estimation functions* (3.12) *for fixed* $\gamma_\ell^\pm > 0$*. Suppose lines 11 and 23 are replaced with fixed values* $\alpha_{k\ell}^\pm = \alpha_\ell^\pm \in (0,1)$ *from* (3.16)*. Let* $\mathbf{s}_{k\ell}^- := \alpha_{k\ell}^+(\widehat{\mathbf{z}}_{k-1,\ell}^- - \mathbf{r}_{k\ell}^-)$ *and* $\mathbf{s}_{k\ell}^+ := \alpha_{k\ell}^-(\mathbf{r}_{k\ell}^+ - \widehat{\mathbf{z}}_{k\ell}^+)$*. Then, for* $\ell = 0, \ldots, L-1$*, the forward pass iterations satisfy*

$$\_\,, \widehat{\mathbf{z}}_{k\ell}^+ = \underset{(\mathbf{z}_{\ell-1}^-, \mathbf{z}_\ell^+)}{\operatorname{argmin}}\ \mathcal{L}_\ell(\mathbf{z}_{\ell-1}^-, \mathbf{z}_\ell^+; \widehat{\mathbf{z}}_{k,\ell-1}^+, \widehat{\mathbf{z}}_{k-1,\ell}^-, \mathbf{s}_{k,\ell-1}^+, \mathbf{s}_{k\ell}^-) \tag{3.20a}$$

$$\mathbf{s}_{k\ell}^+ = \mathbf{s}_{k\ell}^- + \alpha_\ell^+(\widehat{\mathbf{z}}_{k\ell}^+ - \widehat{\mathbf{z}}_{k-1,\ell}^-), \tag{3.20b}$$

whereas the backward pass iterations satisfy

$$\widehat{\mathbf{z}}_{k,\ell-1}^-, \_ = \underset{(\mathbf{z}_{\ell-1}^-, \mathbf{z}_\ell^+)}{\operatorname{argmin}}\ \mathcal{L}_\ell(\mathbf{z}_{\ell-1}^-, \mathbf{z}_\ell^+; \widehat{\mathbf{z}}_{k,\ell-1}^+, \widehat{\mathbf{z}}_{k\ell}^-, \mathbf{s}_{k,\ell-1}^+, \mathbf{s}_{k+1,\ell}^-) \tag{3.21a}$$

$$\mathbf{s}_{k+1,\ell-1}^- = \mathbf{s}_{k,\ell-1}^+ + \alpha_{\ell-1}^-(\widehat{\mathbf{z}}_{k,\ell-1}^+ - \widehat{\mathbf{z}}_{k,\ell-1}^-). \tag{3.21b}$$

Further, any fixed point of Algorithm 1 corresponds to a critical point of the Lagrangian (3.19)*.*

Proof. See Appendix A.3 ☐

Theorem 5 shows that the fixed-$\{\alpha_\ell^\pm\}$ version of ML-VAMP is an ADMM-type algorithm for solving the optimization problem (3.18). In the case that $\alpha_\ell^+ = \alpha_\ell^-$, this algorithm is known as the Peaceman-Rachford Splitting variant of ADMM and its convergence has been studied extensively; see [58, eqn. (3)] and [59], and the references therein. Different from ADMM, the full ML-VAMP algorithm adaptively updates $\{\alpha_{k\ell}^\pm\}$ in a way that exploits the local curvature of the objective in (3.12). Note that, in (3.20a) and (3.21a), the "_" notation means that we compute the joint minimizers over $(\mathbf{z}_{\ell-1}^+, \mathbf{z}_\ell^+)$, but only use one of them at a time for the update step.

3.3.2 Fixed Points of MMSE-ML-VAMP and Connections to Free-Energy Minimization

Recall that $\boldsymbol{z} := \{\boldsymbol{z}_\ell\}_{\ell=0}^{L-1}$ and let $\mathcal{B}$ denote the set of density functions $b(\boldsymbol{z})$ factorizable as $f_0(\boldsymbol{z}_0) f_L(\boldsymbol{z}_{L-1}) \prod_{\ell=1}^{L-1} f_\ell(\boldsymbol{z}_\ell, \boldsymbol{z}_{\ell-1})$. Notice that the true posterior $p(\boldsymbol{z}|\boldsymbol{y})$ from (3.5) belongs to this set. Essentially, this $\mathcal{B}$ captures the chain structure of the factor graph visible in the top panel of Fig. 3.2. For chain-structured (and, more generally, tree-structured) graphs, one can express any $b \in \mathcal{B}$ as [162] (see also [119, Sec. III C] for a succinct description)

$$b(\boldsymbol{z}) = \frac{\prod_{\ell=1}^{L-1} f_\ell(\boldsymbol{z}_\ell, \boldsymbol{z}_{\ell-1})}{\prod_{\ell=1}^{L-2} q_\ell(\boldsymbol{z}_\ell)}, \tag{3.22}$$

where $\{f_\ell(\boldsymbol{z}_\ell, \boldsymbol{z}_{\ell-1})\}$ and $\{q_\ell(\boldsymbol{z}_\ell)\}$ are marginal density functions of $b(\boldsymbol{z})$. As marginal densities, they must satisfy the consistent-marginal equations

$$\begin{aligned} b(\boldsymbol{z}_\ell) &= \int f_\ell(\boldsymbol{z}_\ell, \boldsymbol{z}_{\ell-1})\, \mathrm{d}\boldsymbol{z}_{\ell-1} \\ &= q_\ell(\boldsymbol{z}_\ell) = \int f_{\ell+1}(\boldsymbol{z}_{\ell+1}, \boldsymbol{z}_\ell)\, \mathrm{d}\boldsymbol{z}_{\ell+1}, \quad \forall\, \ell = 1 \dots L-1. \end{aligned} \tag{3.23}$$

Because $p(\boldsymbol{z}|\mathbf{y}) \in \mathcal{B}$, we can express it using variational optimization as

$$p(\boldsymbol{z}|\boldsymbol{y}) = \underset{b \in \mathcal{B}}{\operatorname{argmin}} D_{\mathsf{KL}}(b(\boldsymbol{z}) \| p(\boldsymbol{z}|\boldsymbol{y})), \tag{3.24}$$

where $D_{\mathsf{KL}}(b(\boldsymbol{z})\|p(\boldsymbol{z}|\mathbf{y})) := \int b(\boldsymbol{z}) \ln \frac{b(\boldsymbol{z})}{p(\boldsymbol{z}|\boldsymbol{y})}\, \mathrm{d}\boldsymbol{z}$ is the KL divergence. Plugging $b(\boldsymbol{z})$ from (3.22) into (3.24), we obtain

$$
\begin{aligned}
p(\boldsymbol{z}|\boldsymbol{y}) = \underset{b\in\mathcal{B}}{\operatorname{argmin}} \Bigg\{ & \sum_{\ell=1}^{L} D_{\mathsf{KL}}(f_\ell(\boldsymbol{z}_\ell, \boldsymbol{z}_{\ell-1})\|p(\boldsymbol{z}_\ell|\boldsymbol{z}_{\ell-1})) \\
& + \sum_{\ell=0}^{L-1} h(q_\ell(\boldsymbol{z}_\ell)) \Bigg\} \quad \text{subject to (3.23)},
\end{aligned} \tag{3.25}
$$

where $h(q_\ell(\boldsymbol{z}_\ell)) := -\int q_\ell(\boldsymbol{z}_\ell) \ln q_\ell(\boldsymbol{z}_\ell)\, \mathrm{d}\boldsymbol{z}_\ell$ is the differential entropy of q_ℓ. The cost function in (3.25) is often called the Bethe free energy [162]. In summary, because $\mathcal{B}$ is tree-structured, Bethe-free-energy minimization yields the exact posterior distribution [162].

The constrained minimization (3.25) is computationally intractable, because both the optimization variables $\{f_\ell, q_\ell\}$ and the pointwise linear constraints (3.23) are infinite dimensional. Rather than solving for the exact posterior, we might instead settle for an approximation obtained by relaxing the marginal constraints (3.23) to the following moment-matching conditions, for all $\ell = 0, 1, \ldots L-1$:

$$
\begin{aligned}
\mathbb{E}[\boldsymbol{z}_\ell|q_\ell] &= \mathbb{E}[\boldsymbol{z}_\ell|f_\ell], & \mathbb{E}[\|\boldsymbol{z}_\ell\|^2\,|q_\ell] &= \mathbb{E}[\|\boldsymbol{z}_\ell\|^2\,|f_\ell], \\
\mathbb{E}[\boldsymbol{z}_\ell|q_\ell] &= \mathbb{E}[\boldsymbol{z}_\ell|f_{\ell+1}], & \mathbb{E}[\|\boldsymbol{z}_\ell\|^2\,|q_\ell] &= \mathbb{E}[\|\boldsymbol{z}_\ell\|^2\,|f_{\ell+1}].
\end{aligned} \tag{3.26}
$$

This approach is known as expectation-consistent (EC) approximate inference [111]. Because the constraints on f_ℓ and q_ℓ in (3.26) are finite dimensional, standard Lagrangian-dual methods can be used to compute the optimal solution. Thus, the EC relaxation of the Bethe free energy minimization problem (3.25), i.e.,

$$
\begin{aligned}
\min_{f_\ell} \max_{q_\ell} \Bigg\{ & \sum_{\ell=1}^{L-1} D_{\mathsf{KL}}(f_\ell(\boldsymbol{z}_\ell, \boldsymbol{z}_{\ell-1})\|p(\boldsymbol{z}_\ell|\boldsymbol{z}_{\ell-1})) \\
& + \sum_{\ell=0}^{L-1} h(q_\ell(\boldsymbol{z}_\ell)) \Bigg\} \quad \text{subject to (3.26)},
\end{aligned} \tag{3.27}
$$

yields a tractable approximation to $p(\boldsymbol{z}|\boldsymbol{y})$.

We now establish an equivalence between the fixed points of the MMSE-ML-VAMP algorithm and the first-order stationary points of (3.27). The statement of the theorem uses the belief functions b_ℓ defined in (4.12).

Theorem 6 (MMSE-ML-VAMP)**.** *Consider a fixed point* $\big(\{\boldsymbol{r}_\ell^{\pm}\}, \{\widehat{\boldsymbol{z}}_\ell\}, \{\gamma_\ell^{\pm}\}\big)$ *of Algorithm 3 with MMSE estimation functions* (3.11)*. Then* $\{\gamma_\ell^+\boldsymbol{r}_\ell^+, \frac{\gamma_\ell^+}{2}, \gamma_\ell^-\boldsymbol{r}_\ell^-, \frac{\gamma_\ell^-}{2}\}$*, are Lagrange multipliers for* (3.26) *such that KKT conditions are satisfied for the problem* (3.27) *at primal solutions* $\{f_\ell^*, q_\ell^*\}$*. Furthermore, the marginal densities take the form* $f_\ell^*(\cdot) \propto b_\ell(\cdot|\boldsymbol{r}_\ell^-, \boldsymbol{r}_{\ell-1}^+, \gamma_\ell^-, \gamma_\ell^-, \gamma_{\ell-1}^+)$ *and* $q_\ell^* = \mathcal{N}(\widehat{\boldsymbol{z}}_\ell, \boldsymbol{I}/\eta_\ell)$*, with* $\widehat{\boldsymbol{z}}_\ell$ *and* η_ℓ *given in* (3.16)-(3.17).

Proof. See Appendix A.3. □

The above result shows that MMSE-ML-VAMP is essentially an algorithm to iteratively solve for the parameters $\big(\{\boldsymbol{r}_\ell^{\pm}\}, \{\widehat{\boldsymbol{z}}_\ell\}, \{\gamma_\ell^{\pm}\}\big)$ that characterize the EC fixed points. Importantly, $q_\ell^*(\boldsymbol{z}_\ell)$ and $f^*(\boldsymbol{z}_\ell, \boldsymbol{z}_{\ell-1})$ serve as an approximate marginal posteriors for $\boldsymbol{z}_\ell$ and $(\boldsymbol{z}_\ell, \boldsymbol{z}_{\ell-1})$. This enables us to not only compute the MMSE estimate (i.e., posterior mean), but also other estimates like the MMAE estimate (i.e., the posterior median), or quantiles of the marginal posteriors. Remarkably, in certain cases, these approximate marginal-posterior statistics become exact. This is one of the main contributions of the next section.

3.4 Analysis in the Large-System Limit

3.4.1 LSL model

In the previous section, we established that, for any set of deterministic matrices $\{\mathbf{W}_\ell\}$, MAP-ML-VAMP solves the MAP problem and MMSE-ML-VAMP solves the EC variational inference problem as the iterations $k \to \infty$. In this section, we extend the analysis of [10,127] to the rigorously study the behavior of ML-VAMP at any iteration k for classes of random matrices $\{\mathbf{W}_\ell\}$ in a certain large-system limit (LSL). The model is described in the following set of assumptions.

System model We consider a sequence of systems indexed by N. For each N, let $\boldsymbol{z}_\ell = \boldsymbol{z}_\ell^0(N) \in \mathbb{R}^{N_\ell(N)}$ be "true" vectors generated by neural network (4.1) for layers $\ell = 0, \ldots, L$, such that layer widths satisfy $\lim_{N\to\infty} N_\ell(N)/N = \beta_\ell \in (0, \infty)$. Also, let the weight matrices $\mathbf{W}_\ell$ in (4.1a) each have an SVD given by (3.13), where $\{\boldsymbol{V}_\ell\}$ are drawn uniformly from the set of orthogonal matrices in $\mathbb{R}^{N_\ell \times N_\ell}$ and independent across ℓ. The distribution on the singular values $\mathbf{s}_\ell$ will be described below.

Similar to the VAMP analysis [127], the assumption here is that weight matrices $\mathbf{W}_\ell$ are rotationally invariant, meaning that $\mathbf{V}\mathbf{W}_\ell$ and $\mathbf{W}_\ell\mathbf{V}$ are distributed identically to $\mathbf{W}_\ell$. Gaussian i.i.d. $\mathbf{W}_\ell$ as considered in the original ML-AMP work of [89] satisfy this rotationally invariant assumption, but the rotationally invariant model is more general. In particular, as described in [127], the model can have arbitrary coniditoning which is known to be a major failure mechanism of AMP methods.

ML-VAMP algorithm We assume that we generate estimates $\widehat{\mathbf{z}}_{k\ell}^{\pm}$ from the ML-VAMP algorithm, Algorithm 3. Our analysis will apply to general estimation functions, $\mathbf{g}_\ell(\cdot)$, not necessarily the MAP or MMSE estimators. However, we require two technical conditions: For the non-linear estimators, $\mathbf{g}_\ell^{\pm}$ for $\ell = 2, 4, \ldots L-2$, and $\mathbf{g}_0^+$, $\mathbf{g}_L^-$ act componentwise. Further, these estimators and their derivatives $\frac{\partial \mathbf{g}_\ell^+}{\partial z_\ell^-}, \frac{\partial \mathbf{g}_\ell^-}{\partial z_{\ell-1}^+}, \frac{\partial \mathbf{g}_0^+}{\partial z_0^-}, \frac{\partial \mathbf{g}_L^-}{\partial z_{L-1}^+}$ are uniformly Lipschitz continuous. The technical definition of uniformly Lipschitz continuous is given in Appendix A.1. For the linear layers, $\ell = 1, 3, \ldots L-1$, we assume we apply estimators $\mathbf{g}_\ell^{\pm}$ of the form (3.14) where $\mathbf{G}_\ell^{\pm}$ act componentwise. Further, $\mathbf{G}_\ell^{\pm}$ along with its derivatives are uniformly Lipschitz continuous. We also assume that the activation functions $\boldsymbol{\phi}_\ell$ in equation (4.1b) are componentwise separable and Lipschitz continuous. To simplify the analysis, we will also assume the estimation function parameters $\theta_{k\ell}^{\pm}$ converge to fixed limits,

$$\lim_{N\to\infty} \theta_{k\ell}^{\pm}(N) = \overline{\theta}_{k\ell}^{\pm}, \tag{3.28}$$

for values $\overline{\theta}^{\pm}_{k\ell}$. Importantly, in this assumption, we assume that the limiting parameter values $\overline{\theta}^{\pm}_{k\ell}$ are fixed and not data dependent. However, data dependent parameters can also be modeled [127].

Distribution of the components We follow the framework of Bayati-Montanari and describe the statistics on the unknown quantities via their empirical convergence – see Appendix A.1. For $\ell = 1, 3, \ldots L-1$, define $\overline{\mathbf{b}}_\ell := \mathbf{V}_\ell^{\mathsf{T}} \mathbf{b}_\ell$ and $\overline{\boldsymbol{\xi}}_\ell := \mathbf{V}_\ell^{\mathsf{T}} \boldsymbol{\xi}_\ell$. We assume that the sequence of true vectors $\boldsymbol{z}_0^0$, singular values $\boldsymbol{s}_\ell$, bias vectors $\overline{\mathbf{b}}_\ell$, and noise realizations $\overline{\boldsymbol{\xi}}_\ell$ empirically converge as

$$\left\{z^0_{0,n}\right\} \overset{PL(2)}{=} Z^0_0, \quad \left\{\xi_{\ell,n}\right\} \overset{PL(2)}{=} \Xi_\ell, \qquad \forall\, \ell \text{ even}, \tag{3.29a}$$

$$\left\{(s_{\ell,n}, \overline{b}_{\ell,n}, \overline{\xi}_{\ell,n})\right\} \overset{PL(2)}{=} (S_\ell, \overline{B}_\ell, \overline{\Xi}_\ell), \qquad \forall\, \ell \text{ odd}, \tag{3.29b}$$

to random variables $Z^0_0, \Xi_\ell, S_\ell, \overline{B}_\ell, \overline{\Xi}_\ell$. We will also assume that the singular values are bounded, i.e., $s_{\ell,n} < S_{\ell,\max}\ \forall n$. Also, the initial vectors $\boldsymbol{r}^-_{0\ell}$ converge as,

$$\begin{aligned} \left\{[\boldsymbol{r}^-_{0\ell} - \boldsymbol{z}^0_\ell]_n\right\} &\overset{PL(2)}{=} Q^-_{0\ell}, & \ell &= 0, 2, \ldots, L, \\ \left\{[\boldsymbol{V}_\ell^\top(\boldsymbol{r}^-_{0\ell} - \boldsymbol{z}^0_\ell)]_n\right\} &\overset{PL(2)}{=} Q^-_{0\ell}, & \ell &= 1, 3, \ldots, L-1, \end{aligned} \tag{3.30}$$

where $(Q^-_{0\ell}, Q^-_{1\ell}, \ldots Q^-_{L-1,\ell})$ is jointly Gaussian independent of Z^0_0, $\{\Xi_\ell\}$, $\{S_\ell, \overline{B}_\ell, \overline{\Xi}_\ell\}$.

State Evolution Under the above assumptions, our main result is to show that the asymptotic distribution of the quantities from ML-VAMP algorithm converge to certain distributions. The distributions are described by a set of deterministic parameters $\{\mathbf{K}^+_{k\ell}, \tau^-_{k\ell}, \overline{\alpha}^{\pm}_{k\ell}, \overline{\gamma}^{\pm}_{k\ell}, \overline{\eta}^{\pm}_{k\ell}\}$. The evolve according to a scalar recursion called the state evolution (SE), given in Algorithm 6 in Appendix A.2. We assume $\overline{\alpha}^{\pm}_{k\ell} \in (0,1)$ for all iterations k and $\ell = 0, 1, \ldots L-1$.

3.4.2 SE Analysis in the LSL

Under these assumptions, we can now state our main result. Let $\mathbb{S}^d$ denote the space of symmetric positive definite matrices in $\mathbb{R}^{d\times d}$. The deterministic quantities $\{\mathbf{K}^+_{k\ell}, \tau^-_{k\ell+1}, \overline{\alpha}^{\pm}_{k\ell}, \overline{\gamma}^{\pm}_{k\ell}, \overline{\eta}^{\pm}_{k\ell}\}_{\ell=0}^{L-1}$

referenced in the theorem below are defined in an iteration called the State Evolution given in Algorithm 6 (see Appendix A.2 of Supplementary materials).

Theorem 7. *Consider the system under the above assumptions. There exist deterministic parameters* $\{\mathbf{K}^+_{k\ell}, \tau^-_{k\ell+1}, \overline{\alpha}^\pm_{k\ell}, \overline{\gamma}^\pm_{k\ell}, \overline{\eta}^\pm_{k\ell}\}_{\ell=0}^{L-1}$ *with* $\mathbf{K}_{k\ell} \in \mathbb{S}^2, \tau^-_{k\ell} > 0, \overline{\gamma}^\pm_{k\ell} > 0, \overline{\eta}^\pm_{k\ell} > 0, \overline{\alpha}_{k\ell} \in (0,1)$ *such that the following convergence holds. For any componentwise pseudo-Lipschitz function* $\boldsymbol{\psi}$ *of order 2, iteration index* k*, and layer index* $\ell = 2, 4, \dots L-2$,

$$\begin{aligned}\lim_{N\to\infty} \left\langle \boldsymbol{\psi}\left(\boldsymbol{z}^0_{\ell-1}, \widehat{\boldsymbol{z}}^-_{k,\ell-1}, \widehat{\boldsymbol{z}}^+_{k\ell}\right) \right\rangle &\overset{a.s.}{=} \\ &\mathbb{E}\Big[\psi\Big(\mathsf{A}, g^-_\ell(\mathsf{C} + \phi_\ell(\mathsf{A}, \Xi_\ell), \mathsf{B} + \mathsf{A}, \overline{\gamma}^-_{k\ell}, \overline{\gamma}^+_{k,\ell-1}), \\ &\qquad g^+_\ell(\mathsf{C} + \phi_\ell(\mathsf{A}, \Xi_\ell), \mathsf{B} + \mathsf{A}, \overline{\gamma}^-_{k\ell}, \overline{\gamma}^+_{k,\ell-1})\Big)\Big],\end{aligned} \tag{3.31}$$

$$\lim_{N\to\infty} \langle \boldsymbol{\psi}(\boldsymbol{z}^0_0, \widehat{\boldsymbol{z}}^+_{k0})\rangle \overset{a.s.}{=} \mathbb{E}\left[\psi(Z^0_0, g^+_0(\mathsf{F} + Z^0_0, \overline{\gamma}^-_0)\right], \tag{3.32}$$

$$\lim_{N\to\infty} \langle \boldsymbol{\psi}(\boldsymbol{z}^0_{L-1}, \widehat{\boldsymbol{z}}^-_{k,L-1})\rangle \overset{a.s.}{=} \mathbb{E}\psi(\mathsf{D}, g^-_L(\mathsf{E}+\mathsf{D}, \overline{\gamma}^+_{L-1})), \tag{3.33}$$

where $(\mathsf{A}, \mathsf{B}) \sim \mathcal{N}(0, \mathbf{K}^+_{k,\ell-1})$ *and* $\mathsf{C} \sim \mathcal{N}(0, \tau^-_{k\ell})$ *are mutually independent and independent of* Ξ_ℓ*;* $(\mathsf{D}, \mathsf{E}) \sim \mathcal{N}(0, \mathbf{K}^+_{k,L-1})$ *is independent of* Ξ_L *and* $\mathsf{F} \sim \mathcal{N}(0, \tau^-_{k0})$ *is independent of* Z^0_0.

Similarly for any layer index $\ell = 1, 3, \dots, L-1$*, we have*

$$\begin{aligned}\lim_{N\to\infty} \langle \boldsymbol{\psi}\left(\boldsymbol{V}_{\ell-1}\boldsymbol{z}^0_{\ell-1}, \boldsymbol{V}_{\ell-1}\widehat{\boldsymbol{z}}^-_{k,\ell-1}, \boldsymbol{V}^\top_\ell \widehat{\boldsymbol{z}}^+_{k\ell}\right)\rangle &\overset{a.s.}{=} \\ &\mathbb{E}\Big[\psi\Big(\mathsf{A}, G^-_\ell(\mathsf{C} + \mathsf{D}, \mathsf{B} + \mathsf{A}, S_\ell, \overline{B}_\ell, \overline{\gamma}^-_{k\ell}, \overline{\gamma}^+_{k,\ell-1}), \\ &\qquad G^+_\ell(\mathsf{C} + \mathsf{D}, \mathsf{B} + \mathsf{A}, S_\ell, \overline{B}_\ell, \overline{\gamma}^-_{k\ell}, \overline{\gamma}^+_{k,\ell-1})\Big)\Big],\end{aligned} \tag{3.34}$$

where $(\mathsf{A}, \mathsf{B}) \sim \mathcal{N}(0, \mathbf{K}^+_{k,\ell-1})$ *and* $\mathsf{C} \sim \mathcal{N}(0, \tau^-_{k\ell})$ *are mutually independent and independent of* $(S_\ell, \overline{B}_\ell, \overline{\Xi}_\ell)$*, and* $\mathsf{D} = S_\ell\mathsf{A} + \overline{B}_\ell + \overline{\Xi}_\ell$.

Furthermore, if $\overline{\gamma}^\pm_{k\ell}, \overline{\eta}^\pm_{k\ell}$*, are defined analogous to* (3.9) *using* $\overline{\alpha}^\pm_{k\ell}$*, then for all* ℓ,

$$\lim_{N\to\infty} (\alpha^\pm_{k,\ell}, \gamma^\pm_{k,\ell}, \eta^\pm_{k,\ell}) \overset{a.s.}{=} (\overline{\alpha}^\pm_{k,\ell}, \overline{\gamma}^\pm_{k,\ell}, \overline{\eta}^\pm_{k,\ell}). \tag{3.35}$$

Proof. See Appendix A.4. □

The key value of Theorem 9 is that we can *exactly characterize* the asymptotic joint distribution of the true vectors $\boldsymbol{z}_\ell^0$ and the ML-VAMP estimates $\widehat{\mathbf{z}}_{k\ell}^{\pm}$. The asymptotic joint distribution, can be used to compute various key quantities. For example, suppose we wish to compute the mean squared error (MSE). Let $\psi(z^0, \widehat{\boldsymbol{z}}) = (z^0 - \widehat{\boldsymbol{z}})^2$, whereby $\left\langle \boldsymbol{\psi}(\boldsymbol{z}_\ell^0, \widehat{\mathbf{z}}_\ell^-) \right\rangle = \frac{1}{N} \left\| \boldsymbol{z}_\ell^0 - \widehat{\mathbf{z}}_\ell^- \right\|^2$. Observe that ψ is a pseudo-Lipschitz function of order 2, whereby we can apply Theorem 9. Using (3.31), we get the asymptotic MSE on the k^{th}-iteration estimates for $\ell = 2, 4, \ldots L{-}2$:

$$\begin{aligned}
&\lim_{N_{\ell-1}\to\infty} \tfrac{1}{N_{\ell-1}} \left\| \widehat{\boldsymbol{z}}_{k,\ell-1}^- - \boldsymbol{z}_{\ell-1}^0 \right\|^2 \overset{a.s.}{=} \\
&\qquad \mathbb{E}\left[\left(g_\ell^-(\mathsf{C} + \phi_\ell(\mathsf{A}, \Xi_\ell), \mathsf{B} + \mathsf{A}, \overline{\gamma}_{k\ell}^-, \overline{\gamma}_{k,\ell-1}^+) - \mathsf{A} \right)^2 \right], \\
&\lim_{N_\ell\to\infty} \tfrac{1}{N_\ell} \left\| \widehat{\boldsymbol{z}}_{k\ell}^+ - \boldsymbol{z}_\ell^0 \right\|^2 \overset{a.s.}{=} \\
&\qquad \mathbb{E}\left[\left(g_\ell^+(\mathsf{C} + \phi_\ell(\mathsf{A}, \Xi_\ell), \mathsf{B} + \mathsf{A}, \overline{\gamma}_{k\ell}^-, \overline{\gamma}_{k,\ell-1}^+) - \phi_\ell(\mathsf{A}, \Xi_\ell) \right)^2 \right],
\end{aligned}$$

where we used the fact that ϕ_ℓ is pseudo-Lipschitz of order 2, and $\boldsymbol{z}_\ell^0 = \phi_\ell(\boldsymbol{z}_{\ell-1}^0, \boldsymbol{\xi}_\ell)$ from (4.1b). Similarly, using (3.34), we get the kth-iteration MSE for $\ell = 1, 3, \ldots L{-}1$:

$$\begin{aligned}
&\lim_{N_{\ell-1}\to\infty} \tfrac{1}{N_{\ell-1}} \left\| \widehat{\boldsymbol{z}}_{k,\ell-1}^- - \boldsymbol{z}_{\ell-1}^0 \right\|^2 = \tfrac{1}{N_{\ell-1}} \left\| \boldsymbol{V}_{\ell-1}(\widehat{\boldsymbol{z}}_{k,\ell-1}^- - \boldsymbol{z}_{\ell-1}^0) \right\|^2 \\
&\qquad \overset{a.s.}{=} \mathbb{E}\left[\left(G_\ell^-(\mathsf{C} + \mathsf{D}, \mathsf{B} + \mathsf{A}, S_\ell, \overline{B}_\ell, \overline{\gamma}_{k,l}^+, \overline{\gamma}_{k,\ell-1}^-) - \mathsf{A} \right)^2 \right]. \\
&\lim_{N_\ell\to\infty} \tfrac{1}{N_\ell} \left\| \widehat{\boldsymbol{z}}_{k\ell}^+ - \boldsymbol{z}_\ell^0 \right\|^2 = \tfrac{1}{N_\ell} \left\| \boldsymbol{V}_\ell^\top (\widehat{\boldsymbol{z}}_{k\ell}^+ - \boldsymbol{z}_\ell^0) \right\|^2 \\
&\qquad \overset{a.s.}{=} \mathbb{E}\left[\left(G_\ell^+(\mathsf{C} + \mathsf{D}, \mathsf{B} + \mathsf{A}, S_\ell, \overline{B}_\ell, \overline{\gamma}_{kl}^+, \overline{\gamma}_{k,\ell-1}^-) - \mathsf{D} \right)^2 \right],
\end{aligned}$$

where $\mathsf{D} = S_\ell \mathsf{A} + \overline{B}_\ell + \overline{\Xi}_\ell$. Here we used the rotational invariance of the ℓ_2 norm, and the fact that equation (4.1a) is equivalent to $\boldsymbol{V}_\ell^\top \boldsymbol{z}_\ell^0 = \mathrm{Diag}(\boldsymbol{s}_\ell) \boldsymbol{V}_{\ell-1} \boldsymbol{z}_{\ell-1}^0 + \overline{\mathbf{b}}_\ell$ using the SVD (3.13) of the weight matrices $\boldsymbol{W}_\ell$.

At the heart of the proof lies a key insight: Due to the randomness of the unitary matrices $\boldsymbol{V}_\ell$, the quantities $(\boldsymbol{z}_\ell^0, \boldsymbol{r}_{k\ell}^- - \boldsymbol{z}_\ell^0, \boldsymbol{r}_{k,\ell-1}^+ - \boldsymbol{z}_{\ell-1}^0)$ are asymptotically jointly Gaussian for even ℓ, with the asymptotic covariance matrix of $\{(z_{\ell-1,n}^0, r_{k,\ell-1,n}^+ - z_{\ell-1,n}^0, r_{k\ell,n}^- - z_{\ell,n}^0)\}$ given by $\left[\begin{smallmatrix} \mathbf{K}_{k\ell}^+ & \mathbf{0} \\ \mathbf{0} & \tau_{k\ell}^- \end{smallmatrix}\right]$, where $\mathbf{K}_{k\ell} \in \mathbb{R}^{2\times 2}$ and $\tau_{k\ell}^-$ is a scalar. After establishing the asymptotic Gaussianity of $(\boldsymbol{z}_\ell^0, \boldsymbol{r}_{k\ell}^- - \boldsymbol{z}_\ell^0, \boldsymbol{r}_{k,\ell-1}^+ - \boldsymbol{z}_{\ell-1}^0)$, since $\widehat{\boldsymbol{z}}_\ell$ and $\widehat{\boldsymbol{z}}_{\ell-1}$ are componentwise functions of this triplet, we have the PL(2) convergence result in (3.31). Similarly, for odd ℓ, we can show that $\big(\boldsymbol{V}_{\ell-1}\boldsymbol{z}^0_{\ell-1}, \boldsymbol{V}_{\ell-1}\boldsymbol{r}^+_{k,\ell-1}, \boldsymbol{V}_\ell^\top \boldsymbol{r}^-_{k\ell}\big)$ is asymptotically Gaussian. For these ℓ, $\boldsymbol{V}_{\ell-1}\widehat{\boldsymbol{z}}^-_{k,\ell-1}$ and $\boldsymbol{V}_\ell^\top \widehat{\boldsymbol{z}}^+_{k\ell}$ are functions of the triplet, which gives the result in (3.34).

Due to the asymptotic normality mentioned above, the inputs $(\boldsymbol{r}^-_\ell, \boldsymbol{r}^+_{\ell-1})$ to the estimators $\mathbf{g}^\pm_\ell$ are the true signals $(\boldsymbol{z}^0_{\ell-1}, \boldsymbol{z}^0_\ell)$ plus additive white Gaussian noise (AWGN). Hence, the estimators $\mathbf{g}^\pm_\ell$ act as denoisers, and ML-VAMP effectively reduces the inference problem 3.2 into a sequence of linear transformations and denoising problems. The denoising problems are solved by $\mathbf{g}^\pm_\ell$ for even ℓ, and by $\mathbf{G}^\pm_\ell$ for odd ℓ.

3.4.3 MMSE Estimation and Connections to the Replica Predictions

We next consider the special case of using MMSE estimators corresponding to the true distributions. In this case, the SE equations simplify considerably using the following *MSE functions*: let $\widehat{\mathbf{z}}^-_{\ell-1}$, $\widehat{\mathbf{z}}^+_\ell$ be the MMSE estimates of $\mathbf{z}^0_{\ell-1}$ and $\mathbf{z}^0_\ell$ from the variables $\mathbf{r}^+_{\ell-1}, \mathbf{r}^-_\ell$ under the joint density (4.12). Let $\mathcal{E}^\pm(\cdot)$ be the corresponding mean squared errors,

$$\begin{aligned}\mathcal{E}^+_\ell(\overline{\gamma}^+_{\ell-1}, \overline{\gamma}^-_\ell) &:= \lim_{N\to\infty}\frac{1}{N}\mathbb{E}\left\|\mathbf{z}^0_\ell - \widehat{\mathbf{z}}^+_\ell\right\|^2,\\ \mathcal{E}^-_{\ell-1}(\overline{\gamma}^+_{\ell-1}, \overline{\gamma}^-_\ell) &:= \lim_{N\to\infty}\frac{1}{N}\mathbb{E}\left\|\mathbf{z}^0_{\ell-1} - \widehat{\mathbf{z}}^-_{\ell-1}\right\|^2.\end{aligned} \tag{3.36}$$

Theorem 8 (MSE of MMSE-ML-VAMP)**.** *Consider the system under the assumptions of Theorem 9, with MMSE estimation functions* $\mathbf{g}^\pm_\ell, \mathbf{g}^+_0, \mathbf{g}^-_L$ *from* (3.11) *for the belief estimates in* (4.12) *with* $\gamma^+_{k\ell} = \overline{\gamma}^\pm_{k\ell}$ *from the state-evolution equations. Then, the state evolution equations reduce to*

$$\begin{aligned}\overline{\gamma}^+_{k\ell} &= \frac{1}{\mathcal{E}^+_\ell(\overline{\gamma}^-_{k\ell}, \overline{\gamma}^+_{k,\ell-1})} - \overline{\gamma}^-_{k\ell},\\ \overline{\gamma}^-_{k+1,\ell} &= \frac{1}{\mathcal{E}^-_\ell(\overline{\gamma}^-_{k+1,\ell+1}, \overline{\gamma}^+_{k\ell})} - \overline{\gamma}^+_{k\ell},\end{aligned} \tag{3.37}$$

where $1/\overline{\eta}^+_{k\ell} = \mathcal{E}^+_\ell(\overline{\gamma}^-_{k\ell}, \overline{\gamma}^+_{k,\ell-1})$ *is the MSE of the estimate* $\widehat{\boldsymbol{z}}^+_{k\ell}$.

Proof. See Appendix A.4. □

Since the estimation functions in Theorem 8 are the MSE optimal functions for true densities, we will call this selection of estimation functions the *MMSE matched estimators*. Under the assumption of MMSE matched estimators, the theorem shows that the MSE error has a simple set of recursive expressions.

It is useful to compare the predicted MSE with the predicted optimal values. The works [47, 131] postulate the optimal MSE for inference in deep networks under the LSL model described above using the replica method from statistical physics. Interestingly, it is shown in [131, Thm.2] that the predicted minimum MSE satisfies equations that exactly agree with the fixed points of the updates (3.37). Thus, when the fixed points of (3.37) are unique, ML-VAMP with matched MMSE estimators provably achieves the Bayes optimal MSE predicted by the replica method. Although the replica method is not rigorous, this MSE predictions have been indepedently proven for the Gaussian case in [131] and certain two layer networks in [47]. This situation is similar to several other works relating the MSE of AMP with replica predictions [6, 79, 132]. The consequence is that, if the replica method is correct, ML-VAMP provides a computationally efficient method for inference with testable conditions under which it achieves the Bayes optimal MSE.

3.5 Numerical Simulations

We now numerically investigate the MAP-ML-VAMP and MMSE-ML-VAMP algorithms using two sets of experiments, where in each case the goal was to solve an estimation problem of the form in (3.2) using a neural network of the form in (4.1). We used the Python 3.7 implementation of the ML-VAMP algorithm available on GitHub.[2]

The first set of experiments uses random draws of a synthetic network to validate the claims made about the ML-VAMP state-evolution (SE) in Theorem 9. In addition, it compares MAP-ML-VAMP and MMSE-ML-VAMP to the MAP approach (3.4) using a standard gradient-based solver, ADAM [76]. The second set of experiments applies ML-

[2]See https://github.com/GAMPTeam/vampyre.

VAMP to image inpainting, using images of handwritten digits from the widely used MNIST dataset. Here, MAP-ML-VAMP and MMSE-ML-VAMP are respectively compared to the optimization approach (3.4) using the ADAM solver, and Stochastic Gradient Langevin Dynamics (SGLD) [160], an MCMC-based sampling method that approximates $\mathbb{E}[\boldsymbol{z}|\boldsymbol{y}]$.

3.5.1 Performance on a Synthetic Network

We first considered a 7-layer neural network of the form in (4.1). The first six layers, with dimensions $N_0 = 20$, $N_1 = N_2 = 100$, $N_3 = N_4 = 500$, $N_5 = N_6 = 784$, formed a (deterministic) deep generative prior driven by i.i.d. Gaussian $\boldsymbol{z}_0^0$. The matrices $\mathbf{W}_1, \mathbf{W}_3, \mathbf{W}_5$ and biases $\mathbf{b}_1, \mathbf{b}_3, \mathbf{b}_5$ were drawn i.i.d. Gaussian, and the activation functions ϕ_2, ϕ_4, ϕ_6 were ReLU. The mean of the bias vectors $\mathbf{b}_\ell$ was chosen so that a fixed fraction, ρ, of the linear outputs were positive, so that only the fraction ρ of the ReLU outputs were non-zero. Because this generative network is random rather than trained, we refer to it as "synthetic." The final layer, which takes the form $\mathbf{y} = \mathbf{A}\boldsymbol{z}_6^0 + \boldsymbol{\xi}_6$, generates noisy, compressed measurements of $\boldsymbol{z}_6^0$. Similar to [125], the matrix $\mathbf{A} \in \boldsymbol{R}^{M \times N_6}$ was constructed from the SVD $\boldsymbol{A} = \boldsymbol{U}\,\mathrm{Diag}(\boldsymbol{s})\boldsymbol{V}^\mathsf{T}$, where the singular-vector matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ were drawn uniformly from the set of orthogonal matrices, and the singular values were geometrically spaced (i.e., $s_i/s_{i-1} = \kappa\ \forall i$) to achieve a condition number of $s_1/s_M = 10$. It is known that such matrices cause standard AMP algorithms to fail [125], but not VAMP algorithms [127]. The number of compressed measurements, M, was varied from 10 to 300, and the noise vector $\boldsymbol{\xi}$ was drawn i.i.d. Gaussian with a variance set to achieve a signal-to-noise ratio of $10\log_{10}(\mathbb{E}\|\boldsymbol{A}\boldsymbol{z}_6^0\|^2/\mathbb{E}\|\boldsymbol{\xi}\|^2) = 30$ dB.

To quantify the performance of ML-VAMP, we repeated the following 1000 times. First, we drew a random neural network as described above. Then we ran the ML-VAMP algorithm for 100 iterations, recording the normalized MSE (in dB) of the iteration-k estimate of the network input, $\widehat{\boldsymbol{z}}_{k0}^{\pm}$:

$$\mathrm{NMSE}(\widehat{\boldsymbol{z}}_{k0}^{\pm}) := 10\log_{10}\left[\frac{\|\boldsymbol{z}_0^0 - \widehat{\boldsymbol{z}}_{k0}^{\pm}\|^2}{\|\boldsymbol{z}_0^0\|^2}\right].$$

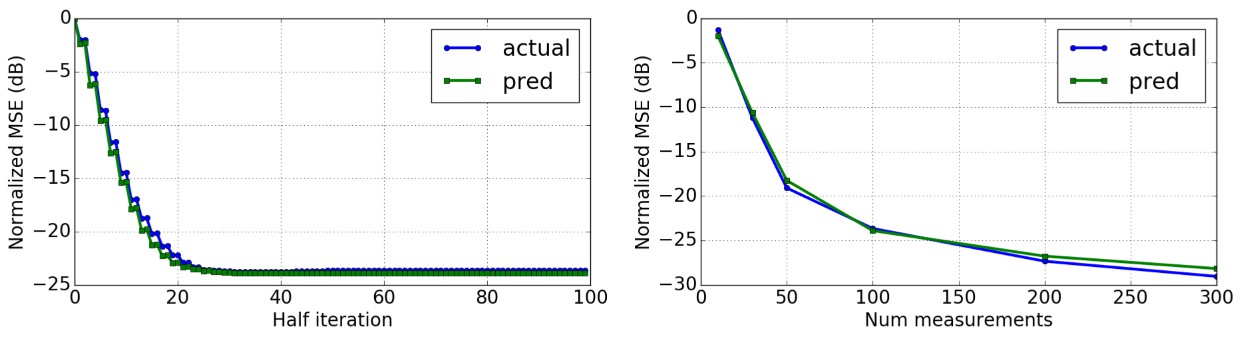


Figure 3.3: NMSE of MMSE-ML-VAMP and its SE prediction when estimating the input to a randomly generated 7-layer neural network (see text of Section 3.5.1). Left panel: Average NMSE versus half-iteration with $M = 100$ measurements. Right panel: Average NMSE verus measurements M after 50 iterations.

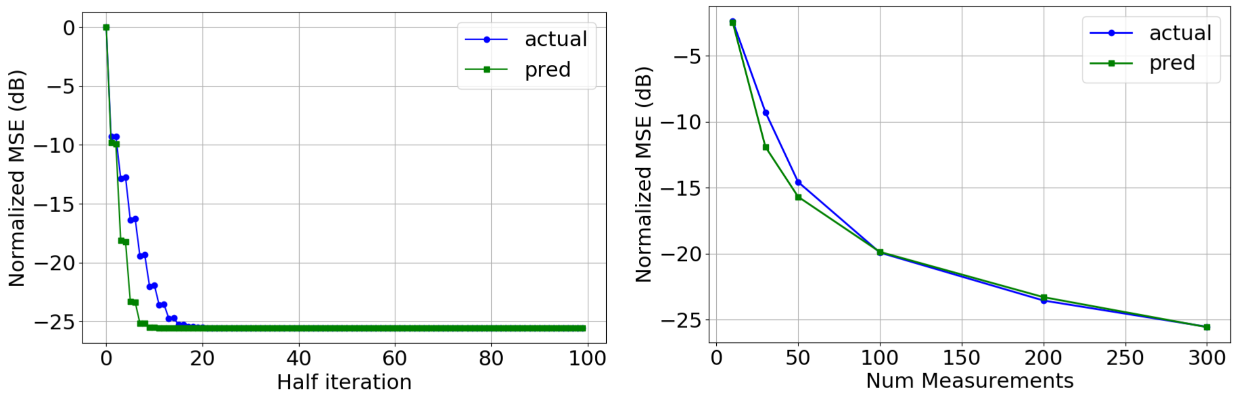


Figure 3.4: Simulation with randomly generated neural network with MAP estimators from equation (3.12). Left panel: Normalized mean squared error (NMSE) for ML-VAMP and the predicted MSE as a function of the iteration with $M = 100$ measurements. Right panel: Final NMSE (50 iterations) for ML-VAMP and the predicted MSE as a function of the number of measurements, M. $\rho = 0.9$

Since ML-VAMP computes two estimates of $\boldsymbol{z}_0^0$ at each iteration, we consider each estimate as corresponding to a "half iteration."

Validation of SE Prediction For MMSE-ML-VAMP, the left panel of Fig. 3.3 shows the NMSE versus half-iteration for $M = 100$ compressed measurements. The value shown is the average over 1000 random realizations. Also shown is the MSE predicted by the ML-VAMP state evolution. Comparing the two traces, we see that the SE predicts the actual behavior of MMSE-ML-VAMP remarkably well, within approximately 1 dB. The right panel

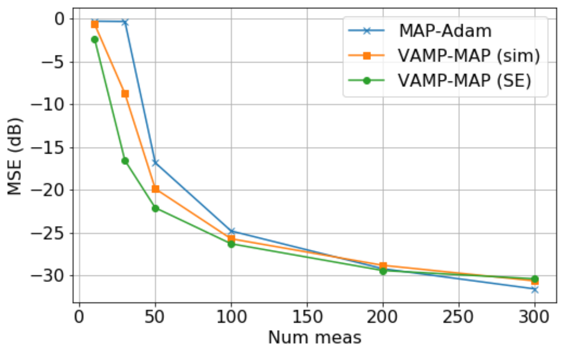


Figure 3.5: Simulation with randomly generated neural network with MAP estimators from equation (3.12). Final NMSE for (a) MAP inference computed by Adam optimizer; (b) MAP inference from ML-VAMP; (c) State evolution prediction.

shows the NMSE after $k = 50$ iterations (i.e., 100 half-iterations) for several number of measurements M. Again we see an excellent agreement between the actual MSE and the SE prediction. In both cases we used the positive fraction $\rho = 0.4$. Analogous results are shown for MAP-ML-VAMP in Fig. 3.4. There we see an excellent agreement between the actual MSE and the SE prediction for iterations $k \geq 15$ and all values of M.

Comparison to ADAM We now compare the MSE of MAP-ML-VAMP and its SE to that of the MAP approach (3.4) using the ADAM optimizer [76], as implemented in Tensorflow. As before, the goal was to recover the input $\mathbf{z}_0^0$ to the 7-layer synthetic network from a measurement of its output. Fig. 3.5 shows the median NMSE over 40 random network realizations for several values of M, the number of measurements. We see that, for $M \geq 100$, the performance of MAP-ML-VAMP closely matches its SE prediction, as well as the performance of the ADAM-based MAP approach (3.4). For $M < 100$, there is a discrepancy between the MSE performance of MAP-ML-VAMP and its SE prediction, which is likely due to the relatively small dimensions involved. Also, for small M, MAP-ML-VAMP appears to achieve slightly better MSE performance than the ADAMP-based MAP approach (3.4). Since both are attempting to solve the same problem, the difference is likely due to ML-VAMP finding better local minima.

3.5.2 Image Inpainting: MNIST dataset

To demonstrate that ML-VAMP can also work on a real-world dataset, we perform inpainting on the MNIST dataset. The MNIST dataset consists of $28 \times 28 = 784$ pixel images of handwritten digits, as shown in the first column of Fig. 3.6.

To start, we trained a 4-layer (deterministic) deep generative prior model from 50 000 digits using a variational autoencoder (VAE) [77]. The VAE "decoder" network was designed to accept 20-dimensional i.i.d. Gaussian random inputs $\boldsymbol{z}_0$ with zero mean and unit variance, and to produce MNIST-like images $\mathbf{x}$. In particular, this network began with a linear layer with 400 outputs, followed by a ReLU activations, followed by a linear layer with 784 units, followed by sigmoid activations that forced the final pixel values to between 0 and 1.

Given an image, $\mathbf{x}$, our measurement process produced $\mathbf{y}$ by erasing rows 10-20 of $\mathbf{x}$, as shown in the second column of Fig. 3.6. This process is known as "occlusion." By appending the occlusion layer onto our deep generative prior, we got a 5-layer network that generates an occluded MNIST image $\boldsymbol{y}$ from a random input $\boldsymbol{z}_0$. The "inpainting problem" is to recover the image $\boldsymbol{x} = \boldsymbol{z}_4$ from the occluded image $\boldsymbol{y}$.

For this inpainting problem, we compared MAP-ML-VAMP and MMSE-ML-VAMP to the MAP estimation approach (3.4) using the ADAM solver, and to Metropolis-Adjusted Langevin Algorithm (MALA) [104,160], an MCMC-based sampling method that approximates $\mathbb{E}[\boldsymbol{z}|\boldsymbol{y}]$ by using discrete Langevin dynamics to generate proposal samples for Metropolis-Hastings algorithm [57]. Example image reconstructions are shown in Fig. 3.6. There we see that the qualitative performance of ML-VAMP is comparable to the baseline solvers.

3.6 Discussion

Inference using deep generative prior models provides a powerful tool for complex inverse problems. Rigorous theoretical analysis of these methods has been difficult due to the non-convex nature of the models. The ML-VAMP methodology for MMSE as well as MAP

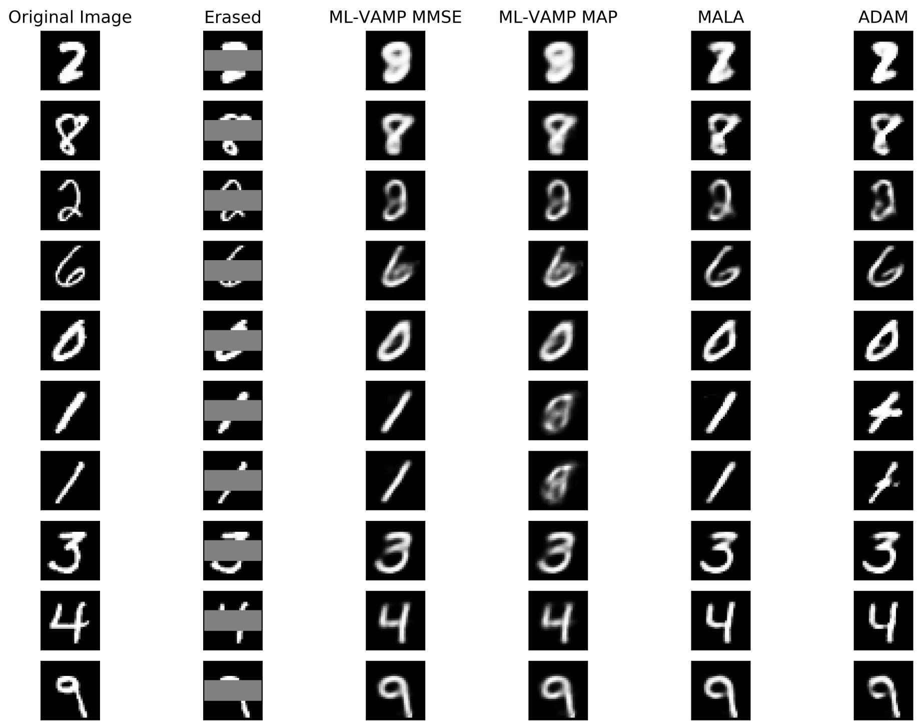


Figure 3.6: MNIST inpainting: Original 28×28 images of handwritten digits (Col 1), with rows 10-20 are erased (Col 2). Comparison of reconstructions using MAP estimation with ADAM solver (Col 3), MAP estimation with ML-VAMP algorithm (Col 4), MMSE estimation with the Metropolis Adjusted Langevin Algorithm (Col 5), and MMSE estimation with ML-VAMP algorithm (Col 6).

estimation provides a principled and computationally tractable method for performing the inference whose performance can be rigorously and precisely characterized in a certain large system limit. The approach thus offers a new and potentially powerful approach for understanding and improving deep neural network based models for inference.

Chapter 4

Multi-Layer Inverse Problems over Matrices

In this chapter we generalize the setup of Chapter 3 to the case where each of $\mathbf{Z}_\ell$ are matrix valued.

We consider the problem of estimating the input and hidden variables of a stochastic multi-layer neural network from an observation of the output. The hidden variables in each layer are represented as matrices with statistical interactions along both rows as well as columns. This problem applies to matrix imputation, signal recovery via deep generative prior models, multi-task and mixed regression, and learning certain classes of two-layer neural networks. We extend a recently-developed algorithm – Multi-Layer Vector Approximate Message Passing (ML-VAMP), for this matrix-valued inference problem. It is shown that the performance of the proposed Multi-Layer Matrix VAMP (ML-Mat-VAMP) algorithm can be exactly predicted in a certain random large-system limit, where the dimensions $N \times d$ of the unknown quantities grow as $N \to \infty$ with d fixed. In the two-layer neural-network learning problem, this scaling corresponds to the case where the number of input features as well as

This work was published in [116] and was coauthoer with Mojtaba Sahraee-Ardakan, Sundeep Rangan, Philip Schniter, and Alyson K. Fletcher training samples grow to infinity but the number of hidden nodes stays fixed. The analysis enables a precise prediction of the parameter and test error of the learning.

4.1 Introduction

Consider an L-layer stochastic neural network given by

$$\mathbf{Z}_\ell^0 = \mathbf{W}_\ell \mathbf{Z}_{\ell-1}^0 + \mathbf{B}_\ell + \Xi_\ell^0, \qquad \ell = 1, 3, \ldots, L-1, \tag{4.1a}$$

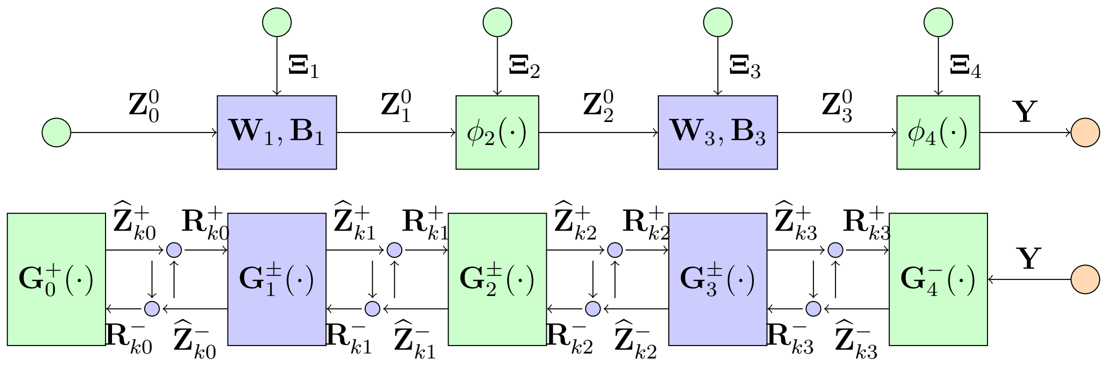
$$\mathbf{Z}_\ell^0 = \phi_\ell(\mathbf{Z}_{\ell-1}^0, \Xi_\ell^0), \qquad \ell = 2, 4, \ldots, L, \tag{4.1b}$$

where, for $\ell = 0, 1, \ldots, L$, we have *true* activations $\mathbf{Z}_\ell^0 \in \mathbb{R}^{n_\ell \times d}$, weights $\mathbf{W}_\ell \in \mathbb{R}^{n_\ell \times n_{\ell-1}}$, bias matrices $\mathbf{B}_\ell \in \mathbb{R}^{n_\ell \times d}$, and *true* noise realizations Ξ_ℓ^0. The activation functions $\phi_\ell : \mathbb{R}^{n_{\ell-1} \times d} \to \mathbb{R}^{n_\ell \times d}$ are known non-linear functions acting row-wise on their inputs. See Fig. 4.1 (TOP). We use the superscript ^{0} in $\mathbf{Z}_\ell^0$ to indicate the true values of the variables, in contrast to estimated values denoted by $\widehat{\mathbf{Z}}_\ell$ discussed later. We model the true values $\mathbf{Z}_0^0$ as a realization of random $\mathbf{Z}_0$, where the rows $\mathbf{z}_{0,i:}^\top$ of $\mathbf{Z}_0$ are i.i.d. with distribution p_0: $p(\mathbf{Z}_0) = \prod_{i=1}^{n_0} p_0(\mathbf{z}_{0,i:})$. Similarly, we also assume that Ξ_ℓ^0 are realizations of random Ξ_ℓ with i.i.d. rows $\boldsymbol{\xi}_{\ell,i:}^\top$. For odd ℓ, the rows $\boldsymbol{\xi}_{\ell,i:}$ are zero-mean multivariate Gaussian with covariance matrix $\mathbf{N}_\ell^{-1} \in \mathbb{R}^{d \times d}$, whereas for even ℓ, the rows $\boldsymbol{\xi}_{\ell,i:}$ can be arbitrarily distributed but i.i.d.

Denoting by $\mathbf{Y} := \mathbf{Z}_L^0 \in \mathbb{R}^{n_L \times d}$ the output of the network, we consider the following matrix inference problem:

$$\text{Estimate } \mathbf{Z} := \{\mathbf{Z}_\ell\}_{\ell=0}^{L-1} \qquad \text{given } \mathbf{Y} := \mathbf{Z}_L^0 \text{ and } \{\mathbf{W}_{2k-1}, \mathbf{B}_{2k-1}, \phi_{2k}\}_{k=1}^{L/2}. \tag{4.2}$$

A key feature of the problem we consider here is that the unknowns, $\mathbf{Z}_\ell$, are *matrix-valued* with d columns with statistical dependencies between the columns. As we will see in Section 4.2, the matrix-valued case applies to several problems of broad interest such as matrix imputation, multi-task and mixed regression problems, sketched clustering. We also show that via this formulation we can analyze the learning in two layer neural networks under some architectural assumptions.



Ξ_1
Ξ_2
Ξ_3
Ξ_4
$\mathbf{Z}_0^0$
$\mathbf{W}_1, \mathbf{B}_1$
$\mathbf{Z}_1^0$
$\phi_2(\cdot)$
$\mathbf{Z}_2^0$
$\mathbf{W}_3, \mathbf{B}_3$
$\mathbf{Z}_3^0$
$\phi_4(\cdot)$
$\mathbf{Y}$
$\mathbf{G}_0^+(\cdot)$
$\widehat{\mathbf{Z}}_{k0}^+$
$\mathbf{R}_{k0}^+$
$\mathbf{G}_1^\pm(\cdot)$
$\widehat{\mathbf{Z}}_{k1}^+$
$\mathbf{R}_{k1}^+$
$\mathbf{G}_2^\pm(\cdot)$
$\widehat{\mathbf{Z}}_{k2}^+$
$\mathbf{R}_{k2}^+$
$\mathbf{G}_3^\pm(\cdot)$
$\widehat{\mathbf{Z}}_{k3}^+$
$\mathbf{R}_{k3}^+$
$\mathbf{G}_4^-(\cdot)$
$\mathbf{Y}$
$\mathbf{R}_{k0}^-$
$\widehat{\mathbf{Z}}_{k0}^-$
$\mathbf{R}_{k1}^-$
$\widehat{\mathbf{Z}}_{k1}^-$
$\mathbf{R}_{k2}^-$
$\widehat{\mathbf{Z}}_{k2}^-$
$\mathbf{R}_{k3}^-$
$\widehat{\mathbf{Z}}_{k3}^-$

Figure 4.1: (TOP) The signal flow graph for *true* values of matrix variables $\{\mathbf{Z}_\ell^0\}_{\ell=0}^3$, given in eqn. (4.1) where $\mathbf{Z}_\ell^0 \in \mathbb{R}^{n_\ell \times d}$. (BOTTOM) Signal flow graph of the ML-MVAMP procedure in Algo. 4. The variables with superscript + and - are updated in the forward and backward pass respectively. ML-MVAMP (Algorithm 4) solves (4.2) by solving a sequence of simpler estimation problems over consecutive pairs $(\mathbf{Z}_\ell, \mathbf{Z}_{\ell-1})$.

In many applications, the inference problem can be performed via minimization of an appropriate cost function. For example, suppose the network (4.1) has no noise Ξ_ℓ for all layers except the final measurement layer, $\ell = L$. In this case, the $\mathbf{Z}_{L-1}^0 = \mathbf{g}(\mathbf{Z}_0^0)$ for some *deterministic function* $\mathbf{g}(\cdot)$ representing the action of the first $L-1$ layers. Inference can then be conducted via a minimization of the form,

$$\widehat{\mathbf{Z}}_{L-1} := \mathbf{g}\left(\underset{\mathbf{Z}_0}{\operatorname{argmin}}\ H_L(\mathbf{Y}, \mathbf{Z}_{L-1}) + H_0(\mathbf{Z}_0), \quad \text{subject to } \mathbf{Z}_{L-1} = \mathbf{g}(\mathbf{Z}_0)\right) \tag{4.3}$$

where the term $H_L(\mathbf{Y}, \mathbf{Z}_{L-1})$ penalizes the prediction error and $H_0(\mathbf{Z}_0)$ is an (optional) regularizer on the network input. For maximum a posteriori (MAP) estimation one takes, $H_L(\mathbf{Y}, \mathbf{Z}_{L-1}) = -\log p(\mathbf{Y}|\mathbf{Z}_{L-1})$, and $H_0(\mathbf{Z}_0) = -\log p(\mathbf{Z}_0)$, where the output probability $p(\mathbf{Y}|\mathbf{Z}_{L-1})$ is defined from the last layer of model (4.1b): $\mathbf{Y} = \mathbf{Z}_L = \phi_L(\mathbf{Z}_{L-1}, \Xi_L)$. The minimization (4.3) can then be solved using a gradient-based method. Encouraging results in image reconstruction have been demonstrated in [16, 53, 73, 97, 142, 150, 163]. Markov-chain Monte Carlo (MCMC) algorithms and Langevin diffusion [23, 160] could also be employed for more complex inference tasks.

However, rigorous analysis of these methods is difficult due to the non-convex nature of the optimization problem. To address this issue, recent works [42, 89, 115] have extended

Approximate Message Passing (AMP) methods to provide inference algorithms for the multi-layer networks. AMP was originally developed in [10, 31, 32, 71] for compressed sensing. Similar to other AMP-type results, the performance of multi-layer AMP-based inference can be precisely characterized in certain high-dimensional random instances. In addition, the mean-squared error for inference of the algorithms match predictions for the Bayes-optimal inference predicted by various techniques from statistical physics [7,47,131]. Thus, AMP-based multi-layer inference provides a computationally tractable estimation framework with precise performance guarantees and testable conditions for optimality in certain high-dimensional random settings.

Prior multi-layer AMP works [42, 61, 91, 115] have considered the case of vector-valued quantities with $d = 1$. The main contribution of this paper is to consider the *matrix-valued* case when $d > 1$. To handle the case when $d > 1$, we extend the Multi-Layer Vector Approximate Message Passing (ML-VAMP) algorithm of [42, 115] to the matrix case. The ML-VAMP method is based on VAMP method of [127], which is closely related to expectation propagation (EP) [96, 145], expectation-consistent approximate inference (EC) [45, 111], S-AMP [19], and orthogonal AMP [86]. We will use "ML-Mat-VAMP" when referring to the matrix extension of ML-VAMP.

Contributions: First, similar to the case of ML-VAMP, we analyze ML-Mat-VAMP in a large system limit, where $n_\ell \to \infty$ and d is fixed, under rotationally invariant random weight matrices $\mathbf{W}_\ell$. In this large system limit, we prove that the mean-squared error (MSE) of the estimates of ML-Mat-VAMP can be exactly predicted by a deterministic set of equations called the *state evolution* (SE). The SE describes how the distribution of the true activations and pre-activations of the network as well as the estimated values generated by ML-Mat-VAMP evolve jointly from one iteration of the algorithm to the other. This extension of the SE equations to the matrix case is not trivial and requires considering correlation across multiple vectors. Indeed, in the case of ML-VAMP, the SE equations involve scalar quantities and

2×2 matrices. For ML-Mat-VAMP, the SE equations involve $d \times d$ and $2d \times 2d$ matrices.

Second, we show that the method can offer precise predictions in important estimation problems that are difficult to analyze via other means. The ML-VAMP was focused on deep reconstruction problems [16, 163]. The matrix version here can be applied to other classes of problems such as multi-task regression, matrix completion and learning the input layer of a neural network. Even though these networks are typically shallow (just $L = 2$ layers), there are no existing methods that can provide the same types of precise results. For example, in the case of learning the input layer of a neural network, our results can exactly predict the test error as a function of the noise statistics, activations, number of training sample and other key modeling parameters.

Notation: Boldface uppercase letters $\mathbf{X}$ denote matrices. $\mathbf{X}_{n:}$ refers to the n^{th} row of $\mathbf{X}$. Random vectors are row-vectors. For a function $f : \mathbb{R}^{1\times m} \to \mathbb{R}^{1\times k}$, its row-wise extension is represented by $\mathbf{f} : \mathbb{R}^{N\times m} \to \mathbb{R}^{N\times k}$, i.e., $[\mathbf{f}(\mathbf{X})]_{n:} = f(\mathbf{X}_{n:})$. We denote the Jacobian matrix of f by $\frac{\partial f}{\partial \boldsymbol{x}}(\boldsymbol{x}) \in \mathbb{R}^{m\times k}$, so that $[\frac{\partial f}{\partial \boldsymbol{x}}(\boldsymbol{x})]_{ij} = \frac{\partial f_i}{\partial \boldsymbol{x}_j}(\boldsymbol{x})$. For its row-wise extension $\mathbf{f}$, we denote by $\left\langle \frac{\partial \mathbf{f}}{\partial \mathbf{X}}(\mathbf{X}) \right\rangle$ the average Jacobian, i.e., $\frac{1}{N}\sum_{n=1}^{N} \frac{\partial f}{\partial \mathbf{X}_{n:}}(\mathbf{X}_{n:}) \in \mathbb{R}^{m\times k}$

4.2 Example Applications

As we describe next, the matrix estimation problem 4.2 is of broad interest and several interesting applications can be formulated under this framework. We share a few examples below.

4.2.1 Multi-task and Mixed Regression Problems

A simple application of the matrix-valued multi-layer inference problem (4.2) is for *multi-task regression* [109]. Consider a generalized linear model of the form,

$$\mathbf{Y} = \phi(\mathbf{X}\mathbf{F}; \boldsymbol{\Xi}), \tag{4.4}$$

where $\mathbf{Y} \in \boldsymbol{R}^{N\times d}$ is a matrix of measured responses, $\mathbf{X} \in \boldsymbol{R}^{N\times p}$ is a known design matrix, $\mathbf{F} \in \boldsymbol{R}^{p\times d}$ are a set regression coefficients to be estimated, and $\boldsymbol{\Xi}$ is noise. The problem can be considered as d separate regression problems – one for each column. However, in some applications, these design "tasks" are related in such a way that it benefits to *jointly* estimate the predictors. To do this, it is common to solve an optimization problem of the form

$$\underset{\mathbf{F}}{\operatorname{argmin}}\left\{ \sum_{j=1}^{d}\sum_{i=1}^{N} L(y_{ij}, [\mathbf{XF}]_{ij}) + \lambda \sum_{k=1}^{p} \rho(\mathbf{F}_{k:}) \right\}, \tag{4.5}$$

where $L(\cdot)$ is a loss function, and $\rho(\cdot)$ is a regularizer that acts on the rows $\mathbf{F}_{k:}$ of $\mathbf{F}$ to couple the prediction coefficients across tasks. For example, the multi-task LASSO [109] uses loss $L(y,z) = (y-z)^2$ and regularization $\rho(\mathbf{F}_{k:}) = \|\mathbf{F}_{k:}\|_2$ to enforce row-sparsity in $\mathbf{F}$. In the compressive-sensing context, multi-task regression is known as the "multiple measurement vector" (MMV) problem, with applications in MEG reconstruction [25], DoA estimation [152], and parallel MRI [83]. An AMP approach to the MMV problem was developed in [167]. The multi-task model (4.4) can be immediately written as a multi-layer network (4.1) by setting: $\mathbf{Z}_0 := \mathbf{F}, \mathbf{W}_0 := \mathbf{X}, \mathbf{Z}_1 := \mathbf{W}_0\mathbf{Z}_0 = \mathbf{XF}, \mathbf{Y} = \mathbf{Z}_2 := \phi(\mathbf{Z}_1, \boldsymbol{\Xi})$. Also, by appropriately setting the prior $p(\mathbf{Z}_0)$, the multi-layer matrix MAP inference (4.3) will match the multi-task optimization (4.5).

In (4.5), the regularization couples the columns of $\mathbf{F}$ but the loss term couples its rows. In *mixed regression* problems, the loss couples the columns of $\mathbf{F}$. For example, consider designing predictors $\mathbf{F} = [\mathbf{f}_1, \mathbf{f}_2]$ for *mixed linear regression* [164], i.e.,

$$y_i = q_i \mathbf{x}_i^{\mathsf{T}} \mathbf{f}_1 + (1-q_i)\mathbf{x}_i^{\mathsf{T}} \mathbf{f}_2 + v_i, \quad q_i \in \{0,1\}, \tag{4.6}$$

where $i = 1,\ldots,N$ and the ith response comes from one of two linear models, but which model is not known. This setting can be modeled by a different output mapping: As before, set $\mathbf{Z}_0 := \mathbf{F}$, $\mathbf{Z}_1 = \mathbf{XF}$ and let the noise in the output layer be $\boldsymbol{\Xi}_1 = [\mathbf{q}, \mathbf{v}]$ which includes the additive noise v_i in (4.6) and the random selection variable q_i. Then, we can write (4.6) via an appropriate function, $\mathbf{y} = \phi_1(\mathbf{Z}_1, \boldsymbol{\Xi}_1)$.

4.2.2 Sketched Clustering

A related problem arises in *sketched clustering* [74], where a massive dataset is nonlinearly compressed down to a short vector $\mathbf{y} \in \mathbb{R}^n$, from which cluster centroids $\mathbf{f}_k \in \mathbb{R}^p$, for $k = 1, \ldots, d$, are then extracted. This problem can be approached via the optimization [75] $\min_{\boldsymbol{\alpha} \geq \mathbf{0}} \min_{\mathbf{F}} \sum_{i=1}^{n} \left| y_i - \sum_{j=1}^{d} \alpha_j e^{\sqrt{-1}\mathbf{x}_i^{\mathsf{T}}\mathbf{f}_j} \right|^2$ where $\mathbf{x}_i \in \mathbb{R}^p$ are known i.i.d. Gaussian vectors. An AMP approach to sketched clustering was developed in [18]. For known $\boldsymbol{\alpha}$, the minimization corresponds to MAP estimation with the multi-layer matrix model with $\mathbf{Z}_0 = \mathbf{F}$, $\mathbf{W}_1 = \mathbf{X}$ $\mathbf{Z}_1 = \mathbf{X}\mathbf{F}$ and using the output mapping, $\phi_1(\mathbf{Z}_1, \Xi) := \sum_{j=1}^{d} \alpha_j e^{\sqrt{-1}\mathbf{Z}_{1,:j}} + \Xi$, where the exponential is applied elementwise and Ξ is i.i.d. Gaussian. The mapping ϕ_1 operates row-wise on $\mathbf{Z}_1$ and Ξ.

4.2.3 Learning the Input Layer of a Two-Layer Neural Network

The matrix inference problem (4.2) can also be applied to learning the input layer weights in a two-layer neural network (NN). Let $\mathbf{X} \in \boldsymbol{R}^{N \times N_{\text{in}}}$ and $\mathbf{Y} \in \boldsymbol{R}^{N \times N_{\text{out}}}$ be training data corresponding to N data samples. Consider the two-layer NN model,

$$\mathbf{Y} = \sigma(\mathbf{X}\boldsymbol{F}_1)\boldsymbol{F}_2 + \Xi, \tag{4.7}$$

with weight matrices $(\boldsymbol{F}_1, \boldsymbol{F}_2)$, componentwise activation function $\sigma(\cdot)$, and noise Ξ. In (4.7), the bias terms are omitted for simplicity. We used the notation "$\boldsymbol{F}_\ell$" for the weights, instead of the standard notation "$\boldsymbol{W}_\ell$," to avoid confusion when (4.7) is mapped to the multi-layer inference network (4.2). Now, our critical assumption is that the weights in the second layer, $\boldsymbol{F}_2$, are known. The goal is to learn only the weights of the first layer, $\boldsymbol{F}_1 \in \mathbb{R}^{N_{\text{in}} \times N_{\text{hid}}}$, from a dataset of N samples $(\mathbf{X}, \mathbf{Y})$.

If the activation is ReLU, i.e., $\sigma(\boldsymbol{H}) = \max_{\{} \boldsymbol{H}, 0\}$ and $\mathbf{Y}$ has a single column (i.e. scalar output per sample), and $\boldsymbol{F}_2$ has all positive entries, we can, without loss of generality, treat the weights $\boldsymbol{F}_2$ as fixed, since they can always be absorbed into the weights $\boldsymbol{F}_1$. In this case,

$\mathbf{y}$ and $\mathbf{F}_2$ are vectors and we can write the ith entry of $\mathbf{y}$ as

$$y_i = \sum_{j=1}^{d} F_{2j}\sigma([\mathbf{X}\boldsymbol{F}_1]_{ij}) + \xi_i = \sum_{j=1}^{d} \sigma([\mathbf{X}\boldsymbol{F}_1]_{ij}F_{2j}) + \xi_i \tag{4.8}$$

Thus, we can assume, without loss of generality, that $\mathbf{F}_2$ is all ones. The parameterization (4.8) is sometimes referred to as the *committee machine* [149]. The committee machine has been recently studied by AMP methods [5] and mean-field methods [94] as a way to understand the dynamics of learning.

To pose the two-layer learning problem as multi-layer inference, define $\boldsymbol{Z}_0 := \boldsymbol{F}_1, \quad \boldsymbol{W}_1 := \mathbf{X}, \quad \boldsymbol{Z}_1 := \mathbf{X}\boldsymbol{F}_1 \quad \Xi_2 := \Xi$, then $\boldsymbol{Y} = \boldsymbol{Z}_2$, where $\boldsymbol{Z}_2$ is the output of a 2-layer inference network of the form in (4.1):

$$\mathbf{Y} = \boldsymbol{Z}_2 = \phi_2(\boldsymbol{Z}_1, \Xi_2) := \sigma(\boldsymbol{Z}_1)\boldsymbol{F}_2 + \Xi_2. \tag{4.9}$$

Note that $\boldsymbol{W}_1$ is known. Also, since we have assumed that $\boldsymbol{F}_2$ is known, the function ϕ_2 is known. Finally, the function ϕ_2 is row-wise separable on both inputs. Thus, the problem of learning the input weights $\boldsymbol{F}_1$ is equivalent to learning the input $\boldsymbol{Z}_0$ of the network (4.9).

4.2.4 Model-Based Matrix completion

Consider an observed matrix $\mathbf{Y} = \mathbf{Z}_L \in \mathbb{R}^{N_L \times d}$ with missing entries $\Omega^c \in [N_L] \times [d]$. The problem is to impute the missing entries of $\mathbf{Y}$. This is an important problem in several applications ranging from recommendation systems, genomics, bioinformatics and more broadly analysis of tabular data. There have been several approaches to solving this data imputation problem, right from 0 imputation and mean imputation to more sophisticated techniques based on generative models.

Consider a generative model based on a multi-layer perceptron as in (4.1) such that the output $\mathbf{Z}_{L-1}$ models the uncorrupted data matrix. Then the imputation problem can be posed as the solution of the MAP optimization problem:

$$\underset{\{\mathbf{Z}_\ell\}_{\ell=0}^{L}}{\text{minimize}} \, \|\mathbf{Y} - \mathbf{Z}_{L-1}\|_\Omega^2 - \log \mathbb{P}(\mathbf{Z}_{L-1}, \mathbf{Z}_{L-2}, \ldots, \mathbf{Z}_0) \tag{4.10}$$

where $\|\mathbf{Y} - \mathbf{Z}_{L-1}\|_\Omega^2 = \sum_{(i,j)\in\Omega}((\mathbf{Y})_{ij} - (\mathbf{Z}_{L-1})_{ij})^2$. One can also similarly construct Bayes estimators such as $\mathbb{E}[\mathbf{Z}_{L-1}|\mathbf{Z}_L]$.

Traditional approaches to matrix completion have looked at regularized convex minimization schemes just like (4.10) where $-\log \mathbb{P}(\mathbf{Z}_{L-1}) = \|\mathbf{Z}_{L-1}\|_*$, which is the nuclear norm, or some other structure inducing convex norms. While the term $-\log \mathbb{P}(\ldots)$ in (4.10) can be thought of as a more general regularization term, this formulation allows for more general application problems with heterogeneous variables.

For example, in imputation of tabular data, it is often the case that some columns correspond to continuous valued variables, whereas other variables are discrete valued modeling Yes/No answers or count data. In such scenarios the $-\log \mathbb{P}(\mathbf{Z}_{L-1}, \ldots)$ allows more flexibility towards modeling using GLMs and other exponential family distributions for every column separately. One simple instance of (4.10) would be a generative model $-\log \mathbb{P}(\mathbf{Z}_{L-1}, \ldots, \mathbf{Z}_0)$ which is trained on some fully observed data $\mathbf{Z}_{L-1}$ using unsupervised learning methods such as Variational Autoencoders (VAE) and Generative Adversarial Networks (GAN).

4.3 Multi-layer Matrix VAMP

4.3.1 MAP and MMSE inference

Observe that the equations (4.1) define a Markov chain over these signals and thus the posterior $p(\mathbf{Z}|\mathbf{Z}_L)$ factorizes as $p(\mathbf{Z}|\mathbf{Z}_L) \propto p(\mathbf{Z}_0) \prod_{\ell=1}^{L-1} p(\mathbf{Z}_\ell|\mathbf{Z}_{\ell-1})\, p(\mathbf{Y}|\mathbf{Z}_{L-1})$. where recall the notation $\mathbf{Z}$ from (4.2). The transition probabilities $p(\mathbf{Z}_\ell|\mathbf{Z}_{\ell-1})$ above are implicitly defined in equation (4.1) and depend on the statistics of noise terms $\mathbf{\Xi}_\ell$. We consider both maximum *a* *posteriori* (MAP) and minimum mean squared error (MMSE) estimation for this posterior:

$$\hat{\mathbf{Z}}_{\mathsf{map}} = \underset{\mathbf{Z}}{\operatorname{argmax}}\ p(\mathbf{Z}|\mathbf{Z}_L) \qquad \hat{\mathbf{Z}}_{\mathsf{mmse}} = \mathbb{E}[\mathbf{Z}|\mathbf{Z}_L] = \int \mathbf{Z}\, p(\mathbf{Z}|\mathbf{Z}_L)\, \mathrm{d}\mathbf{Z} \tag{4.11}$$

4.3.2 Algorithm Details

The ML-Mat-VAMP for approximately computing the MAP and MMSE estimates is similar to the ML-VAMP method in [42, 112]. The specific iterations of ML-Mat-VAMP algorithm are shown in Algorithm 4. The algorithm produces estimates by a sequence of forward and backward pass updates denoted by superscripts $^+$ and $^-$ respectively. The estimates $\widehat{\mathbf{Z}}_\ell^\pm$ are constructed by solving sequential problems $\mathbf{Z} = \{\mathbf{Z}_\ell\}_{\ell=0}^{L-1}$ into a sequence of smaller problems each involving estimation of a single activation or preactivation $\mathbf{Z}_\ell$ via *estimation functions* $\{\mathbf{G}_\ell^\pm(\cdot)\}_{\ell=1}^{L-1}$ which are selected depending on whether one is interested in MAP or MMSE estimation.

To describe the estimation functions, we use the notation that, for a positive definite matrix $\mathbf{\Gamma}$, define the inner product $\langle \mathbf{A}, \mathbf{B} \rangle_\Gamma := \mathrm{Tr}(\mathbf{A}^\mathsf{T}\mathbf{B}\mathbf{\Gamma})$ and let $\|\mathbf{A}\|_{\mathbf{\Gamma}}$ denote the norm induced by this inner product. For $\ell = 1, \ldots, L-1$ define the approximate belief functions

$$b_\ell(\mathbf{Z}_\ell, \mathbf{Z}_{\ell-1}|\mathbf{R}_\ell^-, \mathbf{R}_{\ell-1}^+, \mathbf{\Gamma}_\ell^-, \mathbf{\Gamma}_{\ell-1}^+) \propto p(\mathbf{Z}_\ell|\mathbf{Z}_{\ell-1}) e^{-\frac{1}{2}\left\|\mathbf{Z}_\ell - \mathbf{R}_\ell^-\right\|^2_{\mathbf{\Gamma}_\ell^-} - \frac{1}{2}\left\|\mathbf{Z}_{\ell-1} - \mathbf{R}_{\ell-1}^+\right\|^2_{\mathbf{\Gamma}_{\ell-1}^+}}, \tag{4.12}$$

where $\mathbf{Z}_\ell, \mathbf{R}_\ell^\pm \in \mathbb{R}^{n_\ell \times d}$ and $\mathbf{\Gamma}_\ell^\pm \in \mathbb{R}^{d \times d}$ for all $\ell = 0, 1, \ldots L$. Define $b_0(\mathbf{Z}_0|\mathbf{R}_0^-, \mathbf{\Gamma}_0^-)$ and $b_L(\mathbf{Z}_{L-1}|\mathbf{R}_{L-1}^+, \mathbf{\Gamma}_{L-1}^+)$ similarly. The MAP and MMSE estimation functions are then given by the MAP and MMSE estimates for these belief densities,

$$\mathbf{G}_{\ell,\mathsf{map}}^\pm = (\widehat{\mathbf{Z}}_\ell^+, \widehat{\mathbf{Z}}_{\ell-1}^-) = \operatorname{argmax}\ b_\ell(\mathbf{Z}_\ell, \mathbf{Z}_{\ell-1}) \qquad \mathbf{G}_{\ell,\mathsf{mmse}}^\pm = (\widehat{\mathbf{Z}}_\ell^+, \widehat{\mathbf{Z}}_{\ell-1}^-) = \mathbb{E}[(\mathbf{Z}_\ell, \mathbf{Z}_{\ell-1})|b_\ell] \tag{4.13}$$

where the expectation is with respect to the normalized density proportional to b_ℓ. Thus, the ML-Mat-VAMP algorithm reduces the joint estimation of the vectors $(\mathbf{Z}_0, \ldots, \mathbf{Z}_{L-1})$ to a sequence of simpler estimations on sub-problems with terms $(\mathbf{Z}_{\ell-1}, \mathbf{Z}_\ell)$. We refer to these subproblems as denoisers and denote their solutions by $\mathbf{G}_\ell^\pm$, so that $\widehat{\mathbf{Z}}_\ell^+ = \mathbf{G}_\ell^+$ and $\widehat{\mathbf{Z}}_{\ell-1}^- = \mathbf{G}_\ell^-$ corresponding to lines 9 and 21 of Algorithm 4. The denoisers $\mathbf{G}_0^+$ and $\mathbf{G}_L^-$, which provide updates to $\widehat{\mathbf{Z}}_0^+$ and $\widehat{\mathbf{Z}}_{L-1}^-$, are defined in a similar manner via b_0 and b_L respectively.

The estimation functions (4.13) can be easily computed for the multi-layer matrix network. An important characteristic of these estimators is that they can be computed using maps which are row-wise separable over their inputs and hence are easily parallelizable. To simplify notation, we denote the precision parameters for denoisers $\mathbf{G}_\ell^\pm$ in the k^{th} iteration by

$$\mathbf{\Theta}_{k\ell}^+ := (\mathbf{\Gamma}_{k\ell}^-, \mathbf{\Gamma}_{k,\ell-1}^+), \quad \mathbf{\Theta}_{k\ell}^- := (\mathbf{\Gamma}_{k+1,\ell}^-, \mathbf{\Gamma}_{k,\ell-1}^+), \qquad \mathbf{\Theta}_{k0}^+ := \mathbf{\Gamma}_{k0}^-, \qquad \mathbf{\Theta}_{kL}^- := \mathbf{\Gamma}_{k,L-1}^+. \tag{4.14}$$

Non-linear layers: For ℓ even, since the rows of $\mathbf{\Xi}_\ell$ are i.i.d., the belief density $b_\ell(\mathbf{Z}_\ell, \mathbf{Z}_{\ell-1}|\cdot)$ from (4.12) factors as a product across rows, $b_\ell(\mathbf{Z}_\ell, \mathbf{Z}_{\ell-1}) = \prod_n b_\ell([\mathbf{Z}_\ell]_{n:}, [\mathbf{Z}_{\ell-1}]_{n:})$. Thus, the MAP and MMSE estimates (4.13) can be performed over d-dimensional variables where d is the number of entries in each row. There is no joint estimation across the different n_ℓ rows.

Linear layers: When ℓ is odd, the density $b_\ell(\mathbf{Z}_\ell, \mathbf{Z}_{\ell-1}|\cdot)$ in (4.12) is a Gaussian. Hence, the MAP and MMSE estimates agree and can be computed via least squares. Although for linear layers $[\mathbf{G}_\ell^+, \mathbf{G}_\ell^-](\mathbf{R}_\ell^-, \mathbf{R}_{\ell-1}^+, \mathbf{\Theta}_\ell)$ is not row-wise separable over $(\mathbf{R}_\ell^-, \mathbf{R}_{\ell-1})$, it can be computed using another row-wise denoiser $[\widetilde{\mathbf{G}}_\ell^+, \widetilde{\mathbf{G}}_\ell^-]$ via the SVD of the weight matrix $\mathbf{W}_\ell = \mathbf{V}_\ell \operatorname{Diag}(\mathbf{S}_\ell)\mathbf{V}_{\ell-1}$ as follows. Note that the SVD is only needed to be performed once.:

$$\begin{aligned}
[\mathbf{G}_\ell^+, \mathbf{G}_\ell^-](\mathbf{R}_\ell, \mathbf{R}_{\ell-1}, \mathbf{\Theta}_\ell) &= \operatorname*{argmax}_{\mathbf{Z}_\ell, \mathbf{Z}_{\ell-1}} \|\mathbf{Z}_\ell - \mathbf{W}_\ell \mathbf{Z}_{\ell-1} - \mathbf{B}_\ell\|_{\mathbf{N}_\ell}^2 + \left\|\mathbf{Z}_\ell - \mathbf{R}_\ell^-\right\|_{\mathbf{\Gamma}_\ell^-}^2 + \left\|\mathbf{Z}_{\ell-1} - \mathbf{R}_{\ell-1}^+\right\|_{\mathbf{\Gamma}_{\ell-1}^+}^2 \\
&\overset{(a)}{=} \operatorname*{argmax}_{\mathbf{Z}_\ell, \mathbf{Z}_{\ell-1}} \left\|\mathbf{V}_\ell^\mathsf{T} \mathbf{Z}_\ell - \operatorname{Diag}(\mathbf{S}_\ell)\mathbf{V}_{\ell-1}\mathbf{Z}_{\ell-1} - \mathbf{V}_\ell^\mathsf{T}\mathbf{B}_\ell\right\|_{\mathbf{N}_\ell}^2 + \left\|\mathbf{V}_\ell^\mathsf{T}\mathbf{Z}_\ell - \mathbf{V}_\ell^\mathsf{T}\mathbf{R}_\ell^-\right\|_{\mathbf{\Gamma}_\ell^-}^2 + \left\|\mathbf{V}_{\ell-1}\mathbf{Z}_{\ell-1} - \mathbf{V}_{\ell-1}\mathbf{R}_{\ell-1}^+\right\|_{\mathbf{\Gamma}_{\ell-1}^+}^2 \\
&\overset{(b)}{=} [\mathbf{V}_\ell^\mathsf{T}\widetilde{\mathbf{G}}_\ell^+, \mathbf{V}_{\ell-1}\widetilde{\mathbf{G}}_\ell^-](\boldsymbol{V}_\ell^\mathsf{T}\mathbf{R}_\ell, \boldsymbol{V}_{\ell-1}\mathbf{R}_{\ell-1}, \mathbf{\Theta}_\ell)
\end{aligned}$$

where (a) follows from the rotational invariance of the norm, and (b) follows from the definition of denoiser $[\widetilde{\mathbf{G}}_\ell^+, \widetilde{\mathbf{G}}_\ell^-](\widetilde{\mathbf{R}}_\ell^-, \widetilde{\mathbf{R}}_{\ell-1}^+, \mathbf{\Theta}_\ell)$ given below

$$[\widetilde{\mathbf{G}}_\ell^+, \widetilde{\mathbf{G}}_\ell^-] := \operatorname*{argmax}_{\widetilde{\mathbf{Z}}_\ell, \widetilde{\mathbf{Z}}_{\ell-1}} \left\|\widetilde{\mathbf{Z}}_\ell - \operatorname{Diag}(\mathbf{S}_\ell)\widetilde{\mathbf{Z}}_{\ell-1} - \widetilde{\mathbf{B}}_\ell\right\|_{\mathbf{N}_\ell}^2 \left\|\widetilde{\mathbf{Z}}_\ell - \widetilde{\mathbf{R}}_\ell^-\right\|_{\mathbf{\Gamma}_\ell^-}^2 + \left\|\widetilde{\mathbf{Z}}_{\ell-1} - \widetilde{\mathbf{R}}_{\ell-1}^+\right\|_{\mathbf{\Gamma}_{\ell-1}^+}^2 \tag{4.15}$$

Note that the optimization problem in (4.15), is decomposable accross the rows of variables $\widetilde{\mathbf{Z}}_\ell$ and $\widetilde{\mathbf{Z}}_{\ell-1}$, and hence $[\widetilde{\mathbf{G}}_\ell^+, \widetilde{\mathbf{G}}_\ell^-]$ operates row-wise on its inputs.

Fixed Points: We note that the fixed points of the ML-Mat-VAMP algorithm can be shown to be KKT points of the variational formulations of (4.11), omitted here due to lack of space. This is a direct extention of results from Section 3 of [115]. In particular, we can show that the ML-Mat-VAMP in the MAP inference case is a preconditioned *Peaceman-Rachford splitting* ADMM type algorithm [147].

4.4 Analysis in the Large System Limit

We follow the analysis framework of the ML-VAMP work [42,112], which is itself based on the original AMP analysis in [10]. This analysis is based on considering the asymptotics of certain large random problem instances. We essentially show that under certain assumptions, as the dimension goes to infinity the behavior of the ML-Mat-VAMP algorithm can be characterized by a set of equations that describe how the distribution of rows of hidden matrices evolve at each iteration of the algorithm for all the layers. Specifically, we consider a sequence of problems (4.1) indexed by N such that for each problem the dimensions $n_\ell(N)$ are growing so that $\lim_{N\to\infty} \frac{n_\ell}{N} = \beta_\ell \in (0, \infty)$ are scalar constants. Note that d is finite and does not grow with N.

Distributions of weight matrices: For $\ell = 1, 3, \dots, L-1$, we assume that the weight matrices $\mathbf{W}_\ell$ are generated via the singular value decomposition, $\mathbf{W}_\ell = \mathbf{V}_\ell \operatorname{Diag}(\mathbf{S}_\ell)\mathbf{V}_{\ell-1}$ where $\boldsymbol{V}_\ell \in \mathbb{R}^{n_\ell \times n_\ell}$ are Haar distributed over orthonormal matrices and $\mathbf{S}_\ell = (s_{\ell,1}, \dots, s_{\ell,\min\{n_\ell, n_{\ell-1}\}})$. We will describe the distribution of the components $\mathbf{S}_\ell$ momentarily.

Assumption on Denoisers: We assume that the non-linear denoisers $\mathbf{G}^{\pm}_{2k}$ act row-wise on their inputs $(\mathbf{R}^-_{2k}, \mathbf{R}^+_{2k-1})$. Further these operators and their Jacobian matrices $\frac{\partial \mathbf{G}^+_{2k}}{\partial \mathbf{R}^-_{2k}}, \frac{\partial \mathbf{G}^-_{2k}}{\partial \mathbf{R}^+_{2k-1}}, \frac{\partial \mathbf{G}^+_{0}}{\partial \mathbf{R}^-_{0}}, \frac{\partial \mathbf{G}^-_{L}}{\partial \mathbf{R}^+_{L-1}}$ are *uniformly Lipschitz continuous*, the definition of which is provided in B.2.

Assumption on initialization, true variables: The distribution of the remaining variables is described by a weak limit: For a matrix sequence $\boldsymbol{X} := \boldsymbol{X}(N) \in \mathbb{R}^{N\times d}$, by the notation $\boldsymbol{X} \overset{2}{\Rightarrow} X$ we mean that there exists a random variable X in $\mathbb{R}^d$ with $\mathbb{E}\|X\|^2 < \infty$ such that $\lim\limits_{N\to\infty} \frac{1}{N}\sum_{i=1}^N \psi(\boldsymbol{X}_{i:}) = \mathbb{E}\,\psi(X)$ almost surely, for any bounded continuous function $\psi : \mathbb{R}^d \to \mathbb{R}$, as well as for quadratic functions $\boldsymbol{x}^\top \boldsymbol{P}\boldsymbol{x}$ for any $\boldsymbol{P} \in \mathbb{R}^{d\times d}_{\succeq 0}$. This is also referred to as Wasserstein-2 convergence [99]. For e.g., this property is satisfied for a random $\boldsymbol{X}$ with i.i.d. rows with bounded second moments, but is more general, since it applies to deterministic matrix sequences as well. More details on this weak limit are given in B.2.

Let $\overline{\mathbf{B}}_\ell := \mathbf{V}_\ell^\top \mathbf{B}_\ell$, and $\overline{\mathbf{S}}_\ell \in \mathbb{R}^{n_\ell}$ be the zero-padded vector of singular values of $\mathbf{W}_\ell$, and let $\boldsymbol{\tau}^-_{0\ell} \in \mathbb{R}^{d\times d}_{\succ 0}$. Then we assume that the following empirical convergences hold.$(\mathbf{\Xi}_\ell, \mathbf{R}^-_{0\ell} - \mathbf{Z}^0_\ell) \overset{2}{\Rightarrow} (\Xi_\ell, Q^-_{0\ell})$ for even ℓ and $(\overline{\mathbf{S}}_\ell, \overline{\mathbf{B}}_\ell, \mathbf{\Xi}_\ell, \mathbf{V}_\ell^\top(\mathbf{R}^-_{0\ell} - \mathbf{Z}^0_\ell)) \overset{2}{\Rightarrow} (S_\ell, \overline{B}_\ell, \Xi_\ell, Q^-_{0\ell})$, for odd ℓ. Here $S_\ell \in \mathbb{R}_{\geq 0}$ is bounded, $\overline{B}_\ell \in \mathbb{R}^d$ is bounded, $\Xi_{2\ell-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{N}^{-1}_{2\ell-1})$, and $Q^-_{0\ell} \sim \mathcal{N}(\mathbf{0}, \overline{\Gamma}^-_{0\ell})$, for $\ell = 0, 1, \ldots, L-1$ are all pairwise independent random variables. Additionally, we assume that $\mathbf{Z}^0_0 \overset{2}{\Rightarrow} Z^0$ and that the sequence of initial matrices $\{\mathbf{\Gamma}^-_{0\ell}\}$ satisfies the following pointwise convergence

$$\mathbf{\Gamma}^-_{0\ell}(N) \to \overline{\mathbf{\Gamma}}^-_{0\ell}, \quad \ell = 0, 1, \ldots, L-1 \tag{4.16}$$

4.4.1 Main Result

The main result of this paper concerns the empirical distribution of the rows $[\widehat{\mathbf{Z}}^\pm_\ell]_{n:}, [\mathbf{R}^\pm_\ell]_{n:}$ of the iterates of Algorithm 4. It characterizes the asymptotic behaviour of these empirical distributions in terms of d-dimensional random vectors which are either Gaussians or functions of Gaussians. Let $G^\pm_\ell$ denote maps $\mathbb{R}^{1\times d} \to \mathbb{R}^{1\times d}$, such that (4.13), i.e., $[\mathbf{G}^\pm_\ell(\mathbf{R}^-_\ell, \mathbf{R}^+_{\ell-1}, \mathbf{\Theta})]_{n:} = G^\pm_\ell([\mathbf{R}^-_\ell]_{n:}, [\mathbf{R}^+_{\ell-1}]_{n:}, \mathbf{\Theta})$. Having stated the requisite definitions and assumptions, we can now state our main result.

Theorem 9. *For a fixed iteration index $k \geq 0$, there exist deterministic matrices $\mathbf{K}^+_{k\ell} \in$*

$\mathbb{R}^{2d\times 2d}_{\succ 0}$*, and* $\boldsymbol{\tau}^-_{k\ell}, \overline{\boldsymbol{\Gamma}}^+_{k\ell}$ *and* $\overline{\boldsymbol{\Gamma}}^-_{k\ell}, \in \mathbb{R}^{d\times d}_{\succ 0}$ *such that for even* ℓ*:*

$$\left(\mathbf{Z}^0_{\ell-1}, \mathbf{Z}^0_\ell, \widehat{\mathbf{Z}}^-_{k,\ell-1}, \widehat{\mathbf{Z}}^+_{k\ell}\right) \overset{2}{\Rightarrow} \left(\mathsf{A}, \widetilde{\mathsf{A}}, G^-_\ell(\mathsf{C}+\widetilde{\mathsf{A}}, \mathsf{B}+\mathsf{A}, \overline{\boldsymbol{\Gamma}}^-_{k\ell}, \overline{\boldsymbol{\Gamma}}^+_{k,\ell-1}), G^+_\ell(\mathsf{C}+\widetilde{\mathsf{A}}, \mathsf{B}+\mathsf{A}, \overline{\boldsymbol{\Gamma}}^-_{k\ell}, \overline{\boldsymbol{\Gamma}}^+_{k,\ell-1})\right)$$

where $(\mathsf{A},\mathsf{B}) \sim \mathcal{N}(\mathbf{0}, \mathbf{K}^+_{k,\ell-1})$*,* $\mathsf{C} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\tau}^-_{k\ell})$*,* $\widetilde{\mathsf{A}} = \phi_\ell(\mathsf{A}, \Xi_\ell)$ *and* $(\mathsf{A},\mathsf{B}), \mathsf{C}$ *are independent. For* $\ell = 0$*, the same result holds where the* 1st *and* 3rd *terms are dropped, whereas for* $\ell = L$*, the* 2nd *and* 4th *terms are dropped. Similarly, for odd* ℓ*:*

$$\left(\mathbf{V}^\mathsf{T}_{\ell-1}\mathbf{Z}^0_{\ell-1},\ \mathbf{V}^\mathsf{T}_{\ell-1}\mathbf{Z}^0_\ell,\ \mathbf{V}_\ell\widehat{\mathbf{Z}}^-_{k,\ell-1},\ \mathbf{V}_\ell\widehat{\mathbf{Z}}^+_{k\ell}\right) \overset{2}{\Rightarrow}$$
$$\left(\mathsf{A},\ \widetilde{\mathsf{A}},\ G^-_\ell(\mathsf{C}+\widetilde{\mathsf{A}}, \mathsf{B}+\mathsf{A}, \overline{\boldsymbol{\Gamma}}^-_{k\ell}, \overline{\boldsymbol{\Gamma}}^+_{k,\ell-1}),\ G^+_\ell(\mathsf{C}+\widetilde{\mathsf{A}}, \mathsf{B}+\mathsf{A}, \overline{\boldsymbol{\Gamma}}^-_{k\ell}, \overline{\boldsymbol{\Gamma}}^+_{k,\ell-1})\right)$$

where $(\mathsf{A},\mathsf{B}) \sim \mathcal{N}(\mathbf{0}, \mathbf{K}^+_{k,\ell-1})$*,* $\mathsf{C} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\tau}^-_{k\ell})$*,* $\widetilde{\mathsf{A}} = S_\ell\,\mathsf{A} + \overline{B}_\ell + \Xi_\ell$ *and* $(\mathsf{A},\mathsf{B}), \mathsf{C}$ *are independent.*

Furthermore for $\ell = 0, 1, \ldots L-1$*, we have*

$$(\boldsymbol{\Gamma}^\pm_{k\ell}, \boldsymbol{\Lambda}^\pm_{k\ell}) \xrightarrow{a.s.} (\overline{\boldsymbol{\Gamma}}^\pm_{k\ell}, \overline{\boldsymbol{\Lambda}}^\pm_{k\ell}).$$

The parameters in the distribution, $\{\mathbf{K}^+_{k\ell}, \boldsymbol{\tau}^-_{k\ell}, \overline{\boldsymbol{\Gamma}}^\pm_{k\ell}, \overline{\boldsymbol{\Lambda}}^\pm_{k\ell}\}$ are deterministic and can be computed via a set of recursive equations called the *state evolution* or SE. The SE equations are provided in B.1 The result is similar to those for ML-VAMP in [42, 115] except that the SE equations for ML-Mat-VAMP involve $d \times d$ and $2d \times 2d$ matrix terms; the ML-VAMP SE only requires scalar and 2×2 matrix terms. The result holds for both MAP inference and MMSE inference, the only difference is implicit, i.e., the choice of denoiser $\mathbf{G}_\ell(\cdot)$ from eqn. (4.13).

The importance of Theorem 9 is that the rows of the iterates of the ML-Mat-VAMP Algorithm ($\widehat{\mathbf{Z}}^-_{k,\ell-1}, \widehat{\mathbf{Z}}^+_{k\ell}$ in Algorithm 4) and the rows of the corresponding true values, $\mathbf{Z}^0_{\ell-1}, \mathbf{Z}^0_\ell$, have a simple, asymptotic random vector description of a typical row. We will call this the "row-wise" model. According to this model, for even ℓ, the rows of $\mathbf{Z}^0_{\ell-1}$ converge to a Gaussian $\mathsf{A} \in \boldsymbol{R}^d$ and the rows of $\mathbf{Z}^0_\ell$ converge to the output of the Gaussian through the row-wise function ϕ_ℓ, $\widetilde{\mathsf{A}} = \phi_\ell(\mathsf{A}, \Xi_\ell)$. Then the rows of the estimates $\widehat{\mathbf{Z}}^-_{k,\ell-1}, \widehat{\mathbf{Z}}^+_{k\ell}$ asymptotically approach the outputs of row-wise estimation function $G^-(\cdot)$ and $G^+(\cdot)$ supplied by A and $\widetilde{\mathsf{A}}$

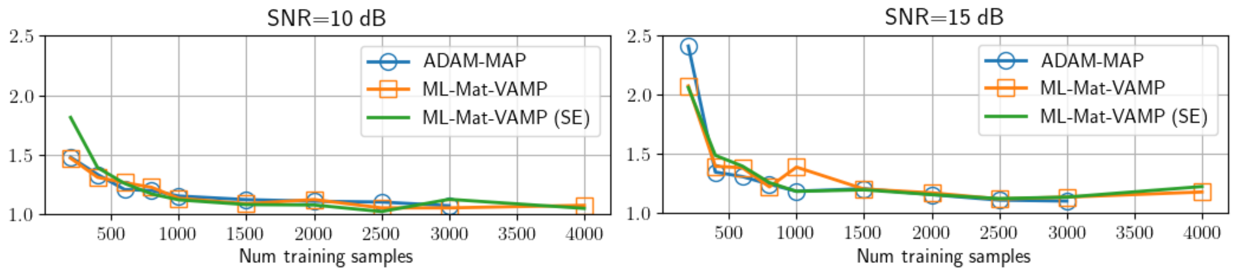


Figure 4.2: Test error in learning the first layer of a 2 layer neural network using ADAM-based gradient descent, ML-Mat-VAMP and its state evolution prediction.

corrupted with Gaussian noise. A similar convergence holds for odd ℓ.

This “row-wise" model enables exact an analysis of the performance of the estimates at each iteration. For example, to compute a weighted mean squared error (MSE) metric at iteration k, the convergence shows that,

$$\frac{1}{n_\ell}\left\|\widehat{\mathbf{Z}}^{+}_{k\ell}-\mathbf{Z}^{0}_{\ell}\right\|^{2}_{\mathbf{H}} \xrightarrow{a.s.} \mathbb{E}\|\mathbf{G}^{+}_{\ell}(\mathsf{C}+\widetilde{\mathsf{A}},\mathsf{B}+\mathsf{A},\mathbf{\Theta}_{k\ell})-\widetilde{\mathsf{A}}\|^{2}_{\mathbf{H}},$$

for even ℓ and any positive semi-definite matrix $\mathbf{H}\in \boldsymbol{R}^{d\times d}$. The norm on the left-hand above acts row-wise, $\|\mathbf{Z}\|^2_{\mathbf{H}} := \sum_i \|\mathbf{Z}_{i:}\|^2_{\mathbf{H}}$. Hence, this asymptotic MSE can be evaluated via expectations of d-dimensional variables from the SE. Similarly, one can obtain exact answers for any other row-wise performance metric of $\{(\widehat{\mathbf{Z}}^{\pm}_{k\ell},\mathbf{Z}^0_\ell)\}_\ell$ for any k.

4.5 Numerical Experiments

We consider the problem of learning the input layer of a two layer neural network as described in Section 4.2.3. We learn the weights $\boldsymbol{F}_1$ of the first layer of a two-layer network by solving problem (4.9). The large system limit analysis in this case corresponds to the input size n_{in} and number of samples N going to infinity with the number of hidden units being fixed. Our experiment take $d=4$ hidden units, $N_{\text{in}}=100$ input units, $N_{\text{out}}=1$ output unit, sigmoid activations and variable number of samples N. The weight vectors $\boldsymbol{F}_1$ and $\boldsymbol{F}_2$ are generated as i.i.d. Gaussians with zero mean and unit variance. The input $\mathbf{X}$ is also i.i.d. Gaussians with variance $1/N_{\rm in}$ so that the average pre-activation has unit variance. Output noise is added at two levels of 10 and 15 dB relative to the mean. We generate 1000 test samples and a variable number of training samples that ranges from 200 to 4000. For each trial and number of training samples, we compare three methods: (i) MAP estimation where the MAP loss function is minimized by the ADAM optimizer [76] in the Keras package of Tensorflow; (ii) Algorithm 4 run for 20 iterations and (iii) the state evolution prediction. The ADAM algorithm is run for 100 epochs with a learning rate $= 0.01$. The expectations in the SE are estimated via Monte-Carlo sampling (hence there is some variation).

Given an estimate $\widehat{\mathbf{F}}_1$ and true value $\mathbf{F}_1^0$, we can compute the test error as follows: Given a new sample $\mathbf{x}$, the true and predicted pre-activations will be $\mathbf{z}_1 = (\mathbf{F}_1^0)^{\mathsf{T}}\mathbf{x}$ and $\widehat{\mathbf{z}}_1 = \widehat{\mathbf{F}}_1^{\mathsf{T}}\mathbf{x}$. Thus, if the new sample $\mathbf{x} \sim \mathcal{N}(0, \frac{1}{N_{\rm in}}\mathbf{I})$, the true and predicted pre-activations, $(\mathbf{z}_1, \widehat{\mathbf{z}}_1)$, will be jointly Gaussian with covariance equal to the empirical $2d \times 2d$ covariance matrix of the rows of $\mathbf{F}_1^0$ and $\widehat{\mathbf{F}}_1$:

$$\mathbf{K} := \tfrac{1}{N_{\rm in}} \textstyle\sum_{k=1}^{N_{\rm in}} \mathbf{u}_k^{\mathsf{T}}\mathbf{u}_k, \quad \mathbf{u}_k = \left[\mathbf{F}_{1,k:}\ \widehat{\mathbf{F}}_{1,k:}\right] \tag{4.17}$$

From this covariance matrix, we can estimate the test error, $\mathbb{E}|y - \widehat{y}|^2 = \mathbb{E}|\mathbf{F}_2^{\mathsf{T}}(\sigma(\mathbf{z}_1) - \sigma(\widehat{\mathbf{z}}_1)|^2$, where the expectation is taken over the Gaussian $(\mathbf{z}_1, \widehat{\mathbf{z}}_1)$ with covariance $\mathbf{K}$. Also, since (4.17) is a row-wise operation, it can be predicted from the ML-Mat-VAMP SE. Thus, the SE can also predict the asymptotic test error. The normalized test error for ADAM-MAP, ML-Mat-VAMP and the ML-Mat-VAMP SE are plotted in Fig. 4.2. The normalized test error is defined as the ratio of the MSE on the test samples to the optimal MSE. Hence, a normalized MSE of one is the minimum value.

Note that since ADAM and ML-Mat-VAMP are solving the same optimization problem, they perform similarly as expected. The main message of this paper is not to develop an algorithm that outperforms ADAM, but rather an algorithm that has theoretical guarantees. The key property of ML-Mat-VAMP is that its asymptotic behavior at all the iterations can be exactly predicted by the state evolution equations. In this example, Fig. 4.2 shows that the normalized test MSE predicted via state evolution (plotted in green) matches the normalized MSE of ML-Mat-VAMP estimates (plotted in orange).

4.6 Discussion

We have developed a general framework for analyzing inference in multi-layer networks with matrix valued quantities in certain high-dimensional random settings. For learning the input layer of a two layer network, the methods enables precise predictions of the expected test error as a function of the parameter statistics, numbers of samples and noise level. This analysis can be valuable in understanding key properties such as generalization error, for example using ML-VAMP, Emami et al. [39] characterizes the generalization error of GLMs under a variety of feature distributions and train-test mismatch. Future work will look to extend these to more complex networks.

Algorithm 4 Multilayer Matrix VAMP (ML-Mat-VAMP)

```
Require: Estimators $\mathbf{G}_0^+$, $\mathbf{G}_L^-$, $\{\mathbf{G}_\ell^\pm\}_{\ell=1}^{L-1}$.
 1: Set $\mathbf{R}_{0\ell}^- = \mathbf{0} \in \mathbb{R}^{n_\ell \times d}$ and initialize $\{\mathbf{\Gamma}_{0\ell}^-\}_{\ell=0}^{L-1} \in \mathbb{R}_{\succ 0}^{d\times d}$.
 2: for $k = 0, 1, \ldots, N_{\rm it} - 1$
 3: // Forward Pass
 4: $\widehat{\mathbf{Z}}_{k0}^+ = \mathbf{G}_0^+(\mathbf{R}_{k0}^-, \mathbf{\Gamma}_{k0}^-)$
 5: $\mathbf{\Lambda}_{k0}^+ = \left\langle \frac{\partial \mathbf{G}_0^+}{\partial \mathbf{R}_0^-}(\mathbf{R}_{k0}^-, \mathbf{\Gamma}_{k0}^-) \right\rangle^{-1} \mathbf{\Gamma}_{k,0}^-,$
 6: $\mathbf{\Gamma}_{k,0}^+ = \mathbf{\Lambda}_{k,0}^+ - \mathbf{\Gamma}_{k,0}^-$
 7: $\mathbf{R}_{k,0}^+ = (\widehat{\mathbf{Z}}_{k,0}^+ \mathbf{\Lambda}_{k,0}^+ - \mathbf{R}_{k,0}^- \mathbf{\Gamma}_{k,0}^-)(\mathbf{\Gamma}_{k,0}^+)^{-1}$
 8: for $\ell = 1, \ldots, L-1$ do
 9:     $\widehat{\mathbf{Z}}_{k\ell}^+ = \mathbf{G}_\ell^+(\mathbf{R}_{k\ell}^-, \mathbf{R}_{k,\ell-1}^+, \mathbf{\Gamma}_{k\ell}^-, \mathbf{\Gamma}_{k,\ell-1}^+)$
10:     $\mathbf{\Lambda}_{k\ell}^+ = \left\langle \frac{\partial \mathbf{G}_\ell^+}{\partial \mathbf{R}_\ell^-}(\ldots) \right\rangle^{-1} \mathbf{\Gamma}_{k\ell}^-,$
11:     $\mathbf{\Gamma}_{k\ell}^+ = \mathbf{\Lambda}_{k\ell}^+ - \mathbf{\Gamma}_{k\ell}^-$
12:     $\mathbf{R}_{k\ell}^+ = (\widehat{\mathbf{Z}}_{k\ell}^+ \mathbf{\Lambda}_{k\ell}^+ - \mathbf{R}_{k\ell}^- \mathbf{\Gamma}_{k\ell}^-)(\mathbf{\Gamma}_{k\ell}^+)^{-1}$
13: end for
14:
15: // Backward Pass
16: $\widehat{\mathbf{Z}}_{k,L-1}^- = \mathbf{G}_L^-(\mathbf{R}_{k,L-1}^+, \mathbf{\Gamma}_{k,L-1}^+)$
17: $\mathbf{\Lambda}_{k,L-1}^- = \left\langle \frac{\partial \mathbf{G}_L^-}{\partial \mathbf{R}_{L-1}^+}(\mathbf{R}_{k,L-1}^+, \mathbf{\Gamma}_{k,L-1}^+) \right\rangle^{-1} \mathbf{\Gamma}_{k,L-1}^+,$
18: $\mathbf{\Gamma}_{k,L-1}^- = \mathbf{\Lambda}_{k,L-1}^- - \mathbf{\Gamma}_{k,L-1}^+$
19: $\mathbf{R}_{k+1,L-1}^- = (\widehat{\mathbf{Z}}_{k,L-1}^- \mathbf{\Lambda}_{k,L-1}^- - \mathbf{R}_{k,0}^+ \mathbf{\Gamma}_{k,0}^+)(\mathbf{\Gamma}_{k,0}^-)^{-1}$
20: for $\ell = L-1, \ldots, 1$ do
21:     $\widehat{\mathbf{Z}}_{k+1,\ell-1}^- = \mathbf{G}_\ell^-(\mathbf{R}_{k+1,\ell}^-, \mathbf{R}_{k,\ell-1}^+, \mathbf{\Gamma}_{k+1,\ell}^-, \mathbf{\Gamma}_{k,\ell-1}^+)$
22:     $\mathbf{\Lambda}_{k+1,\ell-1}^- = \left\langle \frac{\partial \mathbf{G}_\ell^-}{\partial \mathbf{R}_{\ell-1}^+}(\cdots) \right\rangle^{-1} \mathbf{\Gamma}_{k,\ell-1}^+,$
23:     $\mathbf{\Gamma}_{k+1,\ell}^- = \mathbf{\Lambda}_{k\ell}^- - \mathbf{\Gamma}_{k\ell}^+$
24:     $\mathbf{R}_{k+1,\ell-1}^- = (\widehat{\mathbf{Z}}_{k\ell}^- \mathbf{\Lambda}_{k\ell}^- - \mathbf{R}_{k\ell}^+ \mathbf{\Gamma}_{k\ell}^+)(\mathbf{\Gamma}_{k+1,\ell}^-)^{-1}$
25: end for
26: end for
```

Chapter 5

Generalization Error of Learning in Generalized Linear Models

At the heart of machine learning lies the question of generalizability of learned rules over previously unseen data. While over-parameterized models based on neural networks are now ubiquitous in machine learning applications, our understanding of their generalization capabilities is incomplete and this task is made harder by the non-convexity of the underlying learning problems. We provide a general framework to characterize the asymptotic generalization error for single-layer neural networks (i.e., generalized linear models) with arbitrary non-linearities, making it applicable to regression as well as classification problems. This framework enables analyzing the effect of (i) over-parameterization and non-linearity during modeling; (ii) choices of loss function, initialization, and regularizer during learning; and (iii) mismatch between training and test distributions. As examples, we analyze a few special cases, namely linear regression and logistic regression. We are also able to rigorously and analytically explain the *double descent* phenomenon in generalized linear models.

This chapter is based on the work [39] coauthored with Melikasadat Emami, Mojtaba Sahraee-Ardakan, Sundeep Rangan and Alyson K. Fletcher, and was published at ICML 2020.

5.1 Introduction

A fundamental goal of machine learning is *generalization*: the ability to draw inferences about unseen data from finite training examples. Methods to quantify the generalization error are therefore critical in assessing the performance of any machine learning approach.

This paper seeks to characterize the generalization error for a class of generalized linear models (GLMs) of the form

$$y = \phi_{\text{out}}(\left\langle \mathbf{x}, \mathbf{w}^0 \right\rangle, d), \tag{5.1}$$

where $\mathbf{x} \in \boldsymbol{R}^p$ is a vector of input features, y is a scalar output, $\mathbf{w}^0 \in \boldsymbol{R}^p$ are weights to be learned, $\phi_{\text{out}}(\cdot)$ is a known link function, and d is random noise. The notation $\langle \mathbf{x}, \mathbf{w}^0 \rangle$ denotes an inner product. We use the superscript "0" to denote the "true" values in contrast to estimated or postulated quantities. The output may be continuous or discrete to model either regression or classification problems.

We measure the generalization error in a standard manner: we are given training data $(\mathbf{x}_i, y_i)$, $i = 1, \ldots, N$ from which we learn some parameter estimate $\widehat{\mathbf{w}}$ via a regularized empirical risk minimization of the form

$$\widehat{\mathbf{w}} = \underset{\mathbf{w}}{\text{argmin}} F_{\text{out}}(\mathbf{y}, \mathbf{X}\mathbf{w}) + F_{\text{in}}(\mathbf{w}), \tag{5.2}$$

where $\mathbf{X} = [\mathbf{x}_1\, \mathbf{x}_2\, \ldots\, \mathbf{x}_N]^{\mathsf{T}}$, is the data matrix, F_{out} is some output loss function, and F_{in} is some regularizer on the weights. We are then given a new test sample, $\mathbf{x}_{\text{ts}}$, for which the true and predicted values are given by

$$y_{\text{ts}} = \phi_{\text{out}}(\left\langle \mathbf{x}_{\text{ts}}, \mathbf{w}^0 \right\rangle, d_{\text{ts}}), \quad \widehat{y}_{\text{ts}} = \phi(\langle \mathbf{x}_{\text{ts}}, \widehat{\mathbf{w}} \rangle), \tag{5.3}$$

where d_{ts} is the noise in the test sample, and $\phi(\cdot)$ is a postulated inverse link function that may be different from the true function $\phi_{\text{out}}(\cdot)$. The generalization error is then defined as the expectation of some expected loss between y_{ts} and $\widehat{y}_{\text{ts}}$ of the form

$$\mathbb{E}\, f_{\text{ts}}(y_{\text{ts}}, \widehat{y}_{\text{ts}}), \tag{5.4}$$

for some test loss function $f_{\text{ts}}(\cdot)$ such as squared error or prediction error.

Even for this relatively simple GLM model, the behavior of the generalization error is not fully understood. Recent works [29, 93, 99, 136] have characterized the generalization error of various linear models for classification and regression in certain large random problem instances. Specifically, the number of samples N and number of features p both grow without bound with their ratio satisfying $p/N \to \beta \in (0, \infty)$, and the samples in the training data $\mathbf{x}_i$ are drawn randomly. In this limit, the generalization error can be exactly computed. The analysis can explain the so-called *double descent* phenomena [11]: in highly under-regularized settings, the test error may initially *increase* with the number of data samples N before decreasing. See the prior work section below for more details.

Summary of Contributions. Our main result (Theorem 10) provides a procedure for exactly computing the asymptotic value of the generalization error (5.4) for GLM models in a certain random high-dimensional regime called the Large System Limit (LSL). The procedure enables the generalization error to be related to key problem parameters including the sampling ratio $\beta = p/N$, the regularizer, the output function, and the distributions of the true weights and noise. Importantly, our result holds under very general settings including:

(i) arbitrary test metrics f_{ts};

(ii) arbitrary training loss functions F_{out} as well as decomposable regularizers F_{in};

(iii) arbitrary link functions ϕ_{out};

(iv) correlated covariates $\mathbf{x}$;

(v) underparameterized ($\beta < 1$) and overparameterized regimes ($\beta > 1$); and

(vi) distributional mismatch in training and test data.

Section 5.4 discusses in detail the general assumptions on the quantities f_{ts}, F_{out}, F_{in}, and ϕ_{out} under which Theorem 10 holds.

Prior Work. Many recent works characterize generalization error of various machine learning models, including special cases of the GLM model considered here. For example, the precise characterization for asymptotics of prediction error for least squares regression has been provided in [12, 55, 102]. The former confirmed the double descent curve of [11] under a Fourier series model and a noisy Gaussian model for data in the over-parameterized regime. The latter also obtained this scenario under both linear and non-linear feature models for ridge regression and min-norm least squares using random matrix theory. Also, [1] studied the same setting for deep linear and shallow non-linear networks.

The analysis of the the generalization for max-margin linear classifiers in the high dimensional regime has been done in [99]. The exact expression for asymptotic prediction error is derived and in a specific case for two-layer neural network with random first-layer weights, the double descent curve was obtained. A similar double descent curve for logistic regression as well as linear discriminant analysis has been reported by [29]. Random feature learning in the same setting has also been studied for ridge regression in [93]. The authors have, in particular, shown that highly over-parametrized estimators with zero training error are statistically optimal at high signal-to-noise ratio (SNR). The asymptotic performance of regularized logistic regression in high dimensions is studied in [136] using the Convex Gaussian Min-max Theorem in the under-parametrized regime. The results in the current paper can consider all these models as special cases. Bounds on the generalization error of over-parametrized linear models are also given in [9, 107].

Although this paper and several other recent works consider only simple linear models and GLMs, much of the motivation is to understand generalization in deep neural networks where classical intuition may not hold [13, 107, 166]. In particular, a number of recent papers have shown the connection between neural networks in the over-parametrized regime and kernel methods. The works [26, 27] showed that gradient descent on over-parametrized neural networks learns a function in the RKHS corresponding to the random feature kernel. Training dynamics of overparametrized neural networks has been studied by [2, 4, 36, 69], and it is shown that the function learned is in an RKHS corresponding to the neural tangent kernel.

Approximate Message Passing. Our key tool to study the generalization error is approximate message passing (AMP), a class of inference algorithms originally developed in [10, 32, 34] for compressed sensing. We show that the learning problem for the GLM can be formulated as an inference problem on a certain multi-layer network. Multi-layer AMP methods [42, 61, 91, 114] can then be applied to perform the inference. The specific algorithm we use in this work is the multi-layer vector AMP (ML-VAMP) algorithm of [42, 114] which itself builds on several works [19, 45, 86, 111, 127]. The ML-VAMP algorithm is not necessarily the most computationally efficient procedure for the minimization (5.2). For our purposes, the key property is that ML-VAMP enables exact predictions of its performance in the large system limit. Specifically, the error of the algorithm estimates in each iteration can be predicted by a set of deterministic recursive equations called the *state evolution* or SE. The fixed points of these equations provide a way of computing the asymptotic performance of the algorithm. In certain cases, the algorithm can be proven to be Bayes optimal [7, 47, 131].

This approach of using AMP methods to characterize the generalization error of GLMs was also explored in [7] for i.i.d. distributions on the data. The explicit formulae for the asymptotic mean squared error for the regularized linear regression with rotationally invarient data matrices is proved in [50]. The ML-VAMP method in this work enables extensions to correlated features and to mismatch between training and test distributions.

5.2 Generalization Error: System Model

We consider the problem of estimating the weights $\mathbf{w}$ in the GLM model (5.1). As stated in the Introduction, we suppose we have training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$ arranged as $\mathbf{X} := [\mathbf{x}_1\, \mathbf{x}_2 \ldots \mathbf{x}_N]^\intercal \in \boldsymbol{R}^{N\times p}$, $\mathbf{y} := [y_1\, y_2 \ldots y_N]^\intercal \in \boldsymbol{R}^N$. Then we can write

$$\mathbf{y} = \phi_{\text{out}}(\mathbf{X}\mathbf{w}^0, \mathbf{d}), \tag{5.5}$$

where $\phi_{\text{out}}(\mathbf{z}, \mathbf{d})$ is the vector-valued function such that $[\phi_{\text{out}}(\mathbf{z}, \mathbf{d})]_n = \phi_{\text{out}}(z_n, d_n)$ and $\{d_n\}_{n=1}^N$ are general noise.

Given the training data $(\mathbf{X}, \mathbf{y})$, we consider estimates of $\mathbf{w}^0$ given by a regularized empirical risk minimization of the form (5.2). We assume that the loss function F_{out} and regularizer F_{in} are separable functions, i.e., one can write

$$F_{\text{out}}(\mathbf{y}, \mathbf{z}) = \sum_{n=1}^{N} f_{\text{out}}(y_n, z_n), \quad F_{\text{in}}(\mathbf{w}) = \sum_{j=1}^{p} f_{\text{in}}(w_j), \tag{5.6}$$

for some functions $f_{\text{out}} : \mathbb{R}^2 \to \mathbb{R}$ and $f_{\text{in}} : \mathbb{R} \to \mathbb{R}$. Many standard optimization problems in machine learning can be written in this form: logistic regression, support vector machines, linear regression, Poisson regression.

Large System Limit: We follow the LSL analysis of [10] commonly used for analyzing AMP-based methods. Specifically, we consider a sequence of problems indexed by the number of training samples N. For each N, we suppose that the number of features $p = p(N)$ grows linearly with N, i.e.,

$$\lim_{N\to\infty} \frac{p(N)}{N} \to \beta \tag{5.7}$$

for some constant $\beta \in (0, \infty)$. Note that $\beta > 1$ corresponds to the over-parameterized regime and $\beta < 1$ corresponds to the under-parameterized regime.

True parameter: We assume the *true* weight vector $\mathbf{w}^0$ has components whose empirical distribution converges as

$$\lim_{N\to\infty} \{w_n^0\} \overset{PL(2)}{=} W^0, \tag{5.8}$$

for some limiting random variable W^0. The precise definition of empirical convergence is given in Appendix 2.6.1. It means that the empirical distribution $\frac{1}{p}\sum_{i=1}^{p} \delta_{w_i}$ converges, in the Wasserstein-2 metric (see Chap. 6 [157]), to the distribution of the finite-variance random variable W^0. Importantly, the limit (5.8) will hold if the components $\{w_i^0\}_{i=1}^p$ are drawn i.i.d. from the distribution of W^0 with $\mathbb{E}(W^0)^2 < \infty$. However, as discussed in Appendix 2.6.1, the convergence can also be satisfied by correlated sequences and deterministic sequences.

Training data input: For each N, we assume that the training input data samples, $\mathbf{x}_i \in \boldsymbol{R}^p$, $i = 1, \ldots, N$, are i.i.d. and drawn from a p-dimensional Gaussian distribution with zero mean and covariance $\mathbf{\Sigma}_{\rm tr} \in \boldsymbol{R}^{p \times p}$. The covariance can capture the effect of features being correlated. We assume the covariance matrix has an eigenvalue decomposition,

$$\mathbf{\Sigma}_{\rm tr} = \tfrac{1}{p}\mathbf{V}_0^{\mathsf{T}} \mathrm{diag}(\mathbf{s}_{\rm tr}^2)\mathbf{V}_0, \tag{5.9}$$

where $\mathbf{s}_{\rm tr}^2$ are the eigenvalues of $\mathbf{\Sigma}_{\rm tr}$ and $\mathbf{V}_0 \in \boldsymbol{R}^{p \times p}$ is the orthogonal matrix of eigenvectors. The scaling $\frac{1}{p}$ ensures that the total variance of the samples, $\mathbb{E}\|\mathbf{x}_i\|^2$, does not grow with N. We will place a certain random model on $\mathbf{s}_{\rm tr}$ and $\mathbf{V}_0$ momentarily.

Using the covariance (5.9), we can write the data matrix as

$$\mathbf{X} = \mathbf{U}\, \mathrm{diag}(\mathbf{s}_{\rm tr})\mathbf{V}_0, \tag{5.10}$$

where $\mathbf{U} \in \boldsymbol{R}^{N \times p}$ has entries drawn i.i.d. from $\mathcal{N}(0, \frac{1}{p})$. For the purpose of analysis, it is useful to express the matrix $\mathbf{U}$ in terms of its SVD:

$$\mathbf{U} = \mathbf{V}_2 \mathbf{S}_{\rm mp} \mathbf{V}_1, \quad \mathbf{S}_{\rm mp} := \begin{bmatrix} \mathrm{diag}(\mathbf{s}_{\rm mp}) & \mathbf{0} \\ \mathbf{0} & * \end{bmatrix} \tag{5.11}$$

where $\mathbf{V}_1 \in \boldsymbol{R}^{N \times N}$ and $\mathbf{V}_2 \in \boldsymbol{R}^{p \times p}$ are orthogonal and $\mathbf{S}_{\rm mp} \in \mathbb{R}^{N \times p}$ with non-zero entries $\mathbf{s}_{\rm mp} \in \boldsymbol{R}^{\{}^{\min N, p\}}$ only along the principal diagonal. $\mathbf{s}_{\rm mp}$ are the singular values of $\mathbf{U}$. A standard result of random matrix theory is that, since $\mathbf{U}$ is i.i.d. Gaussian with entries $\mathcal{N}(0, \frac{1}{p})$, the matrices $\mathbf{V}_1$ and $\mathbf{V}_2$ are Haar-distributed on the group of orthogonal matrices and $\mathbf{s}_{\rm mp}$ is such that

$$\lim_{N \to \infty} \{s_{{\rm mp},i}\} \overset{PL(2)}{=} S_{\rm mp}, \tag{5.12}$$

where $S_{\rm mp} \geq 0$ is a non-negative random variable such that $S_{\rm mp}^2$ satisfies the Marcencko-Pastur distribution. Details on this distribution are in Appendix 2.9.

Training data output: Given the input data $\mathbf{X}$, we assume that the training outputs $\mathbf{y}$ are generated from (5.5), where the noise $\mathbf{d}$ is independent of $\mathbf{X}$ and has an empirical distribution which converges as

$$\lim_{N\to\infty} \{d_i\} \overset{PL(2)}{=} D. \tag{5.13}$$

Again, the limit (5.13) will be satisfied if $\{d_i\}_{i=1}^N$ are i.i.d. draws of random variable D with bounded second moments.

Test data: To measure the generalization error, we assume now that we are given a test point $\mathbf{x}_{\rm ts}$, and we obtain the true output $y_{\rm ts}$ and predicted output $\widehat{y}_{\rm ts}$ given by (5.3). We assume that the test data inputs are also Gaussian, i.e.,

$$\mathbf{x}_{\rm ts}^{\mathsf{T}} = \mathbf{u}^{\mathsf{T}} \mathrm{diag}(\mathbf{s}_{\rm ts}) \mathbf{V}_0, \tag{5.14}$$

where $\mathbf{u} \in \boldsymbol{R}^p$ has i.i.d. Gaussian components, $\mathcal{N}(0, \frac{1}{p})$, and $\mathbf{s}_{\rm ts}$ and $\mathbf{V}_0$ are the eigenvalues and eigenvectors of the test data covariance matrix. That is, the test data sample has a covariance matrix

$$\boldsymbol{\Sigma}_{\rm ts} = \tfrac{1}{p} \mathbf{V}_0^{\mathsf{T}} \mathrm{diag}(\mathbf{s}_{\rm ts}^2) \mathbf{V}_0. \tag{5.15}$$

In comparison to (5.9), we see that we are assuming that the eigenvectors of the training and test data are the same, but the eigenvalues may be different. In this way, we can capture distributional mismatch between the training and test data. For example, we will be able to measure the generalization error when the test sample is outside a subspace explored by the training data.

To capture the relation between the training and test distributions, we assume that components of $\mathbf{s}_{\rm tr}$ and $\mathbf{s}_{\rm ts}$ converge as

$$\lim_{N\to\infty} \{(s_{{\rm tr},i}, s_{{\rm ts},i})\} \overset{PL(2)}{=} (S_{\rm tr}, S_{\rm ts}), \tag{5.16}$$

to some non-negative, bounded random vector $(S_{\rm tr}, S_{\rm ts})$. The joint distribution on $(S_{\rm tr}, S_{\rm ts})$ captures the relation between the training and test data.

When $S_{\rm tr} = S_{\rm ts}$, our model corresponds to the case when the training and test distribution are matched. Isotropic Gaussian features in both training and test data correspond to covariance matrices $\boldsymbol{\Sigma}_{\rm tr} = \frac{1}{p}\sigma_{\rm tr}^2 \mathbf{I}$, $\boldsymbol{\Sigma}_{\rm ts} = \frac{1}{p}\sigma_{\rm ts}^2 \mathbf{I}$, which can be modeled as $S_{\rm tr} = \sigma_{\rm tr}$, $S_{\rm ts} = \sigma_{\rm ts}$.

We also require that the matrix $\mathbf{V}_0$ is uniformly distributed on the set of $p \times p$ orthogonal matrices.

Generalization error: From the training data, we obtain an estimate $\widehat{\mathbf{w}}$ via a regularized empirical risk minimization (5.2). Given a test sample $\mathbf{x}_{\rm ts}$ and parameter estimate $\widehat{\mathbf{w}}$, the true output $y_{\rm ts}$ and predicted output $\widehat{y}_{\rm tr}$ are given by equation (5.3). We assume the test noise is distributed as $d_{\rm ts} \sim D$, following the same distribution as the training data. The postulated inverse-link function $\phi(\cdot)$ in (5.3) may be different from the true inverse-link function $\phi_{\rm out}(\cdot)$.

The generalization error is defined as the asymptotic expected loss,

$$\mathcal{E}_{\rm ts} := \lim_{N\to\infty} \mathbb{E} f_{\rm ts}(\widehat{y}_{\rm ts}, y_{\rm ts}), \tag{5.17}$$

where $f_{\rm ts}(\cdot)$ is some loss function relevant for the test error (which may be different from the training loss). The expectation in (5.17) is with respect to the randomness in the training as well as test data, and the noise. Our main result provides a formula for the generalization error (5.17).

5.3 Learning GLMs via ML-VAMP

There are many methods for solving the minimization problem (5.2). We apply the ML-VAMP algorithm of [42, 112]. This algorithm is not necessarily the most computationally efficient method. For our purposes, however, the algorithm serves as a constructive proof technique, i.e., it enables exact predictions for generalization error in the LSL as described above. Moreover, in the case when loss function (5.2) is strictly convex, the problem has a unique global minimum, whereby the generalization error of this minimum is agnostic to the choice of algorithm used to find this minimum. To that end, we start by reformulating (5.2) in a form that is amicable to the application of ML-VAMP, Algorithm 5.

Multi-Layer Representation. The first step in applying ML-VAMP to the GLM learning problem is to represent the mapping from the true parameters $\mathbf{w}^0$ to the output $\mathbf{y}$ as a certain

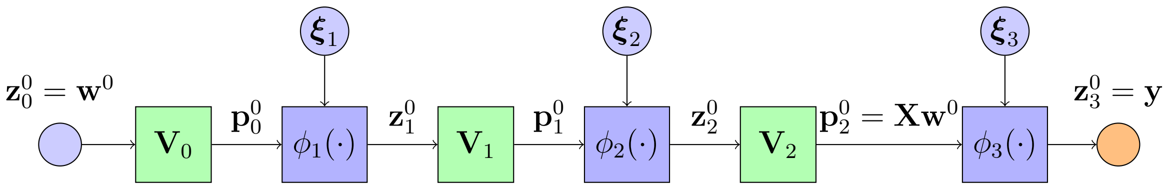


Figure 5.1: Sequence flow representing the mapping from the unknown parameter values $\mathbf{w}^0$ to the vector of responses $\mathbf{y}$ on the training data.

multi-layer network. We combine (5.5), (5.10) and (5.11), so that the mapping $\mathbf{w}^0 \mapsto \mathbf{y}$ can be written as the following sequence of operations (as illustrated in Fig. 5.1):

$$\begin{aligned} \mathbf{z}_0^0 &:= \mathbf{w}^0, & \mathbf{p}_0^0 &:= \mathbf{V}_0\mathbf{z}_0^0, \\ \mathbf{z}_1^0 &:= \phi_1(\mathbf{p}_0^0, \boldsymbol{\xi}_1), & \mathbf{p}_1^0 &:= \mathbf{V}_1\mathbf{z}_1^0, \\ \mathbf{z}_2^0 &:= \phi_2(\mathbf{p}_1^0, \boldsymbol{\xi}_2), & \mathbf{p}_2^0 &:= \mathbf{V}_2\mathbf{z}_2^0, \\ \mathbf{z}_3^0 &:= \phi_3(\mathbf{p}_2^0, \boldsymbol{\xi}_3) = \mathbf{y}, \end{aligned} \tag{5.18}$$

where $\boldsymbol{\xi}_\ell$ are the following vectors:

$$\boldsymbol{\xi}_1 := \mathbf{s}_{\text{tr}}, \quad \boldsymbol{\xi}_2 := \mathbf{s}_{\text{mp}}, \quad \boldsymbol{\xi}_3 := \mathbf{d}, \tag{5.19}$$

and the functions $\phi_\ell(\cdot)$ are given by

$$\begin{aligned} \phi_1(\mathbf{p}_0, \mathbf{s}_{\text{tr}}) &:= \text{diag}(\mathbf{s}_{\text{tr}})\mathbf{p}_0, \\ \phi_2(\mathbf{p}_1, \mathbf{s}_{\text{mp}}) &:= \mathbf{S}_{\text{mp}}\mathbf{p}_1, \\ \phi_3(\mathbf{p}_2, \mathbf{d}) &:= \phi_{\text{out}}(\mathbf{p}_2, \mathbf{d}). \end{aligned} \tag{5.20}$$

We see from Fig. 5.1 that the mapping of true parameters $\mathbf{w}^0 = \mathbf{z}_0^0$ to the observed response vector $\mathbf{y} = \mathbf{z}_3^0$ is described by a multi-layer network of alternating orthogonal operators $\mathbf{V}_\ell$ and non-linear functions $\phi_\ell(\cdot)$. Let $L = 3$ denote the number of layers in this multi-layer network.

The minimization (5.2) can also be represented using a similar signal flow graph. Given a parameter candidate $\mathbf{w}$, the mapping $\mathbf{w} \mapsto \mathbf{X}\mathbf{w}$ can be written using the sequence of vectors

$$\begin{aligned} &\mathbf{z}_0 := \mathbf{w}, && \mathbf{p}_0 := \mathbf{V}_0\mathbf{z}_0, \\ &\mathbf{z}_1 := \mathbf{S}_{\rm tr}\mathbf{p}_0, && \mathbf{p}_1 := \mathbf{V}_1\mathbf{z}_1, \\ &\mathbf{z}_2 := \mathbf{S}_{\rm mp}\mathbf{p}_1, && \mathbf{p}_2 := \mathbf{V}_2\mathbf{z}_2 = \mathbf{X}\mathbf{w}. \end{aligned} \tag{5.21}$$

There are $L = 3$ steps in this sequence, and we let

$$\mathbf{z} = \{\mathbf{z}_0, \mathbf{z}_1, \mathbf{z}_2\}, \quad \mathbf{p} = \{\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2\}$$

denote the sets of vectors across the steps. The minimization in (5.2) can then be written in the following equivalent form:

$$\begin{aligned} &\min_{\mathbf{z},\mathbf{p}} F_0(\mathbf{z}_0) + F_1(\mathbf{p}_0, \mathbf{z}_1) + F_2(\mathbf{p}_1, \mathbf{z}_1) + F_3(\mathbf{p}_2) \\ &\qquad \text{subject to} \quad \mathbf{p}_\ell = \mathbf{V}_\ell \mathbf{z}_\ell, \quad \ell = 0, 1, 2, \end{aligned} \tag{5.22}$$

where the *penalty functions* F_ℓ are defined as

$$\begin{aligned} F_0(\cdot) &= F_{\rm in}(\cdot), & F_1(\cdot,\cdot) &= \delta_{\{\mathbf{z}_1 = \mathbf{S}_{\rm tr}\mathbf{p}_0\}}(\cdot,\cdot), \\ F_2(\cdot,\cdot) &= \delta_{\{\mathbf{z}_2 = \mathbf{S}_{\rm mp}\mathbf{p}_1\}}(\cdot,\cdot), & F_3(\cdot) &= F_{\rm out}(\mathbf{y}, \cdot), \end{aligned} \tag{5.23}$$

where $\delta_{\mathcal{A}}(\cdot)$ is 0 on the set $\mathcal{A}$, and $+\infty$ on $\mathcal{A}^c$.

ML-VAMP for GLM Learning. Using this multi-layer representation, we can now apply the ML-VAMP algorithm from [42, 112] to solve the optimization (5.22). The steps are shown in Algorithm 5. These steps are a special case of the "MAP version" of ML-VAMP in [112], but with a slightly different set-up for the GLM problem. We will call these steps the ML-VAMP GLM Learning Algorithm.

The algorithm operates in a set of iterations indexed by k. In each iteration, a "forward pass" through the layers generates estimates $\widehat{\mathbf{z}}_{k\ell}$ for the hidden variables $\mathbf{z}_\ell^0$, while a "backward pass" generates estimates $\widehat{\mathbf{p}}_{k\ell}$ for the variables $\mathbf{p}_\ell^0$. In each step, the estimates $\widehat{\mathbf{z}}_{k\ell}$ and $\widehat{\mathbf{p}}_{k\ell}$ are produced by functions $\mathbf{g}_\ell^+(\cdot)$ and $\mathbf{g}_\ell^-(\cdot)$ called *estimators* or *denoisers*.

For the MAP version of ML-VAMP algorithm in [112], the denoisers are essentially

Algorithm 5 ML-VAMP GLM Learning Algorithm

```
1: Initialize $\gamma_{0\ell}^- > 0$, $\mathbf{r}_{0\ell}^- = 0$ for $\ell = 0, \dots, L-1$
2:
3: for $k = 0, 1, \dots$ do
4:     // Forward Pass
5:     for $\ell = 0, \dots, L-1$ do
6:         if $\ell = 0$ then
7:             $\widehat{\mathbf{z}}_{k0} = \mathbf{g}_0^+(\mathbf{r}_{k0}^-, \gamma_{k0}^-)$
8:         else
9:             $\widehat{\mathbf{z}}_{k\ell} = \mathbf{g}_\ell^+(\mathbf{r}_{k,\ell-1}^+, \mathbf{r}_{k\ell}^-, \gamma_{k,\ell-1}^+, \gamma_{k\ell}^-)$
10:        end if
11:        $\alpha_{k\ell}^+ = \left\langle \partial\widehat{\mathbf{z}}_{k\ell}/\partial\mathbf{r}_{k\ell}^- \right\rangle$
12:        $\mathbf{r}_{k\ell}^+ = \dfrac{\mathbf{V}_\ell(\widehat{\mathbf{z}}_{k\ell} - \alpha_{k\ell}^+\mathbf{r}_{k\ell}^-)}{1 - \alpha_{k\ell}^+}$
13:        $\gamma_{k\ell}^+ = (1/\alpha_{k\ell}^+ - 1)\gamma_{k\ell}^-$
14:    end for
15:
16:    // Backward Pass
17:    for $\ell = L, \dots, 1$ do
18:        if $\ell = L$ then
19:            $\widehat{\mathbf{p}}_{k,L-1} = \mathbf{g}_L^-(\mathbf{r}_{k,L-1}^+, \gamma_{k,L-1}^+)$
20:        else
21:            $\widehat{\mathbf{p}}_{k,\ell-1} = \mathbf{g}_\ell^-(\mathbf{r}_{k,\ell-1}^+, \mathbf{r}_{k+1,\ell}^-, \gamma_{k,\ell-1}^+, \gamma_{k+1,\ell}^-)$
22:        end if
23:        $\alpha_{k,\ell-1}^- = \left\langle \partial\widehat{\mathbf{p}}_{k,\ell-1}/\partial\mathbf{r}_{k,\ell-1}^+ \right\rangle$
24:        $\mathbf{r}_{k+1,\ell-1}^- = \dfrac{\mathbf{V}_{\ell-1}^\top(\widehat{\mathbf{p}}_{k,\ell-1} - \alpha_{k,\ell-1}^-\mathbf{r}_{k,\ell-1}^+)}{1 - \alpha_{k,\ell-1}^-}$
25:        $\gamma_{k+1,\ell-1}^- = (1/\alpha_{k,\ell-1}^- - 1)\gamma_{k,\ell-1}^+$
26:    end for
27: end for
```

proximal-type operators defined as

$$\text{prox}_{F/\gamma}(\boldsymbol{u}) := \underset{\mathbf{x}}{\text{argmin}}\; F(\mathbf{x}) + \tfrac{\gamma}{2}\left\|\mathbf{x} - \boldsymbol{u}\right\|^2. \tag{5.24}$$

An important property of the proximal operator is that for separable functions F of the form (5.6), we have $[\text{prox}_{F/\gamma}(\boldsymbol{u})]_i = \text{prox}_{f/\gamma}(\boldsymbol{u}_i)$.

In the case of the GLM model, for $\ell = 0$ and L, on lines 7 and 19, the denoisers are proximal operators given by

$$\mathbf{g}_0^+(\mathbf{r}_0^-, \gamma_0^-) = \text{prox}_{F_{\text{in}}/\gamma_0^-}(\mathbf{r}_0^-), \tag{5.25a}$$

$$\mathbf{g}_3^-(\mathbf{r}_2^+, \mathbf{y}, \gamma_2^+) = \text{prox}_{F_{\text{out}}/\gamma_2^+}(\mathbf{r}_2^+). \tag{5.25b}$$

Note that in (5.25b), there is a dependence on $\mathbf{y}$ through the term $F_{\text{out}}(\mathbf{y}, \cdot)$. For the *middle* terms, $\ell = 1, 2$, i.e., lines 9 and 21, the denoisers are given by

$$\mathbf{g}_\ell^+(\mathbf{r}_{\ell-1}^+, \mathbf{r}_\ell^-, \gamma_{\ell-1}^+, \gamma_\ell^-) := \widehat{\mathbf{z}}_\ell, \tag{5.26a}$$

$$\mathbf{g}_\ell^-(\mathbf{r}_{\ell-1}^+, \mathbf{r}_\ell^-, \gamma_{\ell-1}^+, \gamma_\ell^-) := \widehat{\mathbf{p}}_{\ell-1}, \tag{5.26b}$$

where $(\widehat{\mathbf{p}}_{\ell-1}, \widehat{\mathbf{z}}_\ell)$ are the solutions to the minimization

$$\begin{aligned}(\widehat{\mathbf{p}}_{\ell-1}, \widehat{\mathbf{z}}_\ell) := \underset{(\mathbf{p}_{\ell-1}, \mathbf{z}_\ell)}{\text{argmin}}\ & F_\ell(\mathbf{p}_{\ell-1}, \mathbf{z}_\ell) + \frac{\gamma_\ell^-}{2}\|\mathbf{z}_\ell - \mathbf{r}_\ell^-\|^2 \\ & + \frac{\gamma_{\ell-1}^+}{2}\|\mathbf{p}_{\ell-1} - \mathbf{r}_{\ell-1}^+\|^2.\end{aligned} \tag{5.27}$$

The quantity $\langle \partial \boldsymbol{v}/\partial \boldsymbol{u} \rangle$ on lines 11 and 23 denotes the empirical mean $\frac{1}{N}\sum_{n=1}^N \partial v_n/\partial u_n$.

Thus, the ML-VAMP algorithm in Algorithm 5 reduces the joint constrained minimization (5.22) over variables $(\mathbf{z}_0, \mathbf{z}_1, \mathbf{z}_2)$ and $(\mathbf{p}_0, \mathbf{p}_1, \mathbf{p}_2)$ to a set of proximal operations on pairs of variables $(\mathbf{p}_{\ell-1}, \mathbf{z}_\ell)$. As discussed in [112], this type of minimization is similar to ADMM with adaptive step-sizes. Details of the denoisers $\mathbf{g}_\ell^\pm$ and other aspects of the algorithm are given in Appendix C.1.

5.4 Main Result

We make two assumptions. The first assumption imposes certain regularity conditions on the functions f_{ts}, ϕ, ϕ_{out}, and maps $\mathbf{g}_\ell^\pm$ appearing in Algorithm 5. The precise definitions of pseudo-Lipschitz continuity and uniform Lipschitz continuity are given in Appendix 2.6.1 of the supplementary material.

Assumption 1. The denoisers and link functions satisfy the following continuity conditions:

(*a*) The proximal operators in (5.25),

$$\mathbf{g}_0^+(\mathbf{r}_0^-, \gamma_0^-), \quad \mathbf{g}_3^-(\mathbf{r}_2^+, \mathbf{y}, \gamma_2^+),$$

are uniformly Lipschitz continuous in $\mathbf{r}_0^-$ and $(\mathbf{r}_2^+, \mathbf{y})$ over parameters γ_0^- and γ_2^+.

(*b*) The link function $\phi_{\rm out}(p, d)$ is Lipschitz continuous in (p, d). The test error function $f_{\rm ts}(\phi(\widehat{z}), \phi_{\rm out}(z, d))$ is pseduo-Lipschitz continuous in $(\widehat{z}, z, d)$ of order 2.

Our second assumption is that the ML-VAMP algorithm converges. Specifically, let $\mathbf{x}_k = \mathbf{x}_k(N)$ be any set of outputs of Algorithm 5, at some iteration k and dimension N. For example, $\mathbf{x}_k(N)$ could be $\widehat{\mathbf{z}}_{k\ell}(N)$ or $\widehat{\mathbf{p}}_{k\ell}(N)$ for some ℓ, or a concatenation of signals such as $\begin{bmatrix}\mathbf{z}_\ell^0(N) & \widehat{\mathbf{z}}_{k\ell}(N)\end{bmatrix}$.

Assumption 2. Let $\mathbf{x}_k(N)$ be any finite set of outputs of the ML-VAMP algorithm as above. Then there exist limits

$$\mathbf{x}(N) = \lim_{k\to\infty} \mathbf{x}_k(N) \tag{5.28}$$

satisfying

$$\lim_{k\to\infty} \lim_{N\to\infty} \frac{1}{N}\|\mathbf{x}_k(N) - \mathbf{x}(N)\|^2 = 0. \tag{5.29}$$

We are now ready to state our main result.

Theorem 10. *Consider the GLM learning problem* (5.2) *solved by applying Algorithm 5 to the equivalent problem* (5.22) *under the assumptions of Section 5.2 along with Assumptions 1 and 2. Then, there exist constants* $\tau_0^-, \overline{\gamma}_0^+ > 0$ *and* $\mathbf{M} \in \mathbb{R}^{2\times 2}_{\succ 0}$ *such that the following hold:*

(a) The fixed points $\{\widehat{\mathbf{z}}_\ell, \widehat{\mathbf{p}}_\ell\}$*,* $\ell = 0, 1, 2$ *of Algorithm 5 satisfy the KKT conditions for the constrained optimization problem* (5.22)*. Equivalently* $\widehat{\mathbf{w}} := \widehat{\mathbf{z}}_0$ *is a stationary point of* (5.2)*.*

(b) The true parameter $\mathbf{w}^0$ *and its estimate* $\widehat{\mathbf{w}}$ *empirically converge as*

$$\lim_{N\to\infty} \{(w_i^0, \widehat{w}_i)\} \overset{PL(2)}{=} (W^0, \widehat{W}), \tag{5.30}$$

where W^0 is the random variable from (5.8) *and*

$$\widehat{W} = \mathrm{prox}_{f_{\mathrm{in}}/\overline{\gamma}_0^+}(W^0 + Q_0^-), \tag{5.31}$$

with $Q_0^- = \mathcal{N}(0, \tau_0^-)$ independent of W^0.

(c) *The asymptotic generalization error* (5.17) *with $(y_{\mathrm{ts}}, \widehat{y}_{\mathrm{ts}})$ defined as* (5.3) *is given by*

$$\mathcal{E}_{\mathrm{ts}} = \mathbb{E}\, f_{\mathrm{ts}}\Big(\phi_{\mathrm{out}}(Z_{\mathrm{ts}}, D), \phi(\widehat{Z}_{\mathrm{ts}})\Big), \tag{5.32}$$

where $(Z_{\mathrm{ts}}, \widehat{Z}_{\mathrm{ts}}) \sim \mathcal{N}(\mathbf{0}_2, \mathbf{M})$ and independent of D.

Part (a) shows that, similar to gradient descent, Algorithm 5 finds the stationary points of problem (5.2). These stationary points will be unique in strictly convex problems such as linear and logistic regression. Thus, in such cases, the same results will be true for any algorithm that finds such stationary points. Hence, the fact that we are using ML-VAMP is immaterial – our results apply to any solver for (5.2). Note that the convergence to the fixed points $\{\widehat{\mathbf{z}}_\ell, \widehat{\mathbf{p}}_\ell\}$ is assumed from Assumption 2.

Part (b) provides an exact description of the asymptotic statistical relation between the true parameter $\mathbf{w}^0$ and its estimate $\widehat{\mathbf{w}}$. The parameters $\tau_0^-, \overline{\gamma}_0^+ > 0$ and $\mathbf{M}$ can be explicitly computed using a set of recursive equations called the *state evolution* or SE described in Appendix C.2 in the supplementary material.

We can use the expressions to compute a variety of relevant metrics. For example, the $PL(2)$ convergence shows that the MSE on the parameter estimate is

$$\lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} (w_n^0 - \widehat{w}_n)^2 = \mathbb{E}(W^0 - \widehat{W})^2. \tag{5.33}$$

The expectation on the right hand side of (5.33) can then be computed via integration over the joint density of $(W^0, \widehat{W})$ from part (b). In this way, we have a simple and exact method to compute the parameter error. Other metrics such as parameter bias or variance, cosine angle or sparsity detection can also be computed.

Part (c) of Theorem 10 similarly exactly characterizes the asymptotic generalization error. In this case, we would compute the expectation over the three variables $(Z, \widehat{Z}, D)$. In this way, we have provided a methodology for exactly predicting the generalization error from the key parameters of the problems such as the sampling ratio $\beta = p/N$, the regularizer, the output function, and the distributions of the true weights and noise. We provide several examples such as linear regression, logistic regression and SVM in the Appendix C.6. We also recover the result by [55] in Appendix C.6.

Remarks on Assumptions. Note that Assumption 1 is satisfied in many practical cases. For example, it can be verified that it is satisfied in the case when $f_{\text{in}}(\cdot)$ and $f_{\text{out}}(\cdot)$ are convex. Assumption 2 is somewhat more restrictive in that it requires that the ML-VAMP algorithm converges. The convergence properties of ML-VAMP are discussed in [45]. The ML-VAMP algorithm may not always converge, and characterizing conditions under which convergence is possible is an open question. However, experiments in [127] show that the algorithm does indeed often converge, and in these cases, our analysis applies. In any case, we will see below that the predictions from Theorem 10 agree closely with numerical experiments in several relevant cases.

In some special cases equation (5.32) simplifies to yield quantitative insights for interesting modeling artifacts. We discuss these in Appendix C.6 in the supplementary material.

5.5 Experiments

Training and Test Distributions. We validate our theoretical results on a number of synthetic data experiments. For all the experiments, the training and test data is generated following the model in Section 5.2. We generate the training and test eigenvalues as i.i.d. with lognormal distributions,

$$S_{\text{tr}}^2 = A(10)^{0.1u_{\text{tr}}}, \quad S_{\text{ts}}^2 = A(10)^{0.1u_{\text{ts}}},$$

where $(u_{\rm tr}, u_{\rm ts})$ are bivariate zero-mean Gaussian with

$$\operatorname{cov}(u_{\rm tr}, u_{\rm ts}) = \sigma_u^2 \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}.$$

In the case when $\sigma_u^2 = 0$, we obtain eigenvalues that are equal, corresponding to the i.i.d. case. With $\sigma_u^2 > 0$ we can model correlated features. Also, when the correlation coefficient $\rho = 1$, $S_{\rm tr} = S_{\rm ts}$, so there is no training and test mismatch. However, we can also select $\rho < 1$ to experiment with cases when the training and test distributions differ. In the examples below, we consider the following three cases:

(1) i.i.d. features ($\sigma_u = 0$);

(2) correlated features with matching training and test distributions ($\sigma_u = 3$ dB, $\rho = 1$); and

(3) correlated features with train-test mismatch ($\sigma_u = 3$ dB, $\rho = 0.5$).

For all experiments below, the true model coefficients are generated as i.i.d. Gaussian $w_j^0 \sim \mathcal{N}(0,1)$ and we use standard L2-regularization, $f_{\rm in}(w) = \lambda w^2/2$ for some $\lambda > 0$. Our framework can incorporate arbitrary i.i.d. distributions on w_j and regularizers, but we will illustrate just the Gaussian case with L2-regularization here.

Under-regularized linear regression. We first consider the case of under-regularized linear regression where the output channel is $\phi_{\rm out}(p,d) = p + d$ with $d \sim \mathcal{N}(0, \sigma_d^2)$. The noise variance σ_d^2 is set for an SNR level of 10 dB. We use a standard mean-square error (MSE) output loss, $f_{\rm out}(y,p) = (y-p)^2/(2\sigma_d^2)$. Since we are using the L2-regularizer, $f_{\rm in}(w) = \lambda w^2/2$, the minimization (5.2) is standard ridge regression. Moreover, if we were to select $\lambda = 1/\mathbb{E}(w_j^0)^2$, then the ridge regression estimate would correspond to the minimum mean-squared error (MMSE) estimate of the coefficients $\mathbf{w}^0$. However, to study the under-regularized regime, we take $\lambda = (10)^{-4}/\mathbb{E}(w_j^0)^2$.

Fig. 5.2 plots the test MSE for the three cases described above for the linear model. In the figure, we take $p = 1000$ features and vary the number of samples n from $0.2p$

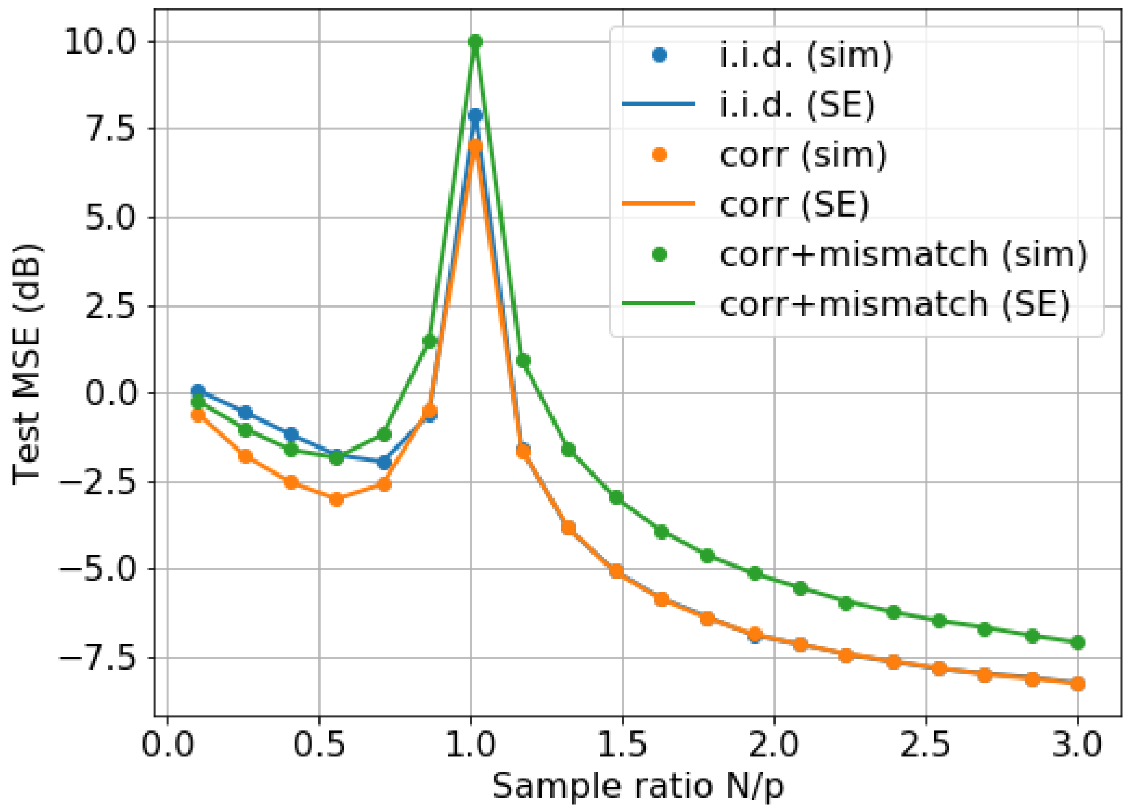


Figure 5.2: Test error for under-regularized linear regression under various train and test distributions

(over-parametrized) to $3p$ (under-paramertrized). For each value of n, we take 100 random instances of the model and compute the ridge regression estimate using the sklearn package and measure the test MSE on the 1000 independent test samples. The simulated values in Fig. 5.2 are the median test error over the 100 random trials. The test MSE is plotted in a normalized dB scale,

$$\text{Test MSE (dB)} = 10\log_{10}\left(\frac{\mathbb{E}(\widehat{y}_{\rm ts} - y_{\rm ts})^2}{\mathbb{E}y_{\rm ts}^2}\right).$$

Also plotted is the state evolution (SE) theoretical test MSE from Theorem 10.

In all three cases in Fig. 5.2, the SE theory exactly matches the simulated values for the test MSE. Note that the case of match training and test distributions for this problem was studied in [55, 93, 99] and we see the *double descent* phenomenon described in their work. Specifically, with highly under-regularized linear regression, the test MSE actually *increases* with more samples n in the over-parametrized regime ($n/p < 1$) and then decreases again in the under-parametrized regime ($n/p > 1$).

Our SE theory can also provide predictions for the correlated feature case. In this particular setting, we see that in the correlated case the test error is slightly lower in the over-parametrized regime since the energy of data is concentrated in a smaller sub-space. Interestingly, there is minimal difference between the correlated and i.i.d. cases for the under-parametrized regime when the training and test data match. When the training and test data are not matched, the test error increases. In all cases, the SE theory can accurately predict these effects.

Logistic Regression. Fig. 5.3 shows a similar plot as Fig. 5.2 for a logistic model. Specifically, we use a logistic output $P(y = 1) = 1/(1 + e^{-p})$, a binary cross entropy output loss $f_{\text{out}}(y, p)$, and ℓ_2-regularization level λ so that the output corresponds to the MAP estimate (we do not perform ridgeless regression in this case). The mean of the training and test eigenvalues $\mathbb{E}S_{\text{tr}}^2 = \mathbb{E}S_{\text{ts}}^2$ are selected such that, if the true coefficients $\mathbf{w}^0$ were known, we could obtain a 5% prediction error. As in the linear case, we generate random instances of the model, use the sklearn package to perform the logistic regression, and evaluate the estimates on 1000 new test samples. We compute the median error rate (1− accuracy) and compare the simulated values with the SE theoretical estimates. The i.i.d. case was considered in [136]. Fig. 5.3 shows that our SE theory is able to predict the test error rate exactly in i.i.d. cases along with a correlated case and a case with training and test mismatch.

Nonlinear Regression. The SE framework can also consider non-convex problems. As an example, we consider a non-linear regression problem where the output function is

$$\phi_{\text{out}}(p, d) = \tanh(p) + d, \quad d \sim \mathcal{N}(0, \sigma_d^2). \tag{5.34}$$

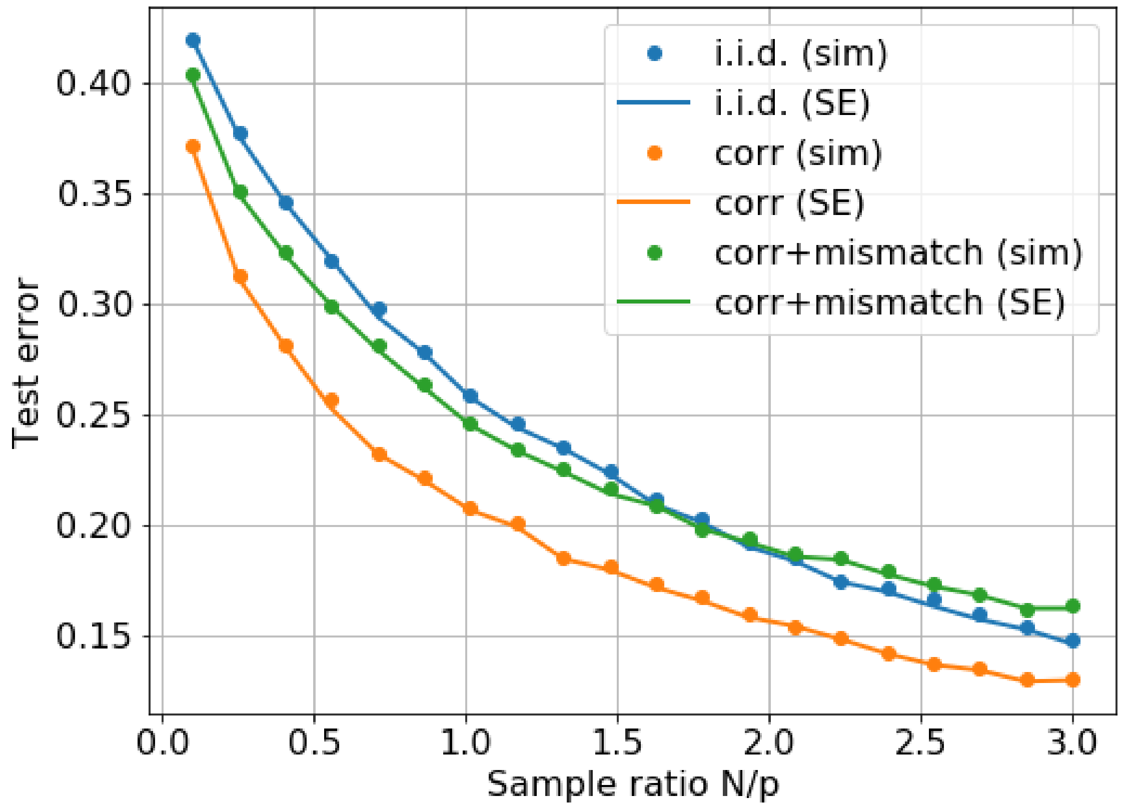


Figure 5.3: Classification error rate with logistic regression under various train and test distributions

The $\tanh(p)$ models saturation in the output. Corresponding to this output, we use a non-linear MSE output loss

$$f_{\text{out}}(y - p) = \frac{1}{2\sigma_d^2}(y - \tanh(p))^2. \tag{5.35}$$

This output loss is non-convex. We scale the data matrix so that the input $\mathbb{E}(p^2) = 9$ so that the $\tanh(p)$ is driven well into the non-linear regime. We also take $\sigma_d^2 = 0.01$.

For the simulation, the non-convex loss is minimized using Tensorflow where the non-linear model is described as a two-layer model. We use the ADAM optimizer [76] with 200 epochs to approach a local minimum of the objective (5.2). Fig. 5.4 plots the median test MSE for the estimate along with the SE theoretical test MSE. We again see that the SE theory is

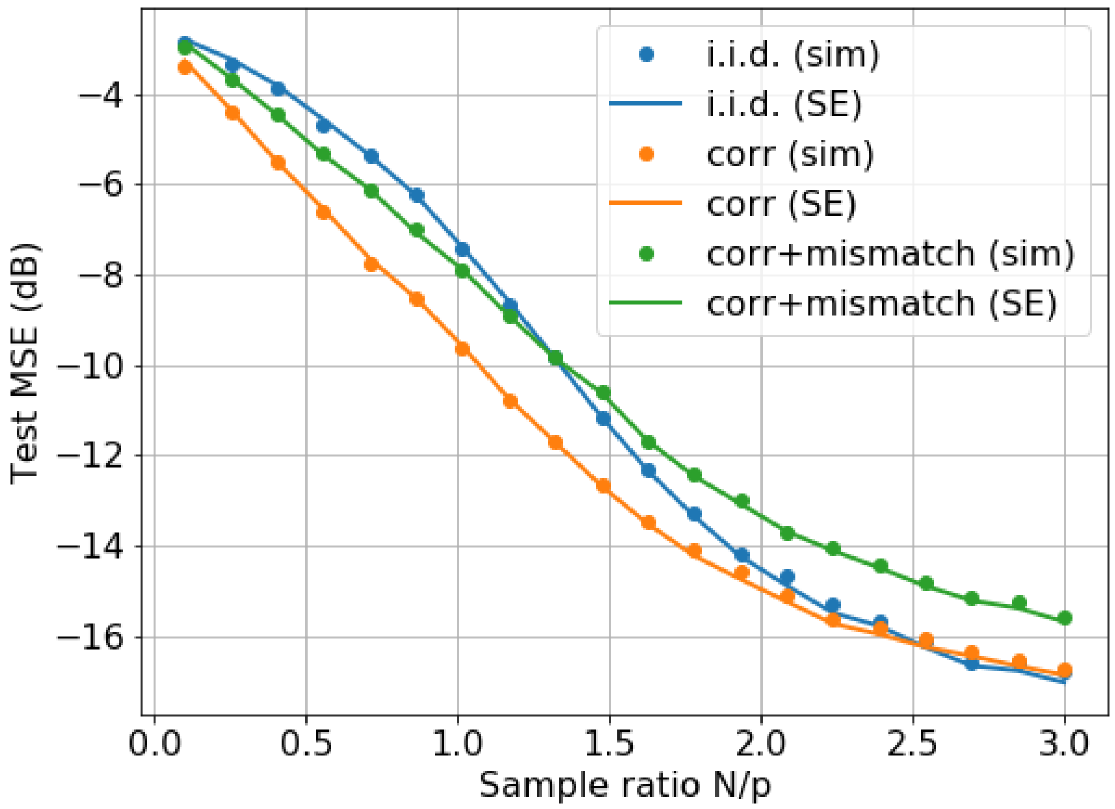


Figure 5.4: Test MSE under a non-linear least square estimation.

able to predict the test MSE in all cases even for this non-convex problem.

5.6 Discussion

In this paper we provide a procedure for exactly computing the asymptotic generalization error of a solution in a generalized linear model (GLM). This procedure is based on scalar quantities which are fixed points of a recursive iteration. The formula holds for a large class of generalization metrics, loss functions, and regularization schemes. Our formula allows analysis of important modeling effects such as

(i) overparameterization,

(ii) dependence between covariates, and

(iii) mismatch between train and test distributions,

which play a significant role in the analysis and design of machine learning systems. We experimentally validate our theoretical results for linear as well as non-linear regression and logistic regression, where a strong agreement is seen between our formula and simulated results.

Chapter 6

Conclusion

In this dissertation we have developed a general framework for analyzing multi layer inverse problems. Such problem are ubiquitous in signal processing and machine learning. A new set of algorithms based on Vector Approximate Message Passing are proposed which are equipped to solve the multi layer versions of linear inverse problems. We name these algorithms Multi-Layer Vector Approximate Message Passing (ML-VAMP). Several important properties of these algorithms have been explored in detail, with rigorous proofs for the mathematical claims.

These algorithms can be configured to solve the optimization based MAP estimation as well as the approximate Bayes estimation. We also present a matrix version of ML-VAMP called ML-Mat-VAMP which can solve generalizations of the multi-layer inverse problem involving matrix-valued unknown variables.

The fixed points of these iterative algorithms have been characterized even for non-convex problems, which ensures that they are accurate at convergence. Several important connections are drawn to contemporary literature on iterative optimization methods such as the Peaceman-Rachford Splitting ADMM algorithm. This interpretation of the algorithm provides a new line of attack for the exact analysis of proximal algorithms for non-convex problems.

Just like Vector Approximate Message Passing, these algorithms also possess a large system limit analysis in the proportional asymptotic regime. In this high dimensional space, the iterations of our iterative algorithms can be exactly tracked via an equivalent scalar model: the so-called State Evolution.

This enables understanding the statistical properties of the solutions to multi layer inverse problems. Moreover, this analysis is exact and has been verified to empirically hold for large random instances of the problems.

Our numerical experiments further illustrate that these algorithms are not just good for analysis but are also computationally efficient and can provide solutions which are perceptually comparable to off-the-shelf optimization solvers such as Adam as well as MCMC based solvers such as MALA.

To demonstrate the analytical power of the ML-VAMP framework, we have applied the ML-VAMP algorithm to solve some important practical problems in machine learning. This class of problems include learning generalized linear models, 1-layer and 2-layer neural networks. The key idea here is to pose these learning problems as equivalent multi-layer inverse problems. The resulting analysis allows machine learning researchers to understand the effects of several key training and modelling hyperparameters such as

1. Degree of overparameterization,
2. Covariance of features,
3. Mismatch between training and test distributions,
4. Choice of loss function,
5. Choice of regularization function, and
6. Choice of regularization strength.

Another noteworthy feature of this analysis is that it does not rely on convex analysis and hence is suitable for understanding non-convex optimization and its impact on learning and generalization.

Our framework thus provides a new set of tools to rigorously understand the behaviour of elementary models, their training procedures, and the several intermediate design choices, in contemporary machine learning and signal processing applications.

Appendices

Appendix A

Proofs from Chapter 3

Organization Appendix A.1 describes the definitions needed to define the State Evolution and introduces the framework for LSL analysis introduced by [10]. Appendix A.2 provides the State Evolution equations for the ML-VAMP algorithm. Appendix A.3 provides proofs for the fixed point results – Theorems 5 and 6. Appendix A.4 provides proofs for the large system limit of the ML-VAMP algorithm as stated in Theorems 9 and 8; these results are the consequence of a more general result – Theorem 12 – stated in Appendix B.4 and proved in Appendix B.5.

A.1 Empirical Convergence of Vector Sequences

We follow the framework of Bayati and Montanari [10], which models various sequences as deterministic, but with components converging empirically to a distribution. We start with a brief review of useful definitions. Let $\mathbf{x}(N) = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$ be a block vector with components $\mathbf{x}_n \in \boldsymbol{R}^r$ for some r. Thus, the vector $\mathbf{x}(N)$ is a vector with dimension rN. Given any function $g : \boldsymbol{R}^r \to \boldsymbol{R}^s$, we define the *componentwise extension* of $g(\cdot)$ as the function,

$$\mathbf{g}(\mathbf{x}) := (g(\mathbf{x}_1), \ldots, g(\mathbf{x}_N)) \in \boldsymbol{R}^{Ns}. \tag{A.1}$$

That is, $\mathbf{g}(\cdot)$ applies the function $g(\cdot)$ on each r-dimensional component. Similarly, we say $\mathbf{g}(\mathbf{x})$ *acts componentwise* on $\mathbf{x}$ whenever it is of the form (A.1) for some function $g(\cdot)$.

Next consider a sequence of block vectors of growing dimension,

$$\mathbf{x}(N) = (\mathbf{x}_1(N), \ldots, \mathbf{x}_N(N)), \qquad N = 1, 2, \ldots,$$

where each component $\mathbf{x}_n(N) \in \boldsymbol{R}^r$. In this case, we will say that $\mathbf{x}(N)$ is a *block vector sequence that scales with N under blocks $\mathbf{x}_n(N) \in \boldsymbol{R}^r$.* When $r = 1$, so that the blocks are scalar, we will simply say that $\mathbf{x}(N)$ is a *vector sequence that scales with N.* Such vector sequences can be deterministic or random. In most cases, we will omit the notational dependence on N and simply write $\mathbf{x}$.

Now, given $p \geq 1$, a function $f : \boldsymbol{R}^r \to \boldsymbol{R}^s$ is called *pseudo-Lipschitz continuous of order p*, if there exists a constant $C > 0$ such that for all $\mathbf{x}_1, \mathbf{x}_2 \in \boldsymbol{R}^r$,

$$\|f(\mathbf{x}_1) - f(\mathbf{x}_2)\| \leq C\|\mathbf{x}_1 - \mathbf{x}_2\| \left[1 + \|\mathbf{x}_1\|^{p-1} + \|\mathbf{x}_2\|^{p-1}\right].$$

Observe that in the case $p = 1$, pseudo-Lipschitz continuity reduces to usual Lipschitz continuity. Given $p \geq 1$, we will say that the block vector sequence $\mathbf{x} = \mathbf{x}(N)$ *converges empirically with p-th order moments* if there exists a random variable $X \in \boldsymbol{R}^r$ such that

(i) $\mathbb{E}\|X\|_p^p < \infty$; and

(ii) for any $f : \boldsymbol{R}^r \to \boldsymbol{R}$ that is pseudo-Lipschitz continuous of order p,

$$\lim_{N\to\infty} \tfrac{1}{N} \sum_{n=1}^{N} f(\mathbf{x}_n(N)) = \mathbb{E}\left[f(X)\right]. \tag{A.2}$$

In (A.2), we have the empirical mean of the components $f(\mathbf{x}_n(N))$ of the componentwise extension $\mathbf{f}(\mathbf{x}(N))$ converging to the expectation $\mathbb{E}[f(X)]$. In this case, with some abuse of notation, we will write

$$\lim_{N\to\infty} \{\mathbf{x}_n\} \overset{PL(p)}{=} X, \tag{A.3}$$

where, as usual, we have omitted the dependence on N in $\mathbf{x}_n(N)$. Importantly, empirical convergence can be defined on deterministic vector sequences, with no need for a probability space. If $\mathbf{x} = \mathbf{x}(N)$ is a random vector sequence, we will often require that the limit (A.3) holds almost surely.

Finally, we introduce the concept of *uniform pseduo-Lipschitz continuity.* Let $\phi(\mathbf{r}, \gamma)$ be a function on $\mathbf{r} \in \boldsymbol{R}^r$ and $\theta \in \boldsymbol{R}^s$. We say that $\phi(\mathbf{r}, \theta)$ is *uniformly Lipschitz continuous* in $\mathbf{r}$ at $\theta = \overline{\theta}$ if there exists constants $L_1, L_2 \geq 0$ and an open neighborhood U of $\overline{\theta}$ such that

$$\|\phi(\mathbf{r}_1, \theta) - \phi(\mathbf{r}_2, \theta)\| \leq L_1 \|\mathbf{r}_1 - \mathbf{r}_2\|, \qquad \forall \mathbf{r}_1, \mathbf{r}_2 \in \boldsymbol{R}^r, \theta \in U \tag{A.4a}$$

$$\|\phi(\mathbf{r}, \theta_1) - \phi(\mathbf{r}, \theta_2)\| \leq L_2 (1 + \|\mathbf{r}\|) \|\theta_1 - \theta_2\|, \qquad \forall \mathbf{r} \in \boldsymbol{R}^r, \theta_1, \theta_2 \in U. \tag{A.4b}$$

A.2 ML-VAMP State Evolution Equations

The state evolution (SE) recursively defines a set of scalar random variables that describe the typical components of the vector quantities produced from the ML-VAMP algorithm. The definition of the random variables are given in Algorithm 6. The algorithm steps mimic those in the ML-VAMP algorithm, Algorithm 3, but with each update producing scalar random variables instead of vectors. The updates use several functions:

$$f_0^0(w_0) = w_0, \qquad f_\ell^0(p_{\ell-1}^0, w_\ell) := f_\ell^0(p_{\ell-1}^0, \xi_\ell) := \phi_\ell(p_{\ell-1}^0, \xi_\ell), \quad \ell = 2, 4, \ldots, L, \tag{A.5a}$$

$$f_\ell^0(p_{\ell-1}^0, w_\ell) := f_\ell^0(p_{\ell-1}^0, (\bar{s}_\ell, \bar{b}_\ell, \bar{\xi}_\ell)) = \bar{s}_\ell p_{\ell-1}^0 + \bar{b}_\ell + \bar{\xi}_\ell, \quad \ell = 1, 3, \ldots, L{-}1, \tag{A.5b}$$

$$h_\ell^\pm(p_{\ell-1}^0, p_{\ell-1}^+, q_\ell^-, w_\ell, \theta_{k\ell}^\pm) = g_\ell^\pm(q_\ell^- + q_\ell^0, p_{\ell-1}^+ + p_{\ell-1}^0, \theta_{k\ell}^\pm), \quad \ell = 2, 4, \ldots L-2, \tag{A.5c}$$

$$h_\ell^\pm(p_{\ell-1}^0, p_{\ell-1}^+, q_\ell^-, w_\ell, \theta_{k\ell}^\pm) = G_\ell^\pm(q_\ell^- + q_\ell^0, p_{\ell-1}^+ + p_{\ell-1}^0, \theta_{k\ell}^\pm), \quad \ell = 1, 3, \ldots L-1, \tag{A.5d}$$

$$h_0^+(q_0^-, w_0\theta_{k0}^+) = g_0^+(q_0^- + w_0, \theta_{k0}^+), \quad h_L^-(p_{L-1}^0, p_{L-1}^+, w_L, \theta_{kL}^-) = g_L^-(p_{L-1}^+ + p_{L-1}^0, \theta_{kL}^-), \tag{A.5e}$$

$$f_0^+(q_0^-, w_0, \Lambda_{k0}^+) := \tfrac{1}{1-\alpha_{k\ell}^+}\left[h_0^+(q_0^-, w_0, \theta_{k0}^+) - w_0 - \alpha_{k0}^+ q_0^-\right], \tag{A.5f}$$

$$f_\ell^+(p_{\ell-1}^0, p_{\ell-1}^+, q_\ell^-, w_\ell, \Lambda_{k\ell}^+) := \tfrac{1}{1-\alpha_{k\ell}^+}\left[h_\ell^+(p_{\ell-1}^0, p_{\ell-1}^+, q_\ell^-, w_\ell, \theta_{k\ell}^+) - q_\ell^0 - \alpha_{k\ell}^+ q_\ell^-\right], \tag{A.5g}$$

$$f_L^-(p_{L-1}^0, p_{L-1}^+, w_L, \Lambda_{kL}^-) := \tfrac{1}{1-\alpha_{k\ell}^-}\left[h_L^-(p_{L-1}^0, p_{L-1}^+, w_L, \theta_{kL}^-) - p_{L-1}^0 - \alpha_{k,L-1}^- p_{L-1}^+\right], \tag{A.5h}$$

$$f_\ell^-(p_{\ell-1}^0, p_{\ell-1}^+, q_\ell^-, w_\ell, \Lambda_{k\ell}^-) := \tfrac{1}{1-\alpha_{k,\ell-1}^-}\left[h_\ell^-(p_{\ell-1}^0, p_{\ell-1}^+, q_\ell^-, w_\ell, \theta_{k\ell}^-) - p_{\ell-1}^0 - \alpha_{k,\ell-1}^- p_{\ell-1}^+\right]. \tag{A.5i}$$

In addition define the *perturbation* random variables W_ℓ (recall from (3.29)) as

$$W_0 = Z_0^0, \qquad W_\ell = \Xi_\ell, \qquad \ell = 2, 4, \ldots, L-2, \tag{A.6a}$$

$$W_\ell = (S_\ell, \overline{B}_\ell, \overline{\Xi}_\ell), \qquad \ell = 1, 3, \ldots, L-1. \tag{A.6b}$$

A.3 Proofs of ML-VAMP Fixed-Point Theorems

A.3.1 Proof of Theorem 5

The linear equalities in defining $\mathbf{s}_{k\ell}^{\pm}$ can be rewritten as,

$$\mathbf{r}_{k\ell}^{+} = \widehat{\mathbf{z}}_{k\ell}^{+} + \frac{1}{\alpha_{k\ell}^{-}}\mathbf{s}_{k\ell}^{+}, \qquad \mathbf{r}_{k+1,\ell}^{-} = \widehat{\mathbf{z}}_{k\ell}^{-} - \frac{1}{\alpha_{k\ell}^{+}}\mathbf{s}_{k+1,\ell}^{-} \tag{A.7}$$

Substituting (A.7) in lines 12 and 24 of Algorithm 3 give the updates (3.20b) and (3.21b) in Theorem 5. It remains to show that the optimization problem in updates (3.20a) and (3.21a) is equivalent to (3.12). It suffices to show that the terms dependent on $(\boldsymbol{z}_{\ell-1}^{-}, \boldsymbol{z}_{\ell}^{+})$ in b_ℓ from (3.12), and $\mathcal{L}_\ell$ from (3.20a) and (3.21a) are identical. This follows immediately on substituting (A.7) in (4.12). Thus there exists a bijective mapping between the fixed points $\{\widehat{\boldsymbol{z}}, \boldsymbol{r}^{+}, \boldsymbol{r}^{-}\}$ (of Algorithm 3) and $\{\widehat{\boldsymbol{z}}, \boldsymbol{s}\}$ (of Theorem 5).

It now remains to be shown that any fixed point of Algorithm 3 is a critical point of the augmented Lagrangian in (3.19). To that end, we need to show that there exists dual parameters $\mathbf{s}_\ell$ such that for all $\ell = 0, \ldots, L-1$,

$$\widehat{\mathbf{z}}_{\ell}^{+} = \widehat{\mathbf{z}}_{\ell}^{-}, \qquad \partial_{\mathbf{z}^{+}}\mathcal{L}(\widehat{\mathbf{z}}^{+}, \widehat{\mathbf{z}}^{-}, \mathbf{s}) \ni \mathbf{0}, \quad \partial_{\mathbf{z}^{-}}\mathcal{L}(\widehat{\mathbf{z}}^{+}, \widehat{\mathbf{z}}^{-}, \mathbf{s}) \ni \mathbf{0}, \tag{A.8}$$

where $\mathcal{L}(\cdot)$ is the Lagrangian in (3.19). Primal feasibility or $\widehat{\mathbf{z}}_{\ell}^{+} = \widehat{\mathbf{z}}_{\ell}^{-}$ was already shown in (3.17). As a consequence of the primal feasibility $\widehat{\mathbf{z}}_{\ell}^{+} = \widehat{\mathbf{z}}_{\ell}^{-}$, observe that

$$\mathbf{s}_{\ell}^{+} - \mathbf{s}_{\ell}^{-} = (\alpha_{\ell}^{+} + \alpha_{\ell}^{-})\widehat{\mathbf{z}}_{\ell} - \alpha_{\ell}^{+}\mathbf{r}_{\ell}^{-} - \alpha_{\ell}^{-}\mathbf{r}_{\ell}^{+} = 0, \tag{A.9}$$

where we have used (3.16). Define $\mathbf{s} := \mathbf{s}^{+} = \mathbf{s}^{-}$. To show the stationarity in (A.8) it suffices

Algorithm 6 State Evolution for ML-VAMP

Require: $f^0_\ell(\cdot)$, $f^\pm_\ell(\cdot)$ and $h^\pm_\ell(\cdot)$ from eqn. (A.5) and initial random variables: Z^0_0, $\{W_\ell, Q^-_{0\ell}\}$ from Section 3.4 and (A.6)

```
 1: // Initial pass
 2: $Q^0_0 = Z^0_0$, $\tau^0_0 = \mathbb{E}(Q^0_0)^2$ and $P^0_0 \sim \mathcal{N}(0,\tau^0_0)$
 3: for $\ell = 1,\ldots,L-1$ do
 4:     $Q^0_\ell = f^0_\ell(P^0_{\ell-1}, W_\ell)$
 5:     $P^0_\ell \sim \mathcal{N}(0,\tau^0_\ell)$, $\tau^0_\ell = \mathbb{E}(Q^0_\ell)^2$
 6: end for
 7:
 8: for $k = 0,1,\ldots$ do
 9:     // Forward Pass
10:     $\widehat{Q}^+_{k0} = h^+_0(Q^-_{k0}, W_0, \overline{\theta}^+_{k0}))$
11:     $\overline{\alpha}^+_{k0} = \mathbb{E}(\frac{\partial h^+_0}{\partial Q^-_{k0}}(Q^-_{k0}, W_0, \overline{\theta}^+_{k0}))$,        $\overline{\Lambda}^+_{k0} = (\overline{\alpha}^+_{k0}, \overline{\theta}^+_{k0})$
12:     $Q^+_{k0} = f^+_0(Q^-_0, W_0, \overline{\Lambda}^+_{k0})$
13:     $(P^0_0, P^+_{k0}) \sim \mathcal{N}(\mathbf{0}, \mathbf{K}^+_{k0})$,        $\mathbf{K}^+_{k0} := \mathrm{Cov}(Q^0_0, Q^+_{k0})$
14:     for $\ell = 1,\ldots,L-1$ do
15:         $\widehat{Q}^+_{k\ell} = h^+_\ell(P^0_{\ell-1}, P^+_{k,\ell-1}, Q^-_{k\ell}, W_\ell, \overline{\theta}^+_{k\ell}))$
16:         $\overline{\alpha}^+_{k\ell} = \mathbb{E}(\frac{\partial h^+_\ell}{\partial Q^-_{k\ell}}(P^0_{\ell-1}, P^+_{k,\ell-1}, Q^-_{k\ell}, W_\ell, \overline{\theta}^+_{k\ell}))$,        $\overline{\Lambda}^+_{k\ell} = (\overline{\alpha}^+_{k\ell}, \overline{\theta}^+_{k\ell})$
17:         $Q^+_{k\ell} = f^+_\ell(P^0_{\ell-1}, P^+_{k,\ell-1}, Q^-_{k\ell}, W_\ell, \overline{\Lambda}^+_{k\ell})$
18:         $(P^0_\ell, P^+_{k\ell}) \sim \mathcal{N}(\mathbf{0}, \mathbf{K}^+_{k\ell})$,        $\mathbf{K}^+_{k\ell} := \mathrm{Cov}(Q^0_\ell, Q^+_{k\ell})$
19:     end for

20:     // Backward Pass
21:     $\widehat{P}^-_{k+1,L-1} = h^-_L(P^0_{L-1}, P^+_{k,L-1}, W_L, \overline{\theta}^-_{k+1,L})$
22:     $\overline{\alpha}^-_{k+1,L} = \mathbb{E}(\frac{\partial h^-_L}{\partial P^+_{k,L-1}}(P^0_{L-1}, P^+_{k,L-1}, W_L, \overline{\theta}^-_{k+1,L}))$,        $\overline{\Lambda}^-_{k+1,L} = (\overline{\alpha}^-_{k+1,L}, \overline{\theta}^-_{k+1,L})$
23:     $P^-_{k+1,L-1} = f^-_L(P^0_{L-1}, P^+_{k,L-1}, W_L, \overline{\Lambda}^-_{k+1,L})$
24:     $Q^-_{k+1,L-1} \sim \mathcal{N}(0, \tau^-_{k+1,L-1})$,        $\tau^-_{k+1,L-1} := \mathbb{E}(P^-_{k+1,L-1})^2$
25:     for $\ell = L-1,\ldots,1$ do
26:         $\widehat{P}^-_{k+1,\ell-1} = h^-_\ell(P^0_{\ell-1}, P^+_{k,\ell-1}, W_\ell, \overline{\theta}^-_{k+1,\ell})$
27:         $\overline{\alpha}^-_{k+1,\ell} = \mathbb{E}(\frac{\partial h^-_\ell}{\partial P^+_{k,L-1}}(P^0_{\ell-1}, P^+_{k,\ell-1}, W_\ell, \overline{\theta}^-_{k+1,\ell}))$,        $\overline{\Lambda}^-_{k+1,\ell} = (\overline{\alpha}^-_{k+1,\ell}, \overline{\theta}^-_{k+1,\ell})$
28:         $P^-_{k+1,\ell-1} = f^-_\ell(P^0_{\ell-1}, P^+_{k,\ell-1}, Q^-_{k+1,\ell}, W_\ell, \overline{\Lambda}^-_{k\ell})$
29:         $Q^-_{k+1,\ell-1} \sim \mathcal{N}(0, \tau^-_{k+1,\ell-1})$,        $\tau^-_{k+1,\ell-1} := \mathbb{E}(P^-_{k+1,\ell-1})^2$
30:     end for
31: end for
```

to show that $\mathbf{s}_\ell$ is a valid dual parameter for which the following stationarity conditions hold,

$$\partial_{\mathbf{z}^-_{\ell-1}} \mathcal{L}_\ell(\widehat{\mathbf{z}}^-_{\ell-1}, \widehat{\mathbf{z}}^+_\ell; \widehat{\mathbf{z}}^+_{\ell-1}, \widehat{\mathbf{z}}^-_\ell, \mathbf{s}_{\ell-1}, \mathbf{s}_\ell) \ni \mathbf{0}, \qquad \partial_{\mathbf{z}^+_\ell} \mathcal{L}_\ell(\widehat{\mathbf{z}}^-_{\ell-1}, \widehat{\mathbf{z}}^+_\ell; \widehat{\mathbf{z}}^+_{\ell-1}, \widehat{\mathbf{z}}^-_\ell, \mathbf{s}_{\ell-1}, \mathbf{s}_\ell) \ni \mathbf{0}, \quad \text{(A.10)}$$

Indeed the above conditions are the stationarity conditions of the optimization problem in (3.20a) and (3.21a). Hence (A.8) holds.

A.3.2 Proof of Theorem 6

Observe that the Lagrangian function for the constrained optimization problem (3.27) for this specific choice of Lagrange multipliers is given by

$$\begin{aligned}
\mathcal{L}(\{b_\ell\}, \{q_\ell\}, \{\boldsymbol{r}^+_\ell\}, \{\boldsymbol{r}^-_\ell\}, \{\gamma^+_\ell\}, \{\gamma^-_\ell\}) &= \sum_{\ell=0}^{L} D_{\mathsf{KL}}(f_\ell(\boldsymbol{z}_\ell, \boldsymbol{z}_{\ell-1}) || p(\boldsymbol{z}_\ell|\boldsymbol{z}_{\ell-1})) + \sum_{\ell=0}^{L-1} H(q_\ell) \\
&+ \sum_{\ell=0}^{L-1} \gamma^-_\ell \boldsymbol{r}^{-\top}_\ell (\mathbb{E}[\boldsymbol{z}_\ell|f_\ell] - \mathbb{E}[\boldsymbol{z}_\ell|q_\ell]) + \gamma^+_\ell \boldsymbol{r}^{+\top}_\ell (\mathbb{E}[\boldsymbol{z}_\ell|f_{\ell+1}] - \mathbb{E}[\boldsymbol{z}_{\ell-1}|q_\ell]) \\
&+ \sum_{\ell=0}^{L-1} \tfrac{\gamma^-_\ell}{2}(\mathbb{E}[\|\boldsymbol{z}_\ell\|^2 |f_\ell] - \mathbb{E}[\|\boldsymbol{z}_\ell\|^2 |q_\ell]) + \tfrac{\gamma^+_\ell}{2}(\mathbb{E}[\|\boldsymbol{z}_\ell\|^2 |f_{\ell+1}] - \mathbb{E}[\|\boldsymbol{z}_\ell\|^2 |q_\ell])
\end{aligned}$$

Notice that the stationarity KKT conditions $\nabla_{f_\ell}\mathcal{L} = \mathbf{0}$ and $\nabla_{q_\ell}\mathcal{L} = \mathbf{0}$ give us the relation

$$f^*_\ell(\boldsymbol{z}_\ell, \boldsymbol{z}_{\ell-1}) \propto p(\boldsymbol{z}_\ell|\boldsymbol{z}_{\ell-1})\exp\left(-\tfrac{\gamma^-_\ell}{2}\left\|\boldsymbol{z}_\ell - \boldsymbol{r}^-_\ell\right\|^2 - \tfrac{\gamma^+_{\ell-1}}{2}\left\|\boldsymbol{z}_{\ell-1} - \boldsymbol{r}^+_{\ell-1}\right\|^2\right) \quad \text{(A.11a)}$$

$$q^*_\ell(\boldsymbol{z}_\ell) \propto \exp\left(-\tfrac{\gamma^-_\ell+\gamma^+_\ell}{2}\left\|\boldsymbol{z}_\ell - \tfrac{\gamma^-_\ell\boldsymbol{r}^-_\ell+\gamma^+_\ell\boldsymbol{r}^+_\ell}{\gamma^-_\ell+\gamma^+_\ell}\right\|^2\right) \quad \text{(A.11b)}$$

where notice that $f^*_\ell = b_\ell$ from (4.12). The primal feasibility KKT conditions (3.26) result in

$$\mathbb{E}\left[\boldsymbol{z}_\ell|f^*_\ell\right] = \mathbb{E}\left[\boldsymbol{z}_\ell|b_\ell\right] = \frac{\gamma^-_\ell\boldsymbol{r}^-_\ell + \gamma^+_\ell\boldsymbol{r}^+_\ell}{\gamma^-_\ell + \gamma^+_\ell}$$

$$\mathbb{E}\left[\|\boldsymbol{z}_\ell\|^2|f^*_\ell\right] = \mathbb{E}\left[\|\boldsymbol{z}_\ell\|^2|b_\ell\right] = (\gamma^-_\ell + \gamma^+_\ell)^{-1}$$

where we have used the Gaussianity of q_ℓ from (A.11b) and relation of $f^*_\ell = b_\ell$ from (A.11a) and (4.12). The quantity on the right is exactly $\widehat{\boldsymbol{z}}_\ell$ for any fixed point of MMSE-ML-VAMP as evident from (3.17). The claim follows from the update (3.11).

A.4 Proofs of Main Results: Theorems 9 and 8

Recall that the main result in Theorem 9 claims the empirical convergence of PL(2) statistics of iterates of the ML-VAMP algorithm 3 to the expectations corresponding statistics of random variables given in Algorithm 6. We prove this result by applying the general convergence result stated in Theorem 12 which shows that under Assumptions 5 and 6, the PL(2) statistics of iterates of Algorithm 10 empirically converge to expectations of corresponding statistics of appropriately defined scalar random variables defined in Algorithm 11.

The proof of Theorem 9 proceeds in two steps. First, we show that the ML-VAMP iterations are a special case of the iterations of Algorithm 10, and similarly Algorithm 6 is a special case of Algorithm 11, for specific choices of vector update functions, parameter statistic functions and parameter update functions, and their componentwise counterparts. The second step is to show that all assumptions required in Theorem 12 are satisfied, and hence the conclusions of Theorem 12 hold.

A.4.1 Proof of Theorem 9

We start by showing that the ML-VAMP iterations from Algorithm 3 are a special case of the Gen-ML recursions from Algorithm 10. Consider the singular value decompositions $\boldsymbol{W}_\ell = \boldsymbol{V}_\ell \operatorname{Diag}(\boldsymbol{s}_\ell)\boldsymbol{V}_{\ell-1}$ from equation (3.13). Then the true signals $\boldsymbol{z}_\ell^0$ in equation (4.1) and the iterates $\{\boldsymbol{r}_\ell^\pm, \widehat{\boldsymbol{z}}_\ell^\pm\}$ of Algorithm 3 can then be expressed via the *transformed* true signals defined below,

$$\begin{aligned} &\mathbf{q}_\ell^0 := \mathbf{z}_\ell^0, \quad \mathbf{p}_\ell^0 := \mathbf{V}_\ell \mathbf{z}_\ell^0 \quad \ell = 0, 2, \ldots, L \\ &\mathbf{q}_\ell^0 := \mathbf{V}_\ell^\intercal \mathbf{z}_\ell^0, \quad \mathbf{p}_\ell^0 := \mathbf{z}_\ell^0 \quad \ell = 1, 3, \ldots, L-1. \end{aligned} \tag{A.12}$$

These signals can be see in the (TOP) of Fig. A.1. Next, for $\ell = 0, 2, \ldots, L-2$, define:

$$\widehat{\mathbf{q}}^{\pm}_{k\ell} := \widehat{\mathbf{z}}^{\pm}_{k\ell}, \quad \mathbf{q}^{\pm}_{k\ell} := \mathbf{r}^{\pm}_{k\ell} - \mathbf{z}^{0}_{\ell}, \qquad \widehat{\mathbf{q}}^{\pm}_{k,\ell+1} := \mathbf{V}^{\mathsf{T}}_{\ell+1}\widehat{\mathbf{z}}^{\pm}_{k,\ell+1}, \quad \mathbf{q}^{\pm}_{k,\ell+1} := \mathbf{V}^{\mathsf{T}}_{\ell+1}(\mathbf{r}^{\pm}_{k,\ell+1} - \mathbf{z}^{0}_{\ell+1}) \tag{A.13a}$$

$$\widehat{\mathbf{p}}^{\pm}_{k\ell} := \mathbf{V}_{\ell}\widehat{\mathbf{z}}^{\pm}_{k\ell}, \quad \mathbf{p}^{\pm}_{k\ell} := \mathbf{V}_{\ell}(\mathbf{r}^{\pm}_{k\ell} - \mathbf{z}^{0}_{\ell}), \qquad \widehat{\mathbf{p}}^{\pm}_{k,\ell+1} := \widehat{\mathbf{z}}^{\pm}_{k,\ell+1}, \quad \mathbf{p}^{\pm}_{k,\ell+1} := \mathbf{r}^{\pm}_{k,\ell+1} - \mathbf{z}^{0}_{\ell+1}, \tag{A.13b}$$

The vectors $\widehat{\mathbf{q}}^{\pm}_{k\ell}$ and $\widehat{\mathbf{p}}^{\pm}_{k\ell}$ represent the estimates of $\mathbf{q}^{0}_{\ell}$ and $\mathbf{p}^{0}_{\ell}$ defined in (A.12). These are outputs of the estimators $\mathbf{g}^{\pm}_{\ell}$ and $\mathbf{G}^{\pm}_{\ell}$. Similarly, the vectors $\mathbf{q}^{\pm}_{k\ell}$ and $\mathbf{p}^{\pm}_{k\ell}$ are the differences $\mathbf{r}^{\pm}_{k\ell} - \mathbf{z}^{0}_{\ell}$ or their transforms. These represent errors on the *inputs* $\mathbf{r}^{\pm}_{k\ell}$ to the estimators $\mathbf{g}^{\pm}_{\ell}(\cdot)$ (even ℓ) and $\mathbf{G}^{\pm}_{\ell}$ (odd ℓ). These vectors can be seen in the (MIDDLE) panel of Fig. A.1

Lemma 2 (ML-VAMP as a special case of Gen-ML)**.** *Consider Algorithms 10 and 11 with*

1. *Initial functions $\mathbf{f}^{0}_{\ell}$ and vector update functions $\mathbf{f}^{\pm}_{\ell}$ given by componentwise extensions of f^{0}_{ℓ} and $f^{\pm}_{\ell}$ respectively from equation* (A.5)*. Parameter statistic functions $\boldsymbol{\varphi}^{+}_{\ell}$ and $\boldsymbol{\varphi}^{-}_{\ell}$ be given by componentwise extensions of $\frac{\partial f^{+}_{\ell}}{\partial q^{-}_{\ell}}$ and $\frac{\partial \mathbf{f}^{+}_{\ell}}{\partial p^{+}_{\ell-1}}$ respectively. Parameter updates $T^{\pm}_{k\ell}(\cdot)$ applied so that $\mu^{\pm}_{k\ell} = \alpha^{\pm}_{k\ell}$ and $\Lambda^{\pm}_{k\ell} = \theta^{\pm}_{k\ell}$, with $\theta^{\pm}_{k\ell}$ given in equation* (3.8)*.*

2. *Perturbation vectors $\boldsymbol{w}_{\ell}$ given by $\boldsymbol{w}_0 = \boldsymbol{z}^{0}_{0}$, $\boldsymbol{w}_{2\ell} = \boldsymbol{\xi}_{2\ell}$ and $\boldsymbol{w}_{2\ell-1} = (\boldsymbol{s}_{2\ell-1}, \overline{\mathbf{b}}_{2\ell-1}, \overline{\boldsymbol{\xi}}_{2\ell-1})$ for $\ell = 1, 2, \ldots \frac{L}{2}$. Perturbation random variables W_{ℓ} given by* (A.6)*.*

Then we have that

1. *Lines 2-6 of Algorithm 10 are equivalent to equation* (4.1) *with definitions of $\boldsymbol{p}^{0}_{\ell}, \boldsymbol{q}^{0}_{\ell}$ given in equation* (A.12)*. Lines 8-35 of Algorithm 10 are equivalent to the ML-VAMP iterations in Algorithm 3 with definitions of $\boldsymbol{p}^{\pm}_{\ell}, \widehat{\boldsymbol{p}}^{\pm}_{\ell}, \boldsymbol{q}^{\pm}_{\ell}, \widehat{\boldsymbol{q}}^{\pm}_{\ell}$, given in equation* (A.13)*.*

2. *Algorithm 11 is equivalent to Algorithm 6.*

Lemma 3. *Assumptions 5 and 6 are satisfied by the conditions in Theorem 9.*

The lemmas follow from the direct substitution of the quantities keeping in mind (3.13). As a consequence of the lemmas, we can apply the result of Theorem 12 under the conditions given in Theorem 9. The convergence of $(\alpha^{\pm}_{k\ell}, \gamma^{\pm}_{k\ell}, \eta^{\pm}_{k\ell})$ follows from the convergence of $\Lambda^{\pm}_{k\ell}$.

Theorem 12 leads to the conclusion that the following triplets are asymptotically normal

$$(\boldsymbol{z}^0_{\ell-1}, \boldsymbol{r}^+_{\ell-1} - \boldsymbol{z}^0_{\ell-1}, \boldsymbol{r}^-_\ell - \boldsymbol{z}^0_\ell) \equiv (\boldsymbol{p}^0_{\ell-1}, \boldsymbol{p}^+_{\ell-1}, \boldsymbol{q}^-_\ell), \qquad \forall\ \ell \text{ even},$$
$$\big(\boldsymbol{V}_{\ell-1}\boldsymbol{z}^0_{\ell-1}, \boldsymbol{V}_{\ell-1}(\boldsymbol{r}^+_{\ell-1} - \boldsymbol{z}^0_{\ell-1}), \boldsymbol{V}_\ell^\top(\boldsymbol{r}^-_\ell - \boldsymbol{z}^0_\ell)\big) \equiv (\boldsymbol{p}^0_{\ell-1}, \boldsymbol{p}^+_{\ell-1}, \boldsymbol{q}^-_\ell), \qquad \forall\ \ell \text{ odd}.$$

The results in Theorem 9 follows from the argument definition of PL(2) convergence defined in Appendix A.1

A.4.2 Proof of Theorem 8

Recall the update equations for $(\overline{\alpha}^\pm_{k\ell}, \overline{\gamma}^\pm_{k\ell}, \overline{\eta}^\pm_{k\ell})$ analogous to (3.9). Fix the iteration index k and let ℓ be even. We showed earlier after stating Theorem 9 that

$$\tfrac{1}{N_\ell}\left\|\widehat{\boldsymbol{z}}^+_{k\ell} - \boldsymbol{z}^0_\ell\right\| \xrightarrow{a.s.} \mathbb{E}\left(g^+_\ell(\mathsf{C} + \phi_\ell(\mathsf{A}, \Xi_\ell), \mathsf{B} + \mathsf{A}, \overline{\gamma}^-_{k\ell}, \overline{\gamma}^+_{k,\ell-1}) - \phi_\ell(\mathsf{A}, \Xi_\ell)\right)^2 =: \mathcal{E}^+_\ell(\overline{\gamma}^-_{k\ell}, \overline{\gamma}^+_{k,\ell-1})$$

We also know that $\eta^+_{k\ell} \xrightarrow{a.s.} \overline{\eta}^+_{k\ell} = \frac{\overline{\gamma}^-_{k\ell}}{\overline{\alpha}^+_{k\ell}}$. We need to show that the two limits coincide or equivalently $\frac{\overline{\alpha}^+_{k\ell}}{\overline{\gamma}^-_{k\ell}} = \mathcal{E}^+_\ell(\overline{\gamma}^-_{k\ell}, \overline{\gamma}^+_{k,\ell-1})$. In case of MMSE estimation, where $g^\pm_{\ell,\mathsf{mmse}}$ from (3.7) is applied, we can simplify $\overline{\alpha}^\pm_{k\ell}$. From line 11 of Algorithm 6, then we have

$$\overline{\alpha}^+_{k\ell} = \mathbb{E}\frac{\partial h^+_\ell}{\partial Q^-_\ell}(P^0_{\ell-1}, P^+_{k,\ell-1}, Q^-_{k\ell}, W_\ell, \overline{\theta}^+_{k\ell}) = \mathbb{E}\frac{\partial g^+_\ell}{\partial Q^-_\ell}(Q^-_{k\ell} + Q^0_\ell, P^+_{k,\ell-1} + P^0_{\ell-1}, \overline{\theta}^\pm_{k\ell})$$
$$= \mathbb{E}\frac{\partial}{\partial Q^-_\ell}\int \tfrac{p(z_\ell|z_{\ell-1})}{Z}\exp\big(-\frac{\overline{\gamma}^-_{k\ell}}{2}(z_\ell - Q^-_\ell - Q^0_\ell)^2 - \frac{\overline{\gamma}^+_{k,\ell-1}}{2}(z_{\ell-1} - P^+_{k,\ell-1} - P^0_{\ell-1})^2\big) z_\ell dz_\ell dz_{\ell-1},$$

for a normalizing factor Z. The last expectation above is with respect to the density of $(P^0_{\ell-1}, P^+_{k,\ell-1}, Q^-_{k\ell})$ which are Gaussian and $Q^0_\ell = \phi_\ell(P_{\ell-1}, \Xi_\ell)$. Exchanging the order of the integration and the partial derivative, gives the desired expression for $\mathcal{E}^+_\ell$.

A.5 General Multi-Layer Recursions

To analyze Algorithm 3, we consider a more general class of recursions as given in Algorithm 10 and depicted in Fig. A.1. The Gen-ML recursions generates (i) a set of *true vectors* $\boldsymbol{q}^0_\ell$ and $\boldsymbol{p}^0_\ell$ and (ii) *iterated vectors* $\mathbf{q}^\pm_{k\ell}$ and $\mathbf{p}^\pm_{k\ell}$. The true vectors are generated by a single forward pass, whereas the iterated vectors are generated via a sequence of forward and backward

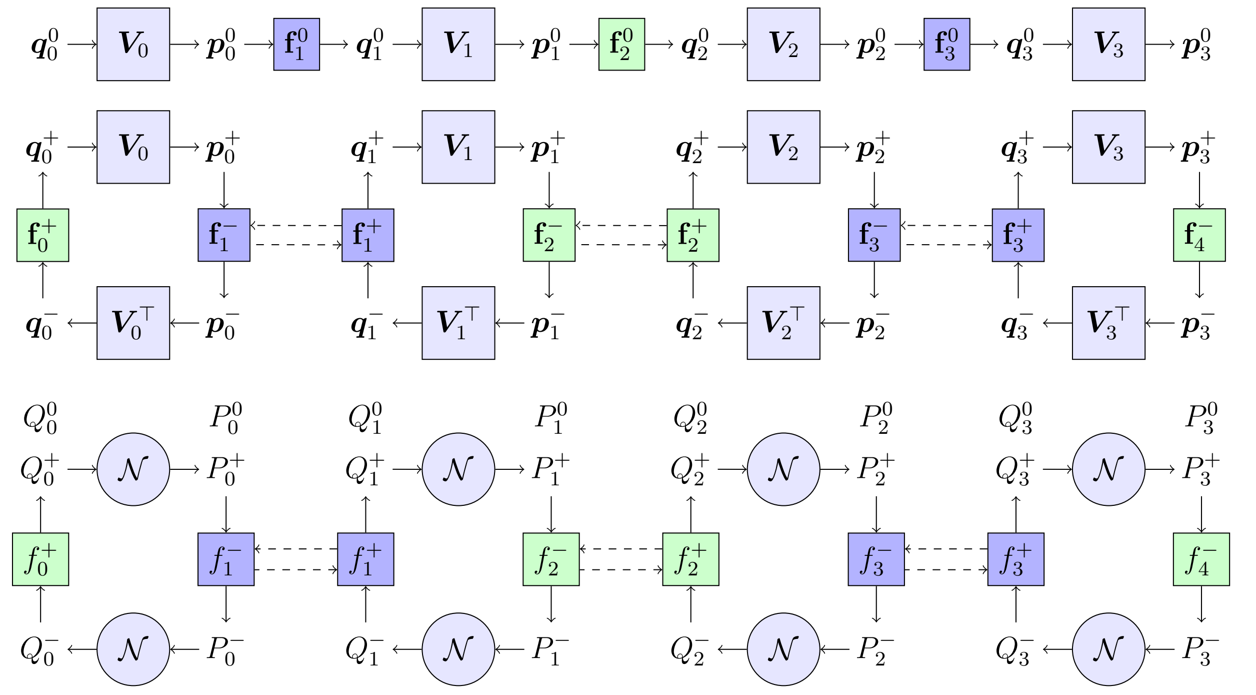

q_0^0 V_0 p_0^0 $\mathbf{f}_1^0$ q_1^0 V_1 p_1^0 $\mathbf{f}_2^0$ q_2^0 V_2 p_2^0 $\mathbf{f}_3^0$ q_3^0 V_3 p_3^0
q_0^+ V_0 p_0^+ q_1^+ V_1 p_1^+ q_2^+ V_2 p_2^+ q_3^+ V_3 p_3^+
$\mathbf{f}_0^+$ $\mathbf{f}_1^-$ $\mathbf{f}_1^+$ $\mathbf{f}_2^-$ $\mathbf{f}_2^+$ $\mathbf{f}_3^-$ $\mathbf{f}_3^+$ $\mathbf{f}_4^-$
q_0^- $V_0^\top$ p_0^- q_1^- $V_1^\top$ p_1^- q_2^- $V_2^\top$ p_2^- q_3^- $V_3^\top$ p_3^-
Q_0^0 P_0^0 Q_1^0 P_1^0 Q_2^0 P_2^0 Q_3^0 P_3^0
Q_0^+ $\mathcal{N}$ P_0^+ Q_1^+ $\mathcal{N}$ P_1^+ Q_2^+ $\mathcal{N}$ P_2^+ Q_3^+ $\mathcal{N}$ P_3^+
f_0^+ f_1^- f_1^+ f_2^- f_2^+ f_3^- f_3^+ f_4^-
Q_0^- $\mathcal{N}$ P_0^- Q_1^- $\mathcal{N}$ P_1^- Q_2^- $\mathcal{N}$ P_2^- Q_3^- $\mathcal{N}$ P_3^-

Figure A.1: (TOP) The equations (4.1) with equivalent quantities defined in (A.12). $\mathbf{f}_\ell^0$ defined using (A.5a) and (A.5b).
(MIDDLE) The GEN-ML recursions in Algorithm 10. These are also equivalent to ML-VAMP recursions from Algorithm 3 (See Lemma 7) if $\boldsymbol{p}^\pm, \boldsymbol{q}^\pm$ are as defined in equations (A.13) and $\mathbf{f}_\ell^\pm$ given by equations (A.5f-A.5i).
(BOTTOM) Quantities in the GEN-ML-SE recursions. These are also equivalent to ML-VAMP SE recursions from Algorithm 6 (See Lemma 7)

passes through a multi-layer system. In proving the State Evolution for the ML-VAMP algorithm, one would then associate the terms $\mathbf{q}_{k\ell}^\pm$ and $\mathbf{p}_{k\ell}^\pm$ with certain error quantities in the ML-VAMP recursions. To account for the effect of the parameters $\gamma_{k\ell}^\pm$ and $\alpha_{k\ell}^\pm$ in ML-VAMP, the Gen-ML algorithm describes the parameter update through a sequence of *parameter lists* $\Lambda_{k\ell}^\pm$. The parameter lists are ordered lists of parameters that accumulate as the algorithm progresses. The true and iterated vectors from Algorithm 10 are depicted in the signal flow graphs on the (TOP) and (MIDDLE) panel of Fig. A.1 respectively. The iteration index k for the iterated vectors $\boldsymbol{q}_{k\ell}, \boldsymbol{p}_{k\ell}$ has been dropped for simplifying notation.

The functions $\mathbf{f}_\ell^0(\cdot)$ that produce the true vectors $\boldsymbol{q}_\ell^0, \boldsymbol{p}_\ell^0$ are called *initial vector functions* and use the initial parameter list Λ_{01}^-. The functions $\mathbf{f}_{k\ell}^\pm(\cdot)$ that produce the vectors $\mathbf{q}_{k\ell}^\pm$ and

$\mathbf{p}^{\pm}_{k\ell}$ are called the *vector update functions* and use parameter lists $\Lambda^{\pm}_{kl}$. The parameter lists are initialized with Λ^{-}_{01} in line 2. As the algorithm progresses, new parameters $\lambda^{\pm}_{k\ell}$ are computed and then added to the lists in lines 12, 18, 25 and 31. The vector update functions $\mathbf{f}^{\pm}_{k\ell}(\cdot)$ may depend on any sets of parameters accumulated in the parameter list. In lines 11, 17, 24 and 30, the new parameters $\lambda^{\pm}_{k\ell}$ are computed by: (1) computing average values $\mu^{\pm}_{k\ell}$ of componentwise functions $\boldsymbol{\varphi}^{\pm}_{k\ell}(\cdot)$; and (2) taking functions $T^{\pm}_{k\ell}(\cdot)$ of the average values $\mu^{\pm}_{k\ell}$. Since the average values $\mu^{\pm}_{k\ell}$ represent statistics on the components of $\boldsymbol{\varphi}^{\pm}_{k\ell}(\cdot)$, we will call $\boldsymbol{\varphi}^{\pm}_{k\ell}(\cdot)$ the *parameter statistic functions.* We will call the $T^{\pm}_{k\ell}(\cdot)$ the *parameter update functions.* The functions $\mathbf{f}^{0}_{\ell}, \mathbf{f}^{\pm}_{k\ell}, \boldsymbol{\varphi}^{\pm}_{\ell}$ also take as input some perturbation vectors $\boldsymbol{w}_{\ell}$.

Similar to our analysis of the ML-VAMP Algorithm, we consider the following large-system limit (LSL) analysis of Gen-ML. Specifically, we consider a sequence of runs of the recursions indexed by N. For each N, let $N_\ell = N_\ell(N)$ be the dimension of the signals $\mathbf{p}^{\pm}_{\ell}$ and $\mathbf{q}^{\pm}_{\ell}$ as we assume that $\lim\limits_{N\to\infty} \frac{N_\ell}{N} = \beta_\ell \in (0,\infty)$ is a constant so that N_ℓ scales linearly with N. We then make the following assumptions. See Appendix A.1 for an overview of empirical convergence of sequences which we use in the assumptions.

Assumption 3. For vectors in the Gen-ML Algorithm (Algorithm 10), we assume:

(a) The matrices $\mathbf{V}_\ell$ are Haar distributed on the set of $N_\ell \times N_\ell$ orthogonal matrices and are independent from one another and from the vectors $\boldsymbol{q}^{0}_{0}$, $\mathbf{q}^{-}_{0\ell}$, perturbation vectors $\mathbf{w}_\ell$.

(b) The components of the initial conditions $\mathbf{q}^{-}_{0\ell}$, and perturbation vectors $\mathbf{w}_\ell$ converge jointly empirically with limits,

$$\lim_{N\to\infty} \{q^{-}_{0\ell,n}\} \overset{PL(2)}{=} Q^{-}_{0\ell}, \quad \lim_{N\to\infty} \{w_{\ell,n}\} \overset{PL(2)}{=} W_\ell, \tag{A.14}$$

where $Q^{-}_{0\ell}$ and W_ℓ are random variables such that $(Q^{-}_{00}, \cdots, Q^{-}_{0,L-1})$ is a jointly Gaussian random vector. Also, for $\ell = 0, \ldots, L-1$, the random variables $W_\ell, P^{0}_{\ell-1}$ and $Q^{-}_{0\ell}$ are independent. We also assume that the initial parameter list converges as

$$\lim_{N\to\infty} \Lambda^{-}_{01}(N) \xrightarrow{a.s.} \overline{\Lambda}^{-}_{01}, \tag{A.15}$$

to some list $\overline{\Lambda}_{01}^{-}$. The limit (B.11) means that every element in the list $\lambda(N) \in \Lambda_{01}^{-}(N)$ converges to a limit $\lambda(N) \to \overline{\lambda}$ as $N \to \infty$ almost surely.

(c) The *vector update functions* $\mathbf{f}_{k\ell}^{\pm}(\cdot)$ and *parameter update functions* $\boldsymbol{\varphi}_{k\ell}^{\pm}(\cdot)$ act componentwise. For e.g., in the k^{th} forward pass, at stage ℓ, we assume that for each output component n,

$$\begin{aligned}\left[\mathbf{f}_{k\ell}^{+}(\mathbf{p}_{\ell-1}^{0}, \mathbf{p}_{k,\ell-1}^{+}, \mathbf{q}_{k\ell}^{-}, \mathbf{w}_{\ell}, \Lambda_{k\ell}^{+})\right]_n &= f_{k\ell}^{+}(p_{\ell-1,n}^{0}, p_{k,\ell-1,n}^{+}, q_{k\ell,n}^{-}, w_{\ell,n}, \Lambda_{k\ell}^{+}) \\ \left[\boldsymbol{\varphi}_{k\ell}^{+}(\mathbf{p}_{\ell-1}^{0}, \mathbf{p}_{k,\ell-1}^{+}, \mathbf{q}_{k\ell}^{-}, \mathbf{w}_{\ell}, \Lambda_{k\ell}^{+})\right]_n &= \varphi_{k\ell}^{+}(p_{\ell-1,n}^{0}, p_{k,\ell-1,n}^{+}, q_{k\ell,n}^{-}, w_{\ell,n}, \Lambda_{k\ell}^{+}),\end{aligned}$$

for some scalar-valued functions $f_{k\ell}^{+}(\cdot)$ and $\varphi_{k\ell}^{+}(\cdot)$. Similar definitions apply in the reverse directions and for the initial vector functions $\mathbf{f}_{\ell}^{0}(\cdot)$. We will call $f_{k\ell}^{\pm}(\cdot)$ the *vector update component functions* and $\varphi_{k\ell}^{\pm}(\cdot)$ the *parameter update component functions.*

Next we define a set of *deterministic* constants $\{\mathbf{K}_{k\ell}^{+}, \tau_{k\ell}^{-}, \overline{\mu}_{k\ell}^{\pm}, \overline{\Lambda}_{kl}^{\pm}, \tau_{\ell}^{0}\}$ and *scalar* random variables $\{Q_{\ell}^{0}, P_{\ell}^{0}, Q_{k\ell}^{\pm}, P_{\ell}^{\pm}\}$ which are recursively defined through Algorithm 11, which we call the *Gen-ML State Evolution* (SE). These recursions in Algorithm closely mirror those in the Gen-ML algorithm (Algorithm 10). The vectors $\mathbf{q}_{k\ell}^{\pm}$ and $\mathbf{p}_{k\ell}^{\pm}$ are replaced by random variables $Q_{k\ell}^{\pm}$ and $P_{k\ell}^{\pm}$; the vector and parameter update functions $\mathbf{f}_{k\ell}^{\pm}(\cdot)$ and $\boldsymbol{\varphi}_{k\ell}^{\pm}(\cdot)$ are replaced by their component functions $f_{k\ell}^{\pm}(\cdot)$ and $\varphi_{k\ell}^{\pm}(\cdot)$; and the parameters $\lambda_{k\ell}^{\pm}$ are replaced by their limits $\overline{\lambda}_{k\ell}^{\pm}$. We refer to $\{Q_{\ell}^{0}, P_{\ell}^{0}\}$ as *true random variables* and $\{Q_{k\ell}^{\pm}, P_{kl}^{\pm}\}$ as iterated random variables. The signal flow graph for the true and iterated random variables in Algorithm 11 is given in the (BOTTOM) panel of Fig. A.1. The iteration index k for the iterated random variables $\{Q_{k\ell}^{\pm}, P_{kl}^{\pm}\}$ to simplify notation.

We also assume the following about the behaviour of component functions around the quantities defined in Algorithm 11. The iteration index k has been dropped for simplifying notation.

Assumption 4. For component functions f, φ and parameter update functions T we assume:

(a) $T_{k\ell}^{\pm}(\mu_{k\ell}^{\pm}, \cdot)$ are continuous at $\mu_{k\ell}^{\pm} = \overline{\mu}_{k\ell}^{\pm}$

(b) $f^+_{k\ell}(p^0_{\ell-1}, p^+_{k,\ell-1}, q^-_{k\ell}, w_\ell, \Lambda^+_{k\ell})$, $\frac{\partial f^+_{k\ell}}{\partial q^-_{k\ell}}(p^0_{\ell-1}, p^+_{k,\ell-1}, q^-_{k\ell}, w_\ell, \Lambda^+_{k\ell})$ and $\varphi^+_{k\ell}(p^0_{\ell-1}, p^+_{k,\ell-1}, q^-_{k\ell}, w_\ell, \Lambda^+_{k,\ell-1})$ are uniformly Lipschitz continuous in $(p^0_{\ell-1}, p^+_{k,\ell-1}, q^-_{k\ell}, w_\ell)$ at $\Lambda^+_{k\ell} = \overline{\Lambda}^+_{k\ell}$, $\Lambda^+_{k,\ell-1} = \overline{\Lambda}^+_{k,\ell-1}$. Similarly, $f^-_{k+1,\ell}(p^0_{\ell-1}, p^+_{k,\ell-1}, q^-_{k+1,\ell}, w_\ell, \Lambda^-_{k\ell})$, $\frac{\partial f^-_{k\ell}}{\partial p^+_{k,\ell-1}}(p^0_{\ell-1}, p^+_{k,\ell-1}, q^-_{k+1,\ell}, w_\ell, \Lambda^-_{k\ell})$, and $\varphi^-_{k\ell}(p^0_{\ell-1}, p^+_{k,\ell-1}, q^-_{k+1,\ell}, w_\ell, \Lambda^-_{k+1,\ell+1})$ are uniformly Lipschitz continuous in $(p^0_{\ell-1}, p^+_{k,\ell-1}, q^-_{k+1,\ell}, w_\ell)$ at $\Lambda^-_{k\ell} = \overline{\Lambda}^-_{k\ell}$, $\Lambda^-_{k+1,\ell+1} = \overline{\Lambda}^-_{k+1,\ell+1}$.

(c) $f^0_\ell(p^0_{\ell-1}, w_\ell, \Lambda^-_{01})$ are uniformly Lipschitz continuous in $(p^0_{k,\ell-1}, w_\ell)$ at $\Lambda^-_{k+1,\ell} = \overline{\Lambda}^-_{k+1,\ell}$.

(d) Vector update functions $\mathbf{f}^\pm_{k\ell}$ are *asymptotically divergence free* meaning

$$\lim_{N\to\infty} \left\langle \frac{\partial \mathbf{f}^+_{k\ell}}{\partial \mathbf{q}^-_{k\ell}}(\mathbf{p}^+_{k,\ell-1}, \mathbf{q}^-_{k\ell}, \mathbf{w}_\ell, \overline{\Lambda}^+_{k\ell}) \right\rangle = 0, \quad \lim_{N\to\infty} \left\langle \frac{\partial \mathbf{f}^-_{k\ell}}{\partial \mathbf{p}^+_{k,\ell-1}}(\mathbf{p}^+_{k,\ell-1}, \mathbf{q}^-_{k+1,\ell}, \mathbf{w}_\ell, \overline{\Lambda}^-_{k\ell}) \right\rangle = 0 \tag{A.16}$$

We are now ready to state the general result regarding the empirical convergence of the true and iterated vectors from Algorithm 10 in terms of random variables defined in Algorithm 11.

Theorem 11. *Consider the iterates of the Gen-ML recursion (Algorithm 10) and the corresponding random variables and parameter limits defined by the SE recursions (Algorithm 11) under Assumptions 5 and 6. Then,*

(a) For any fixed $k \geq 0$ and fixed $\ell = 1, \ldots, L-1$, the parameter list $\Lambda^+_{k\ell}$ converges as

$$\lim_{N\to\infty} \Lambda^+_{k\ell} = \overline{\Lambda}^+_{k\ell} \tag{A.17}$$

almost surely. Also, the components of $\mathbf{w}_\ell$, $\mathbf{p}^0_{\ell-1}$, $\mathbf{q}^0_\ell$, $\mathbf{p}^+_{0,\ell-1}, \ldots, \mathbf{p}^+_{k,\ell-1}$ and $\mathbf{q}^\pm_{0\ell}, \ldots, \mathbf{q}^\pm_{k\ell}$ almost surely jointly converge empirically with limits,

$$\lim_{N\to\infty} \left\{(p^0_{\ell-1,n}, p^+_{i,\ell-1,n}, q^-_{j\ell,n}, q^0_{\ell,n}, q^+_{j\ell,n})\right\} \overset{PL(2)}{=} (P^0_{\ell-1}, P^+_{i,\ell-1}, Q^-_{j\ell}, Q^0_\ell, Q^+_{j\ell}), \tag{A.18}$$

for all $0 \leq i, j \leq k$, where the variables $P^0_{\ell-1}$, $P^+_{i,\ell-1}$ and $Q^-_{j\ell}$ are zero-mean jointly Gaussian random variables independent of W_ℓ and with covariance matrix given by

$$\mathrm{Cov}(P^0_{\ell-1}, P^+_{i,\ell-1}) = \mathbf{K}^+_{i,\ell-1}, \quad \mathbb{E}(Q^-_{j\ell})^2 = \tau^-_{j\ell}, \quad \mathbb{E}(P^+_{i,\ell-1}Q^-_{j\ell}) = 0, \quad \mathbb{E}(P^0_{\ell-1}Q^-_{j\ell}) = 0, \tag{A.19}$$

and Q^0_ℓ and $Q^+_{j\ell}$ are the random variable in line 19:

$$Q^0_\ell = f^0_\ell(P^0_{\ell-1}, W_\ell), \quad Q^+_{j\ell} = f^+_\ell(P^0_{\ell-1}, P^+_{j,\ell-1}, Q^-_{j\ell}, W_\ell, \overline{\Lambda}^+_{j\ell}). \tag{A.20}$$

An identical result holds for $\ell = 0$ with all the variables $\mathbf{p}^+_{i,\ell-1}$ and $P^+_{i,\ell-1}$ removed.

(b) *For any fixed $k \geq 1$ and fixed $\ell = 1, \ldots, L-1$, the parameter lists $\Lambda^-_{k\ell}$ converge as*

$$\lim_{N\to\infty} \Lambda^-_{k\ell} = \overline{\Lambda}^-_{k\ell} \tag{A.21}$$

almost surely. Also, the components of $\mathbf{w}_\ell$, $\mathbf{p}^0_{\ell-1}$, $\mathbf{p}^\pm_{0,\ell-1}, \ldots, \mathbf{p}^\pm_{k-1,\ell-1}$, and $\mathbf{q}^-_{0\ell}, \ldots, \mathbf{q}^-_{k\ell}$ almost surely jointly converge empirically with limits,

$$\lim_{N\to\infty} \left\{(p^0_{\ell-1,n}, p^+_{i,\ell-1,n}, q^-_{j\ell,n}, p^-_{j,\ell-1,n})\right\} \overset{PL(2)}{=} (P^0_{\ell-1}, P^+_{i,\ell-1}, Q^-_{j\ell}, P^-_{j,\ell-1}), \tag{A.22}$$

for all $0 \leq i \leq k-1$ and $0 \leq j \leq k$, where the variables $P^0_{\ell-1}$, $P^+_{i,\ell-1}$ and $Q^-_{j\ell}$ are zero-mean jointly Gaussian random variables independent of W_ℓ and with covariance matrix given by equation (B.15) *and $P^-_{j\ell}$ is the random variable in line 32:*

$$P^-_{j\ell} = f^-_\ell(P^0_{\ell-1}, P^+_{j-1,\ell-1}, Q^-_{j\ell}, W_\ell, \overline{\Lambda}^-_{j\ell}). \tag{A.23}$$

An identical result holds for $\ell = L$ with all the variables $\mathbf{q}^-_{j\ell}$ and $Q^-_{j\ell}$ removed.

For $k = 0$, $\Lambda^-_{01} \to \overline{\Lambda}^-_{01}$ almost surely, and $\{(w_{\ell,n}, p^0_{\ell-1,n}, q^-_{j\ell,n})\}$ empirically converge to independent random variables $(W_\ell, P^0_{\ell-1}, Q^-_{0\ell})$.

Proof. Appendix B.5 in the supplementary materials is dedicated to proving this result. □

A.6 Proof of Theorem 12

A.6.1 Overview of the Induction Sequence

The proof is similar to that of [127, Theorem 4], which provides a SE analysis for VAMP on a single-layer network. The critical challenge here is to extend that proof to multi-layer recursions. Many of the ideas in the two proofs are similar, so we highlight only the key differences between the two.

Similar to the SE analysis of VAMP in [127], we use an induction argument. However, for the multi-layer proof, we must index over both the iteration index k and layer index ℓ. To this end, let $\mathcal{H}^+_{k\ell}$ and $\mathcal{H}^-_{k\ell}$ be the hypotheses:

- $\mathcal{H}^+_{k\ell}$: The hypothesis that Theorem 12(a) is true for a given k and ℓ, where $0 \le \ell \le L-1$.
- $\mathcal{H}^-_{k\ell}$: The hypothesis that Theorem 12(b) is true for a given k and ℓ, where $1 \le \ell \le L$.

We prove these hypotheses by induction via a sequence of implications,

$$\{\mathcal{H}^-_{0\ell}\}_{\ell=1}^{L} \cdots \Rightarrow \mathcal{H}^-_{k1} \Rightarrow \mathcal{H}^+_{k0} \Rightarrow \cdots \Rightarrow \mathcal{H}^+_{k,L-1} \Rightarrow \mathcal{H}^-_{k+1,L} \Rightarrow \cdots \Rightarrow \mathcal{H}^-_{k+1,1} \Rightarrow \cdots, \tag{A.24}$$

beginning with the hypotheses $\{\mathcal{H}^-_{0\ell}\}$ for all $\ell = 1, \ldots, L-1$.

A.6.2 Base Case: Proof of $\{\mathcal{H}^-_{0\ell}\}_{\ell=1}^{L}$

The base case corresponds to the Hypotheses $\{\mathcal{H}^-_{0\ell}\}_{\ell=1}^{L}$. Note that Theorem 12(b) states that for $k = 0$, we need $\Lambda^-_{01} \to \overline{\Lambda}^-_{01}$ almost surely, and $\{(w_{\ell,n}, p^0_{\ell-1,n}, q^-_{j\ell,n})\}$ empirically converge to independent random variables $(W_\ell, P^0_{\ell-1}, Q^-_{0\ell})$. These follow directly from equations (B.10) and (B.11) in Assumption 1 (a).

A.6.3 Inductive Step: Proof of $\mathcal{H}^+_{k,\ell+1}$

Fix a layer index $\ell = 1, \ldots, L-1$ and an iteration index $k = 0, 1, \ldots$. We show the implication $\cdots \implies \mathcal{H}^+_{k,\ell+1}$ in (B.20). All other implications can be proven similarly using symmetry arguments.

Definition 4 (Induction hypothesis)**.** The hypotheses prior to $\mathcal{H}^+_{k,\ell+1}$ in the sequence (B.20), but not including $\mathcal{H}^+_{k,\ell+1}$, are true.

The inductive step then corresponds to the following result.

Lemma 4. *Under the induction hypothesis, $\mathcal{H}^+_{k,\ell+1}$ holds*

Before proving the inductive step in Lemma 9, we prove two intermediate lemmas. Let us start by defining some notation. Define $\mathbf{P}^+_{k\ell} := \left[\mathbf{p}^+_{0\ell} \cdots \mathbf{p}^+_{k\ell}\right] \in \boldsymbol{R}^{N_\ell \times (k+1)}$, be a matrix whose columns are the first $k+1$ values of the vector $\mathbf{p}^+_\ell$. We define the matrices $\mathbf{P}^-_{k\ell}$, $\mathbf{Q}^+_{k\ell}$ and $\mathbf{Q}^-_{k\ell}$ in a similar manner with values of $\mathbf{p}^-_\ell, \mathbf{q}^+_\ell$ and $\mathbf{q}^-_\ell$ respectively.

Note that except the initial vectors $\{\mathbf{w}_\ell, \mathbf{q}^-_{0\ell}\}_{\ell=1}^L$, all later iterates in Algorithm 10 are random due to the randomness of $\boldsymbol{V}_\ell$. Let $\mathfrak{G}^\pm_{k\ell}$ denote the collection of random variables associated with the hypotheses, $\mathcal{H}^\pm_{k\ell}$. That is, for $\ell = 1, \ldots, L-1$,

$$\mathfrak{G}^+_{k\ell} := \left\{\mathbf{w}_\ell, \mathbf{p}^0_{\ell-1}, \mathbf{P}^+_{k,\ell-1}, \mathbf{q}^0_\ell, \mathbf{Q}^-_{k\ell}, \mathbf{Q}^+_{k\ell}\right\}, \quad \mathfrak{G}^-_{k\ell} := \left\{\mathbf{w}_\ell, \mathbf{p}^0_{\ell-1}, \mathbf{P}^+_{k-1,\ell-1}, \mathbf{q}^0_\ell, \mathbf{Q}^-_{k\ell}, \mathbf{P}^-_{k,\ell-1}\right\}. \quad \text{(A.25)}$$

For $\ell = 0$ and $\ell = L$ we set, $\mathfrak{G}^+_{k0} := \left\{\mathbf{w}_0, \mathbf{Q}^-_{k0}, \mathbf{Q}^+_{k0}\right\}, \quad \mathfrak{G}^-_{kL} := \left\{\mathbf{w}_L, \mathbf{p}^0_{L-1}, \mathbf{P}^+_{k-1,L-1}, \mathbf{P}^-_{k,L-1}\right\}.$

Let $\overline{\mathfrak{G}}^+_{k\ell}$ be the sigma algebra generated by the union of all the sets $\mathfrak{G}^\pm_{k'\ell'}$ as they have appeared in the sequence (B.20) up to and including the final set $\mathfrak{G}^+_{k\ell}$. Thus, the sigma algebra $\overline{\mathfrak{G}}^+_{k\ell}$ contains all *information* produced by Algorithm 10 immediately *before* line 20 in layer ℓ of iteration k. Note also that the random variables in Algorithm 11 immediately before defining $P^+_{k,\ell}$ in line 20 are all $\overline{\mathfrak{G}}^+_{k\ell}$ measurable.

Observe that the matrix $\mathbf{V}_\ell$ in Algorithm 10 appears only during matrix-vector multiplications in lines 20 and 32. If we define the matrices, $\mathbf{A}_{k\ell} := \left[\mathbf{p}^0_\ell, \mathbf{P}^+_{k-1,\ell}\ \mathbf{P}^-_{k\ell}\right], \quad \mathbf{B}_{k\ell} := \left[\mathbf{q}^0_\ell, \mathbf{Q}^+_{k-1,\ell}\ \mathbf{Q}^-_{k\ell}\right]$, all the vectors in the set $\overline{\mathfrak{G}}^+_{k\ell}$ will be unchanged for all matrices $\mathbf{V}_\ell$ satisfying the linear constraints

$$\mathbf{A}_{k\ell} = \mathbf{V}_\ell \mathbf{B}_{k\ell}. \quad \text{(A.26)}$$

Hence, the conditional distribution of $\mathbf{V}_\ell$ given $\overline{\mathfrak{G}}^+_{k\ell}$ is precisely the uniform distribution on the set of orthogonal matrices satisfying (B.21). The matrices $\mathbf{A}_{k\ell}$ and $\mathbf{B}_{k\ell}$ are of dimensions $N_\ell \times 2k+2$. From [127, Lemmas 3,4], this conditional distribution is given by

$$\mathbf{V}_\ell|_{\overline{\mathfrak{G}}^+_{k\ell}} \stackrel{d}{=} \mathbf{A}_{k\ell}(\mathbf{A}^\mathsf{T}_{k\ell}\mathbf{A}_{k\ell})^{-1}\mathbf{B}^\mathsf{T}_{k\ell} + \mathbf{U}_{\mathbf{A}^\perp_{k\ell}} \widetilde{\mathbf{V}}_\ell \mathbf{U}^\mathsf{T}_{\mathbf{B}^\perp_{k\ell}}, \quad \text{(A.27)}$$

where $\mathbf{U}_{\mathbf{A}^\perp_{k\ell}}$ and $\mathbf{U}_{\mathbf{B}^\perp_{k\ell}}$ are $N_\ell \times (N_\ell - (2k+2))$ matrices whose columns are an orthonormal basis for Range$(\mathbf{A}_{k\ell})^\perp$ and Range$(\mathbf{B}_{k\ell})^\perp$. The matrix $\widetilde{\mathbf{V}}_\ell$ is Haar distributed on the set of $(N_\ell - (2k+2)) \times (N_\ell - (2k+2))$ orthogonal matrices and is independent of $\overline{\mathfrak{G}}^+_{k\ell}$.

Next, similar to the proof of [127, Thm. 4], we can use (B.22) to write the conditional distribution of $\mathbf{p}^+_{k\ell}$ (from line 20 of Algorithm 10) given $\overline{\mathfrak{G}}^+_{k\ell}$ as a sum of two terms

$$\mathbf{p}^+_{k\ell}|_{\overline{\mathfrak{G}}^+_{k\ell}} = \mathbf{V}_\ell|_{\overline{\mathfrak{G}}^+_{k\ell}} \, \mathbf{q}^+_{k\ell} \overset{d}{=} \mathbf{p}^{+\text{det}}_{k\ell} + \mathbf{p}^{+\text{ran}}_{k\ell}, \tag{A.28a}$$

$$\mathbf{p}^{+\text{det}}_{k\ell} := \mathbf{A}_{k\ell}(\mathbf{B}^\mathsf{T}_{k\ell}\mathbf{B}_{k\ell})^{-1}\mathbf{B}^\mathsf{T}_{k\ell}\mathbf{q}^+_{k\ell} \tag{A.28b}$$

$$\mathbf{p}^{+\text{ran}}_{k\ell} := \mathbf{U}_{\mathbf{B}^\perp_k}\widetilde{\mathbf{V}}^\mathsf{T}_\ell\mathbf{U}^\mathsf{T}_{\mathbf{A}^\perp_k}\mathbf{q}^+_{k\ell}. \tag{A.28c}$$

where we call $\mathbf{p}^{+\text{det}}_{k\ell}$ the *deterministic* term and $\mathbf{p}^{+\text{ran}}_{k\ell}$ the *random* term. The next two lemmas characterize the limiting distributions of the deterministic and random terms.

Lemma 5. *Under the induction hypothesis, the components of the "deterministic" term $\mathbf{p}^{+\text{det}}_{k\ell}$ along with the components of the vectors in $\overline{\mathfrak{G}}^+_{k\ell}$ converge empirically. In addition, there exists constants $\beta^+_{0\ell}, \ldots, \beta^+_{k-1,\ell}$ such that*

$$\lim_{N\to\infty}\{p^{+\text{det}}_{k\ell,n}\} \overset{PL(2)}{=} P^{+\text{det}}_{k\ell} := \beta^0_\ell P^0_\ell + \sum_{i=0}^{k-1}\beta_{i\ell}P^+_{i\ell}, \tag{A.29}$$

where $P^{+\text{det}}_{k\ell}$ is the limiting random variable for the components of $\mathbf{p}^{\text{det}}_{k\ell}$.

Proof. The proof is similar that of [127, Lem. 6], but we go over the details as there are some important differences in the multi-layer case. Define $\widetilde{\mathbf{P}}^+_{k-1,\ell} = \left[\mathbf{p}^0_\ell,\ \mathbf{P}^+_{k-1,\ell}\right]$, $\widetilde{\mathbf{Q}}^+_{k-1,\ell} = \left[\mathbf{q}^0_\ell,\ \mathbf{Q}^+_{k-1,\ell}\right]$, which are the matrices in $\mathbb{R}^{N_\ell\times(k+1)}$. We can then write $\mathbf{A}_{k\ell}$ and $\mathbf{B}_{k\ell}$ from (B.21) as

$$\mathbf{A}_{k\ell} := \left[\widetilde{\mathbf{P}}^+_{k-1,\ell}\ \mathbf{P}^-_{k\ell}\right], \quad \mathbf{B}_{k\ell} := \left[\widetilde{\mathbf{Q}}^+_{k-1,\ell}\ \mathbf{Q}^-_{k\ell}\right], \tag{A.30}$$

We first evaluate the asymptotic values of various terms in (B.23b). By definition of $\mathbf{B}_{k\ell}$ in (B.25),

$$\mathbf{B}^\mathsf{T}_{k\ell}\mathbf{B}_{k\ell} = \begin{bmatrix} (\widetilde{\mathbf{Q}}^+_{k-1,\ell})^\mathsf{T}\widetilde{\mathbf{Q}}^+_{k-1,\ell} & (\widetilde{\mathbf{Q}}^+_{k-1,\ell})^\mathsf{T}\mathbf{Q}^-_{k\ell} \\ (\mathbf{Q}^-_{k\ell})^\mathsf{T}\widetilde{\mathbf{Q}}^+_{k-1,\ell} & (\mathbf{Q}^-_{k\ell})^\mathsf{T}\mathbf{Q}^-_{k\ell} \end{bmatrix}$$

We can then evaluate the asymptotic values of these terms as follows: For $0 \le i, j \le k-1$ the asymptotic value of the $(i+2, j+2)^{\text{nd}}$ entry of the matrix $(\widetilde{\mathbf{Q}}^+_{k-1,\ell})^\mathsf{T}\widetilde{\mathbf{Q}}^+_{k-1,\ell}$ is given by

$$\lim_{N\to\infty}\tfrac{1}{N_\ell}\left[(\widetilde{\mathbf{Q}}^+_{k-1,\ell})^\mathsf{T}\widetilde{\mathbf{Q}}^+_{k-1,\ell}\right]_{i+2,j+2} \overset{(a)}{=} \lim_{N\to\infty}\frac{1}{N_\ell}(\mathbf{q}^+_{i\ell})^\mathsf{T}\mathbf{q}^+_{j\ell} = \lim_{N\to\infty}\tfrac{1}{N_\ell}\sum_{n=1}^{N_\ell}q^+_{i\ell,n}q^+_{j\ell,n} \overset{(b)}{=} \mathbb{E}\left[Q^+_{i\ell}Q^+_{j\ell}\right]$$

where (a) follows since the $(i+2)^{\text{th}}$ column of $\widetilde{\mathbf{Q}}^+_{k-1,\ell}$ is $\mathbf{q}^+_{i\ell}$, and (b) follows due to the empirical convergence assumption in (B.14). Also, since the first column of $\widetilde{\mathbf{Q}}^+_{k-1,\ell}$ is $\mathbf{q}^0_\ell$, we obtain that

$$\lim_{N_\ell\to\infty} \tfrac{1}{N_\ell}(\widetilde{\mathbf{Q}}^+_{k-1,\ell})^\mathsf{T}\widetilde{\mathbf{Q}}^+_{k-1,\ell} = \mathbf{R}^+_{k-1,\ell} \qquad \text{and} \qquad \lim_{N_\ell\to\infty} \tfrac{1}{N_\ell}(\mathbf{Q}^-_{k\ell})^\mathsf{T}\mathbf{Q}^-_{k\ell} = \mathbf{R}^-_{k\ell},$$

where $\mathbf{R}^+_{k-1,\ell}$ is the covariance matrix of $(Q^0_\ell, Q^+_{0\ell}, \ldots, Q^+_{k-1,\ell})$, and $\mathbf{R}^-_{k\ell}$ is the covariance matrix of the vector $(Q^-_{0\ell}, \ldots, Q^-_{k\ell})$. For the matrix $(\widetilde{\mathbf{Q}}^+_{k-1,\ell})^\mathsf{T}\mathbf{Q}^-_{k\ell}$, first observe that the limit of the divergence free condition (B.12) implies

$$\mathbb{E}\left[\frac{\partial f^+_{i\ell}(P^+_{i,\ell-1}, Q^-_{i\ell}, W_\ell, \overline{\Lambda}_{i\ell})}{\partial q^-_{i\ell}}\right] = \lim_{N_\ell\to\infty}\left\langle \frac{\partial \mathbf{f}^+_{i\ell}(\mathbf{p}^+_{i,\ell-1}, \mathbf{q}^-_{i\ell}, \mathbf{w}_\ell, \overline{\Lambda}^+_{i\ell})}{\partial \mathbf{q}^-_{i\ell}}\right\rangle = 0, \tag{A.31}$$

for any i. Also, by the induction hypothesis $\mathcal{H}^+_{k\ell}$,

$$\mathbb{E}(P^+_{i,\ell-1}Q^-_{j\ell}) = 0, \quad \mathbb{E}(P^0_{\ell-1}Q^-_{j\ell}) = 0, \tag{A.32}$$

for all $0 \le i, j \le k$. Therefore using (B.16), the cross-terms $\mathbb{E}(Q^+_{i\ell}Q^-_{j\ell})$ are given by

$$\begin{aligned}\mathbb{E}(f^+_{i\ell}(P^0_{\ell-1}, P^+_{i,\ell-1}, Q^-_{i\ell}, W_\ell, \overline{\Lambda}_{i\ell})Q^-_{j\ell}) &\overset{(a)}{=} \mathbb{E}\left[\tfrac{\partial f^+_{i\ell}(P^0_{\ell-1}, P^+_{i,\ell-1}, Q^-_{i\ell}, W_\ell, \overline{\Lambda}^+_{i\ell})}{\partial P^0_{\ell-1}}\right]\mathbb{E}(P^0_{\ell-1}Q^-_{j\ell}) \\ + \mathbb{E}\left[\tfrac{\partial f^+_{i\ell}(P^0_{\ell-1}, P^+_{i,\ell-1}, Q^-_{i\ell}, W_\ell, \overline{\Lambda}^+_{i\ell})}{\partial P^+_{i,\ell-1}}\right]\mathbb{E}(P^+_{i,\ell-1}Q^-_{j\ell}) &+ \mathbb{E}\left[\tfrac{\partial f^+_{i\ell}(P^0_{\ell-1}, P^+_{i,\ell-1}, Q^-_{i\ell}, W_\ell, \overline{\Lambda}^+_{i\ell})}{\partial Q^-_{i\ell}}\right]\mathbb{E}(Q^-_{i\ell}Q^-_{j\ell}) \overset{(b)}{=} 0,\end{aligned} \tag{A.33}$$

(a) follows from Stein's Lemma; and (b) follows from (B.27), and (B.28). Consequently,

$$\lim_{N_\ell\to\infty} \tfrac{1}{N_\ell}\mathbf{B}^\mathsf{T}_{k\ell}\mathbf{B}_{k\ell} = \begin{bmatrix} \mathbf{R}^+_{k-1,\ell} & \mathbf{0} \\ \mathbf{0} & \mathbf{R}^-_{k\ell}\end{bmatrix}, \quad \text{and} \quad \lim_{N_\ell\to\infty} \tfrac{1}{N_\ell}\mathbf{B}^\mathsf{T}_{k\ell}\mathbf{q}^+_{k\ell} = \begin{bmatrix} \mathbf{b}^+_{k\ell} \\ \mathbf{0}\end{bmatrix}, \tag{A.34}$$

where $\mathbf{b}^+_{k\ell} := \left[\mathbb{E}(Q^+_{0\ell}Q^+_{k\ell}),\ \mathbb{E}(Q^+_{1\ell}Q^+_{k\ell}),\ \cdots, \mathbb{E}(Q^+_{k-1,\ell}Q^+_{k\ell})\right]^\mathsf{T}$, is the vector of correlations. We again have $\mathbf{0}$ in the second term because $\mathbb{E}[Q^+_{i\ell}Q_{j\ell-}] = 0$ for all $0 \le i, j \le k$. Hence we have

$$\lim_{N_\ell\to\infty} (\mathbf{B}^\mathsf{T}_{k\ell}\mathbf{B}_{k\ell})^{-1}\mathbf{B}^\mathsf{T}_{k\ell}\mathbf{q}^+_{k\ell} = \begin{bmatrix} \boldsymbol{\beta}^+_{k\ell} \\ \mathbf{0}\end{bmatrix}, \quad \boldsymbol{\beta}^+_{k\ell} := \left[\mathbf{R}^+_{k-1,\ell}\right]^{-1}\mathbf{b}^+_{k\ell}. \tag{A.35}$$

Therefore, $\mathbf{p}^{+\text{det}}_{k\ell}$ equals

$$\mathbf{A}_{k\ell}(\mathbf{B}^\mathsf{T}_{k\ell}\mathbf{B}_{k\ell})^{-1}\mathbf{B}^\mathsf{T}_{k\ell}\mathbf{q}^+_{k\ell} = \left[\widetilde{\mathbf{P}}^+_{k-1,\ell}\ \mathbf{P}^-_{k,\ell}\right]\begin{bmatrix} \beta^+_{k\ell} \\ \mathbf{0}\end{bmatrix} + O\left(\tfrac{1}{N_\ell}\right) = \beta^0_\ell\mathbf{p}^0_\ell + \sum_{i=0}^{k-1}\beta^+_{i\ell}\mathbf{p}^+_{i\ell} + O\left(\tfrac{1}{N_\ell}\right),$$

where β^0_ℓ and $\beta^+_{i\ell}$ are the components of $\boldsymbol{\beta}^+_{k\ell}$ and the term $O(\frac{1}{N_\ell})$ means a vector sequence,

$\boldsymbol{\xi}(N) \in \boldsymbol{R}^{N_\ell}$ such that $\lim_{N\to\infty} \frac{1}{N}\|\boldsymbol{\xi}(N)\|^2 = 0$. A continuity argument then shows the empirical convergence (B.24). □

Lemma 6. *Under the induction hypothesis, the components of the "random" term* $\mathbf{p}^{+\mathrm{ran}}_{k\ell}$ *along with the components of the vectors in* $\overline{\mathfrak{G}}^{+}_{k\ell}$ *almost surely converge empirically. The components of* $\mathbf{p}^{+\mathrm{ran}}_{k\ell}$ *converge as*

$$\lim_{N\to\infty} \{p^{+\mathrm{ran}}_{k\ell,n}\} \overset{PL(2)}{=} U_{k\ell}, \tag{A.36}$$

where $U_{k\ell}$ *is a zero mean Gaussian random variable independent of the limiting random variables corresponding to the variables in* $\overline{\mathfrak{G}}^{+}_{k\ell}$.

Proof. The proof is very similar to that of [127, Lemmas 7,8]. □

We are now ready to prove Lemma 9.

Proof of Lemma 9. Using the partition (B.23a) and Lemmas 10 and 11, we see that the components of the vector sequences in $\overline{\mathfrak{G}}^{+}_{k\ell}$ along with $\mathbf{p}^{+}_{k\ell}$ almost surely converge jointly empirically, where the components of $\mathbf{p}^{+}_{k\ell}$ have the limit

$$\lim_{N_\ell\to\infty} \left\{p^{+}_{k\ell,n}\right\} = \lim_{N_\ell\to\infty} \left\{p^{\mathrm{det}}_{k\ell,n} + p^{\mathrm{ran}}_{k\ell,n}\right\} \overset{PL(2)}{=} \beta^0_\ell P^0_\ell + \sum_{i=0}^{k-1} \beta^{+}_{i\ell} P^{+}_{i\ell} + U_{k\ell} =: P^{+}_{k\ell}. \tag{A.37}$$

Note that the above PL(2) convergence can be shown using the same arguments involved in showing that if $X_N|\mathcal{F} \overset{d}{\Longrightarrow} X|\mathcal{F}$, and $Y_N|\mathcal{F} \overset{d}{\Longrightarrow} c$, then $(X_N, Y_N)|\mathcal{F} \overset{d}{\Longrightarrow} (X, c)|\mathcal{F}$ for some constant c and sigma-algebra $\mathcal{F}$.

We first establish the Gaussianity of $P^{+}_{k\ell}$. Observe that by the induction hypothesis, $\mathcal{H}^{-}_{k,\ell+1}$ holds whereby $(P^0_\ell, P^{+}_{0\ell}, \ldots, P^{+}_{k-1,\ell}, Q^{-}_{0,\ell+1}, \ldots, Q^{-}_{k,\ell+1})$, is jointly Gaussian. Since U_k is Gaussian and independent of $(P^0_\ell, P^{+}_{0\ell}, \ldots, P^{+}_{k-1,\ell}, Q^{-}_{0,\ell+1}, \ldots, Q^{-}_{k,\ell+1})$, we can conclude from (B.34) that

$$(P^0_\ell, P^{+}_{0\ell}, \ldots, P^{+}_{k-1,\ell}, P^{+}_{k\ell}, Q^{-}_{0,\ell+1}, \ldots, Q^{-}_{k,\ell+1}) \text{ is jointly Gaussian.} \tag{A.38}$$

We now need to prove the correlations of this jointly Gaussian random vector as claimed by $\mathcal{H}^{+}_{k,\ell+1}$. Since $\mathcal{H}^{-}_{k,\ell+1}$ is true, we know that (B.15) is true for all $i = 0, \ldots, k-1$ and $j = 0, \ldots, k$ and $\ell = \ell + 1$. Hence, we need only to prove the additional identity for $i = k$, namely the equations: $\text{Cov}(P^0_\ell, P^+_{k\ell})^2 = \mathbf{K}^+_{k\ell}$ and $\mathbb{E}(P^+_{k\ell}Q^-_{j,\ell+1}) = 0$. First observe that

$$\mathbb{E}(P^+_{k\ell})^2 \overset{(a)}{=} \lim_{N_\ell\to\infty} \frac{1}{N_\ell}\|\mathbf{p}^+_{k\ell}\|^2 \overset{(b)}{=} \lim_{N_\ell\to\infty} \frac{1}{N_\ell}\|\mathbf{q}^+_{k\ell}\|^2 \overset{(c)}{=} \mathbb{E}\left(Q^+_{k\ell}\right)^2$$

where (a) follows from the fact that the components of $\mathbf{p}^+_{k\ell}$ converge empirically to $P^+_{k\ell}$; (b) follows from line 20 in Algorithm 10 and the fact that $\mathbf{V}_\ell$ is orthogonal; and (c) follows from the fact that the components of $\mathbf{q}^+_{k\ell}$ converge empirically to $Q^+_{k\ell}$ from hypothesis $\mathcal{H}^+_{k,\ell}$. Since $\mathbf{p}^0_\ell = \mathbf{V}_\ell\mathbf{q}^0$, we similarly obtain that $\mathbb{E}(P^0_\ell P^+_{k\ell}) = \mathbb{E}(Q^0_\ell Q^+_{k\ell}), \quad \mathbb{E}(P^0_\ell)^2 = \mathbb{E}(Q^0_\ell)^2$, from which we conclude

$$\text{Cov}(P^0_\ell, P^+_{k\ell}) = \text{Cov}(Q^0_\ell, Q^+_{k\ell}) =: \mathbf{K}^+_{k\ell}, \tag{A.39}$$

where the last step follows from the definition of $\mathbf{K}^+_{k\ell}$ in line 20 of Algorithm 11. Finally, we observe that for $0 \le j \le k$

$$\mathbb{E}(P^+_{k\ell}Q^-_{j,\ell+1}) \overset{(a)}{=} \beta^0_\ell\mathbb{E}(P^0_\ell Q^-_{j,\ell+1}) + \sum_{i=0}^{k-1}\beta^+_{i\ell}\mathbb{E}(P^+_{i\ell}Q^-_{j,\ell+1}) + \mathbb{E}(U_{k\ell}Q^-_{j,\ell+1}) \overset{(a)}{=} 0, \tag{A.40}$$

where (a) follows from (B.34) and, in (b), we used the fact that $\mathbb{E}(P^0_\ell Q^-_{j,\ell+1}) = 0$ and $\mathbb{E}(P^+_{i\ell}Q^-_{j,\ell+1}) = 0$ since (B.15) is true for $i \le k{-}1$ corresponding to $\mathcal{H}^-_{k,\ell+1}$ and $\mathbb{E}(U_{k\ell}Q^-_{j,\ell+1}) = 0$ since $U_{k\ell}$ is independent of $\overline{\mathfrak{G}}^+_{k\ell}$, and $Q^-_{j,\ell+1}$ is $\overline{\mathfrak{G}}^+_{k\ell}$ measurable. Thus, with (B.35) and (B.36), we have proven all the correlations in (B.15) corresponding to $\mathcal{H}^+_{k,\ell+1}$.

Next, we prove the convergence of the parameter lists $\Lambda^+_{k,\ell+1}$ to $\overline{\Lambda}^+_{k,\ell+1}$. Since $\Lambda^+_{k\ell} \to \overline{\Lambda}^+_{k\ell}$ due to hypothesis $\mathcal{H}^+_{k\ell}$, and $\varphi^+_{k,\ell+1}(\cdot)$ is uniformly Lipschitz continuous, we have that $\lim_{N\to\infty}\mu^+_{k,\ell+1}$ from line 17 in Algorithm 10 converges almost surely as

$$\lim_{N\to\infty}\left\langle \boldsymbol{\varphi}^+_{k,\ell+1}(\mathbf{p}^0_\ell, \mathbf{p}^+_{k\ell}, \mathbf{q}^-_{k,\ell+1}, \mathbf{w}_{\ell+1}, \overline{\Lambda}^+_{k\ell})\right\rangle = \mathbb{E}\left[\varphi^+_{k,\ell+1}(P^0_\ell, P^+_{k\ell}, Q^-_{k,\ell+1}, W_{\ell+1}, \overline{\Lambda}^+_{k\ell})\right] = \overline{\mu}^+_{k,\ell+1}, \tag{A.41}$$

where $\overline{\mu}^+_{k,\ell+1}$ is the value in line 17 in Algorithm 11. Since $T^+_{k,\ell+1}(\cdot)$ is continuous, we have that $\lambda^+_{k,\ell+1}$ in line 18 in Algorithm 10 converges as $\lim_{N\to\infty}\lambda^+_{k,\ell+1} = T^+_{k,\ell+1}(\overline{\mu}^+_{k,\ell+1}, \overline{\Lambda}^+_{k\ell}) =: \overline{\lambda}^+_{k,\ell+1}$, from line 18 in Algorithm 11. Therefore, we have the limit

$$\lim_{N\to\infty}\Lambda^+_{k,\ell+1} = \lim_{N\to\infty}(\Lambda^+_{k,\ell}, \lambda^+_{k,\ell+1}) = (\overline{\Lambda}^+_{k,\ell}, \overline{\lambda}^+_{k,\ell+1}) = \overline{\Lambda}^+_{k,\ell+1}, \tag{A.42}$$

which proves the convergence of the parameter lists stated in $\mathcal{H}^+_{k,\ell+1}$. Finally, using (B.38), the empirical convergence of the vector sequences $\mathbf{p}_\ell^0$, $\mathbf{p}_{k\ell}^+$ and $\mathbf{q}_{k,\ell+1}^-$ and the uniform Lipschitz continuity of the update function $f_{k,\ell+1}^+(\cdot)$ we obtain that $\lim_{N\to\infty}\left\{q_{k,\ell+1,n}^+\right\}$ equals

$$\left\{f_{k,\ell+1}^+(p_{\ell,n}^0, p_{k\ell,n}^-, q_{k,\ell+1,n}^-, w_{\ell+1,n}, \Lambda_{k,\ell+1}^+)\right\} = f_{k,\ell+1}^+(P_\ell^0, P_{k\ell}^-, Q_{k,\ell+1}^-, W_{\ell+1}, \overline{\Lambda}_{k,\ell+1}^+) =: Q_{k,\ell+1}^+,$$

which proves the claim (B.16) for $\mathcal{H}_{k,\ell+1}^+$. This completes the proof. $\square$

Algorithm 7 General Multi-Layer (Gen-ML) Recursion

```
Require: Initial vector functions $\mathbf{f}^0_\ell$, vector update functions $\mathbf{f}^\pm_{k\ell}(\cdot)$, parameter statistic functions $\boldsymbol{\varphi}^\pm_{k\ell}(\cdot)$, parameter update functions $T^\pm_{k\ell}(\cdot)$, orthogonal matrices $\mathbf{V}_\ell$, disturbance vectors $\mathbf{w}^\pm_\ell$.
1: // Initialization
2: Initialize parameter list $\Lambda^-_{01}$ and vectors $\mathbf{p}^0_0$ and $\mathbf{q}^-_{0\ell}$ for $\ell = 0,\ldots,L-1$
3: $\mathbf{q}^0_0 = \mathbf{f}^0_0(\mathbf{w}_0), \quad \mathbf{p}^0_0 = \mathbf{V}_0\mathbf{q}^0_0$
4: for $\ell = 1,\ldots,L-1$ do
5:     $\mathbf{q}^0_\ell = \mathbf{f}^0_\ell(\mathbf{p}^0_{\ell-1}, \mathbf{w}_\ell, \Lambda^-_{01})$
6:     $\mathbf{p}^0_\ell = \mathbf{V}_\ell\mathbf{q}^0_\ell$
7: end for
8:
9: for $k = 0, 1, \ldots$ do
10:     // Forward Pass
11:     $\lambda^+_{k0} = T^+_{k0}(\mu^+_{k0}, \Lambda^-_{0k}), \quad \mu^+_{k0} = \big\langle \boldsymbol{\varphi}^+_{k0}(\mathbf{q}^-_{k0}, \mathbf{w}_0, \Lambda^-_{0k})\big\rangle$
12:     $\Lambda^+_{k0} = (\Lambda^-_{k1}, \lambda^+_{k0})$
13:     $\mathbf{q}^+_{k0} = \mathbf{f}^+_{k0}(\mathbf{q}^-_{k0}, \mathbf{w}_0, \Lambda^+_{k0})$
14:     $\mathbf{p}^+_{k0} = \mathbf{V}_0\mathbf{q}^+_{k0}$
15:     for $\ell = 1,\ldots,L-1$ do
16:         $\lambda^+_{k\ell} = T^+_{k\ell}(\mu^+_{k\ell}, \Lambda^+_{k,\ell-1}), \quad \mu^+_{k\ell} = \big\langle \boldsymbol{\varphi}^+_{k\ell}(\mathbf{p}^0_{\ell-1}, \mathbf{p}^+_{k,\ell-1}, \mathbf{q}^-_{k\ell}, \mathbf{w}_\ell, \Lambda^+_{k,\ell-1})\big\rangle$
17:         $\Lambda^+_{k\ell} = (\Lambda^+_{k,\ell-1}, \lambda^+_{k\ell})$
18:         $\mathbf{q}^+_{k\ell} = \mathbf{f}^+_{k\ell}(\mathbf{p}^0_{\ell-1}, \mathbf{p}^+_{k,\ell-1}, \mathbf{q}^-_{k\ell}, \mathbf{w}_\ell, \Lambda^+_{k\ell})$
19:         $\mathbf{p}^+_{k\ell} = \mathbf{V}_\ell\mathbf{q}^+_{k\ell}$
20:     end for
21:
22:     // Backward Pass
23:     $\lambda^-_{k+1,L} = T^-_{kL}(\mu^-_{kL}, \Lambda^+_{k,L-1}), \quad \mu^-_{kL} = \big\langle \boldsymbol{\varphi}^-_{kL}(\mathbf{p}^+_{k,L-1}, \mathbf{w}_L, \Lambda^+_{k,L-1})\big\rangle$
24:     $\Lambda^-_{k+1,L} = (\Lambda^+_{k,L-1}, \lambda^+_{k+1,L})$
25:     $\mathbf{p}^-_{k+1,L-1} = \mathbf{f}^-_{kL}(\mathbf{p}^0_{L-1}, \mathbf{p}^+_{k,L-1}, \mathbf{w}_L, \Lambda^-_{k+1,L})$
26:     $\mathbf{q}^-_{k+1,L-1} = \mathbf{V}^{\mathsf{T}}_{L-1}\mathbf{p}_{k+1,L-1}$
27:     for $\ell = L-1,\ldots,1$ do
28:         $\lambda^-_{k+1,\ell} = T^-_{k\ell}(\mu^-_{k\ell}, \Lambda^-_{k+1,\ell+1}), \quad \mu^-_{k\ell} = \big\langle \boldsymbol{\varphi}^-_{k\ell}(\mathbf{p}^0_{\ell-1}, \mathbf{p}^+_{k,\ell-1}, \mathbf{q}^-_{k+1,\ell}, \mathbf{w}_\ell, \Lambda^-_{k+1,\ell+1})\big\rangle$
29:         $\Lambda^-_{k+1,\ell} = (\Lambda^-_{k+1,\ell+1}, \lambda^-_{k+1,\ell})$
30:         $\mathbf{p}^-_{k+1,\ell-1} = \mathbf{f}^-_{k\ell}(\mathbf{p}^0_{\ell-1}, \mathbf{p}^+_{k,\ell-1}, \mathbf{q}^-_{k+1,\ell}, \mathbf{w}_\ell, \Lambda^-_{k+1,\ell})$
31:         $\mathbf{q}^-_{k+1,\ell-1} = \mathbf{V}^{\mathsf{T}}_{\ell-1}\mathbf{p}^-_{k+1,\ell-1}$
32:     end for
33: end for
```

Algorithm 8 Gen-ML State Evolution (SE)

```
Require: Vector update component functions $f^0_\ell(\cdot)$ and $f^\pm_{k\ell}(\cdot)$, parameter statistic component functions $\varphi^\pm_{k\ell}(\cdot)$, parameter update functions $T^\pm_{k\ell}(\cdot)$, initial parameter list limit: $\overline{\Lambda}^-_{01}$, initial random variables $W_\ell$, $Q^-_{0\ell}$, $\ell = 0,\ldots,L-1$.
1: // Initial pass
2: $Q^0_0 = f^0_0(W_0, \overline{\Lambda}^-_{01}), \quad P^0_0 \sim \mathcal{N}(0,\tau^0_0), \quad \tau^0_0 = \mathbb{E}(Q^0_0)^2$
3: for $\ell = 1,\ldots,L-1$ do
4:     $Q^0_\ell = f^0_\ell(P^0_{\ell-1}, W_\ell, \overline{\Lambda}^-_{01}), \quad P^0_\ell \sim \mathcal{N}(0,\tau^0_\ell), \quad \tau^0_\ell = \mathbb{E}(Q^0_\ell)^2$
5: end for
6:
7: for $k = 0,1,\ldots$ do
8:     // Forward Pass
9:     $\overline{\lambda}^+_{k0} = T^+_{k0}(\overline{\mu}^+_{k0}, \overline{\Lambda}^-_{0k}), \quad \overline{\mu}^+_{k0} = \mathbb{E}(\varphi^+_{k0}(Q^-_{k0}, W_0, \overline{\Lambda}^-_{0k}))$
10:    $\overline{\Lambda}^+_{k0} = (\overline{\Lambda}^-_{k1}, \overline{\lambda}^+_{k0})$
11:    $Q^+_{k0} = f^+_{k0}(Q^-_{k0}, W_0, \overline{\Lambda}^+_{k0})$
12:    $(P^0_0, P^+_{k0}) \sim \mathcal{N}(\mathbf{0}, \mathbf{K}^+_{k0}), \quad \mathbf{K}^+_{k0} = \mathrm{Cov}(Q^0_0, Q^+_{k0})$
13:    for $\ell = 1,\ldots,L-1$ do
14:        $\overline{\lambda}^+_{k\ell} = T^+_{k\ell}(\overline{\mu}^+_{k\ell}, \overline{\Lambda}^+_{k,\ell-1}), \quad \overline{\mu}^+_{k\ell} = \mathbb{E}(\varphi^+_{k\ell}(P^0_{\ell-1}, P^+_{k,\ell-1}, Q^-_{k\ell}, W_\ell, \overline{\Lambda}^+_{k,\ell-1}))$
15:        $\overline{\Lambda}^+_{k\ell} = (\overline{\Lambda}^+_{k,\ell-1}, \overline{\lambda}^+_{k\ell})$
16:        $Q^+_{k\ell} = f^+_{k\ell}(P^0_{\ell-1}, P^+_{k,\ell-1}, Q^-_{k\ell}, W_\ell, \overline{\Lambda}^+_{k\ell})$
17:        $(P^0_\ell, P^+_{k\ell}) \sim \mathcal{N}(\mathbf{0}, \mathbf{K}^+_{k\ell}), \quad \mathbf{K}^+_{k\ell} = \mathrm{Cov}(Q^0_\ell, Q^+_{k\ell})$
18:    end for
19:
20:    // Backward Pass
21:    $\overline{\lambda}^-_{k+1,L} = T^-_{kL}(\overline{\mu}^-_{kL}, \overline{\Lambda}^+_{k,L-1}), \quad \overline{\mu}^-_{kL} = \mathbb{E}(\varphi^-_{kL}(P^0_{L-1}, P^+_{k,L-1}, W_L, \overline{\Lambda}^+_{k,L-1}))$
22:    $\overline{\Lambda}^-_{k+1,L} = (\overline{\Lambda}^+_{k,L-1}, \overline{\lambda}^+_{k+1,L})$
23:    $P^-_{k+1,L-1} = f^-_{kL}(P^0_{L-1}, P^+_{k,L-1}, W_L, \overline{\Lambda}^-_{k+1,L})$
24:    $Q^-_{k+1,L-1} \sim \mathcal{N}(0, \tau^-_{k+1,L-1}), \quad \tau^-_{k+1,L-1} = \mathbb{E}(P^-_{k+1,L-1})^2$
25:    for $\ell = L-1,\ldots,1$ do
26:        $\overline{\lambda}^-_{k+1,\ell} = T^-_{k\ell}(\overline{\mu}^-_{k\ell}, \overline{\Lambda}^-_{k+1,\ell+1}), \quad \overline{\mu}^-_{k\ell} = \mathbb{E}(\varphi^-_{k\ell}(P^0_{\ell-1}, P^+_{k,\ell-1}, Q^-_{k+1,\ell}, W_\ell, \overline{\Lambda}^-_{k+1,\ell+1}))$
27:        $\overline{\Lambda}^-_{k+1,\ell} = (\overline{\Lambda}^-_{k+1,\ell+1}, \overline{\lambda}^-_{k+1,\ell})$
28:        $P^-_{k+1,\ell-1} = f^-_{k\ell}(P^0_{\ell-1}, P^+_{k,\ell-1}, Q^-_{k+1,\ell}, W_\ell, \overline{\Lambda}^-_{k+1,\ell})$
29:        $Q^-_{k+1,\ell-1} \sim \mathcal{N}(0, \tau^-_{k+1,\ell-1}), \quad \tau^-_{k+1,\ell-1} = \mathbb{E}(P^-_{k+1,\ell-1})^2$
30:    end for
31: end for
```

Appendix B

Proofs from Chapter 4

B.1 State Evolution Equations

The state evolution equations given in Algo. 9 define an iteration indexed by k of constant matrices $\{\mathbf{K}^+_{k\ell}, \boldsymbol{\tau}^-_{kl}, \overline{\mathbf{\Gamma}}^{\pm}_{kl}\}_{\ell=0}^{L}$. These constants appear in the statement of the main result in Theorem 9. The iterations in Algo. 9 also iteratively define a few $\mathbb{R}^{1\times d}$ valued random vectors $\{Q^0_\ell, P^0_\ell, Q^\pm_{k\ell}, P^\pm_{k\ell}\}$ which are either multivariate Gaussian or functions of Multivariate Gaussians. In order to state Algorithm 9, we need to define certain random variables and functions appearing therein which are described below. Let $L_{\text{odd}} = \{1, 3, \ldots, L-1\}$ and $L_{\text{even}} = \{2, 4, \ldots, L-2\}$.

Define $\{\overline{\mathbf{\Theta}}^{\pm}_{k\ell}\}$ similar to $\mathbf{\Theta}^{\pm}_{k\ell}$ from equation (4.14) using $\{\overline{\mathbf{\Gamma}}^{\pm}_{k\ell}\}$. Further, for $\ell = 1, 2, \ldots, L-1$ define

$$\overline{\mathbf{\Omega}}^{+}_{k\ell} := (\overline{\mathbf{\Lambda}}^{+}_{k\ell}, \overline{\mathbf{\Gamma}}^{+}_{k\ell}, \overline{\mathbf{\Gamma}}^{-}_{k\ell}),\ \overline{\mathbf{\Omega}}^{-}_{k\ell} := (\overline{\mathbf{\Lambda}}^{+}_{k,\ell-1}, \overline{\mathbf{\Gamma}}^{-}_{k,\ell-1}, \overline{\mathbf{\Gamma}}^{-}_{k,\ell-1}),$$

and $\overline{\mathbf{\Omega}}^{+}_{k0}$ and $\overline{\mathbf{\Omega}}^{-}_{kL}$. Now define random variables W_ℓ as

$$\begin{aligned} &W_0 = Z^0_0, \quad W_L = (Y, \Xi_L), \quad W_\ell = \Xi_\ell, \quad \forall\, \ell \in L_{\text{even}}, \\ &W_\ell = (S_\ell, \overline{B}_\ell, \Xi_\ell), \qquad\qquad\qquad\quad\ \forall\, \ell \in L_{\text{odd}}. \end{aligned} \tag{B.1}$$

Algorithm 9 State Evolution for ML-Mat-VAMP (Algo. 4)

Require: Functions $\{f^0_\ell\}$ from (B.2), $\{h^\pm_\ell\}$ from (B.3), and $\{f^\pm_\ell\}$ from (B.4). Perturbation random variables $\{W_\ell\}$ from (B.1). Initial random vectors $\{Q^-_{0\ell}\}_{\ell=0}^{L-1}$ with Initial covariance matrices $\{\boldsymbol{\tau}^-_{0\ell}\}_{\ell=0}^{L-1}$ from Section 4. Initial matrices $\{\overline{\boldsymbol{\Gamma}}^-_{0\ell}\}_{\ell=0}^{L}$ from (4.16).

```
1:  // Initial Pass
2:  $Q^0_0 = W_0$, $\boldsymbol{\tau}^0_0 = \mathrm{Cov}(Q^0_0)$ and $P^0_0 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\tau}^0_0)$
3:  for $\ell = 1, \ldots, L-1$ do
4:      $Q^0_\ell = f^0_\ell(P^0_{\ell-1}, W_\ell)$
5:      $P^0_\ell \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\tau}^0_\ell)$,        $\boldsymbol{\tau}^0_\ell = \mathrm{Cov}(Q^0_\ell)$
6:  end for

7:  for $k = 0, 1, \ldots$ do
8:      // Forward Pass
9:      $\widehat{Q}^+_{k0} = h^+_0(Q^-_{k0}, W_0, \overline{\boldsymbol{\Theta}}^+_{k0})$
10:     $\overline{\boldsymbol{\Lambda}}^+_{k0} = (\mathbb{E}\frac{\partial \widehat{Q}^+_{k0}}{\partial Q^-_0})^{-1}\overline{\boldsymbol{\Gamma}}^-_{k,0}$
11:     $\overline{\boldsymbol{\Gamma}}^+_{k0} = \overline{\boldsymbol{\Lambda}}^+_{k0} - \overline{\boldsymbol{\Gamma}}^-_{k0}$
12:     $Q^+_{k0} = f^+_0(Q^-_{k0}, W_0, \overline{\boldsymbol{\Omega}}^+_{k0})$
13:     $(P^0_0, P^+_{k0}) \sim \mathcal{N}(\mathbf{0}, \mathbf{K}^+_{k0})$,        $\mathbf{K}^+_{k0} := \mathrm{Cov}(Q^0_0, Q^+_{k0})$

14:     for $\ell = 1, \ldots, L-1$ do
15:         $\widehat{Q}^+_{k\ell} = h^+_\ell(P^0_{\ell-1}, P^+_{k,\ell-1}, Q^-_{k\ell}, W_\ell, \overline{\boldsymbol{\Theta}}^+_{k\ell})$
16:         $\overline{\boldsymbol{\Lambda}}^+_{k\ell} = (\mathbb{E}\frac{\partial \widehat{Q}^+_{k\ell}}{\partial Q^-_{k\ell}})^{-1}\overline{\boldsymbol{\Gamma}}^-_{k\ell}$
17:         $\overline{\boldsymbol{\Gamma}}^+_{k\ell} = \overline{\boldsymbol{\Lambda}}^+_{k\ell} - \overline{\boldsymbol{\Gamma}}^-_{k\ell}$
18:         $Q^+_{k\ell} = f^+_\ell(P^0_{\ell-1}, P^+_{k,\ell-1}, Q^-_{k\ell}, W_\ell, \overline{\boldsymbol{\Omega}}^+_{k\ell})$
19:         $(P^0_\ell, P^+_{k\ell}) \sim \mathcal{N}(\mathbf{0}, \mathbf{K}^+_{k\ell})$,     $\mathbf{K}^+_{k\ell} := \mathrm{Cov}(Q^0_\ell, Q^+_{k\ell})$
20:     end for

21:     // Backward Pass
22:     $\widehat{P}^-_{k+1,L-1} = h^-_L(P^0_{L-1}, P^+_{k,L-1}, W_L, \overline{\boldsymbol{\Theta}}^-_{k+1,L})$
23:     $\overline{\boldsymbol{\Lambda}}^-_{k+1,L} = (\mathbb{E}\frac{\partial \widehat{P}^-_{k+1,L-1}}{\partial P^+_{L-1}})^{-1}\overline{\boldsymbol{\Gamma}}^+_{kL}$
24:     $\overline{\boldsymbol{\Gamma}}^-_{k+1,L-1} = \overline{\boldsymbol{\Lambda}}^-_{k+1,L-1} - \overline{\boldsymbol{\Gamma}}^+_{k,L-1}$,
25:     $P^-_{k+1,L-1} = f^-_L(P^0_{L-1}, P^+_{k,L-1}, W_L, \overline{\boldsymbol{\Omega}}^-_{k+1,L})$
26:     $Q^-_{k+1,L-1} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\tau}^-_{k+1,L-1})$,  $\boldsymbol{\tau}^-_{k+1,L-1} := \mathrm{Cov}(P^-_{k+1,L-1})$
27:     for $\ell = L-2, \ldots, 0$ do
28:         $\widehat{P}^-_{k+1,\ell} = h^-_\ell(P^0_\ell, P^+_{k\ell}, Q^-_{k+1,\ell+1}, W_\ell, \overline{\boldsymbol{\Theta}}^-_{k+1,\ell})$
29:         $\overline{\boldsymbol{\Lambda}}^-_{k+1,\ell} = (\mathbb{E}\frac{\partial \widehat{P}^-_{k+1,\ell}}{\partial P^+_{k,\ell}})^{-1}\overline{\boldsymbol{\Gamma}}^+_{k,\ell}$
30:         $\overline{\boldsymbol{\Gamma}}^-_{k+1,\ell} = \overline{\boldsymbol{\Lambda}}^-_{k+1,\ell} - \overline{\boldsymbol{\Gamma}}^+_{k,\ell}$,
31:         $P^-_{k+1,\ell} = f^-_\ell(P^0_\ell, P^+_{k\ell}, Q^-_{k+1,\ell+1}, W_\ell, \overline{\boldsymbol{\Omega}}^-_{k+1,\ell})$
32:         $Q^-_{k+1,\ell} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\tau}^-_{k+1,\ell})$,     $\boldsymbol{\tau}^-_{k+1,\ell} := \mathrm{Cov}(P^-_{k+1,\ell})$
33:     end for
34: end for
```

Define functions $\{f_\ell^0\}_{\ell=1}^L$ as

$$\begin{aligned} f_\ell^0(P_{\ell-1}^0, W_\ell) &:= S_\ell P_{\ell-1}^0 + \overline{B}_\ell + \Xi_\ell, \quad \forall\, \ell \in L_{\text{odd}}, \\ f_\ell^0(P_{\ell-1}^0, W_\ell) &:= \phi_\ell(P_{\ell-1}^0, \Xi_\ell), \quad \forall\, \ell \in L_{\text{even}} \cup \{L\}. \end{aligned} \tag{B.2}$$

and using (4.14) define functions $\{h_\ell^\pm,\}_{\ell=1}^L$, h_0^+ and h_L^- as

$$\begin{aligned} &h_\ell^\pm(P_{\ell-1}^0, P_{\ell-1}^+, Q_\ell^-, W_\ell, \mathbf{\Theta}_{k\ell}^\pm) = G_\ell^\pm(Q_\ell^- + Q_\ell^0, P_{\ell-1}^+ + P_{\ell-1}^0, \mathbf{\Theta}_{k\ell}^\pm), \quad \forall\, \ell \in L_{\text{even}}, \\ &h_\ell^\pm(P_{\ell-1}^0, P_{\ell-1}^+, Q_\ell^-, W_\ell, \mathbf{\Theta}_{k\ell}^\pm) = \widetilde{G}_\ell^\pm(Q_\ell^- + Q_\ell^0, P_{\ell-1}^+ + P_{\ell-1}^0, \mathbf{\Theta}_{k\ell}^\pm), \quad \forall\, \ell \in L_{\text{odd}} \\ &h_0^+(Q_0^-, W_0, \mathbf{\Theta}_{k0}^+) = G_0^+(Q_0^- + W_0, \mathbf{\Theta}_{k0}^+), \\ &h_L^-(P_{L-1}^0, P_{L-1}^+, W_L, \mathbf{\Theta}_{kL}^-) = G_L^-(P_{L-1}^+ + P_{L-1}^0, \mathbf{\Theta}_{kL}^-). \end{aligned} \tag{B.3}$$

Note that $[G_\ell^+, G_\ell^-]$ and $[\widetilde{G}_\ell^+, \widetilde{G}_\ell^-]$ are maps from $\mathbb{R}^{1\times d} \to \mathbb{R}^{1\times d}$ such that their row-wise extensions are the denoisers $[\mathbf{G}_\ell^+, \mathbf{G}_\ell^-]$ and $[\widetilde{\mathbf{G}}_\ell^+, \widetilde{\mathbf{G}}_\ell^-]$ respectively. Using (B.3) define functions $\{f_\ell^\pm\}_{\ell=1}^{L-1}$, f_0^+ and f_L^- as

$$\begin{aligned} f_\ell^+(P_{\ell-1}^0, P_{\ell-1}^+, Q_\ell^-, W_\ell, \mathbf{\Omega}_{k\ell}^+) &= \left[\left(h_\ell^+ - Q_\ell^0\right)\mathbf{\Lambda}_{k\ell}^+ - Q_\ell^-\mathbf{\Gamma}_{k\ell}^-\right](\mathbf{\Gamma}_{k\ell}^+)^{-1}, \\ f_\ell^-(P_{\ell-1}^0, P_{\ell-1}^+, Q_\ell^-, W_\ell, \mathbf{\Omega}_{k\ell}^-) &= \left[\left(h_\ell^- - P_{\ell-1}^0\right)\mathbf{\Lambda}_{k,\ell-1}^- - P_{\ell-1}^+\mathbf{\Gamma}_{k,\ell-1}^+\right](\mathbf{\Gamma}_{k,\ell-1}^-)^{-1}. \\ f_0^+(Q_0^-, W_0, \mathbf{\Omega}_{k0}^+) &= \left[\left(h_0^+ - W_0\right)\mathbf{\Lambda}_{k0}^+ - Q_0^-\mathbf{\Gamma}_{k0}^-\right](\mathbf{\Gamma}_{k0}^+)^{-1}, \\ f_L^-(P_{L-1}^0, P_{L-1}^+, W_L, \mathbf{\Omega}_{kL}^-) &= \left[\left(h_L^- - P_{L-1}^0\right)\mathbf{\Lambda}_{k,L-1}^- - P_{L-1}^+\mathbf{\Gamma}_{k,L-1}^+\right](\mathbf{\Gamma}_{k,L-1}^-)^{-1}. \end{aligned} \tag{B.4}$$

B.2 Large System Limit Details

The analysis of Algorithm 4 in the large system limit is based on [10] and is by now standard in the theory of AMP-based algorithms. The goal is to characterize ensemble row-wise averages of iterates of the algorithm using *simpler* finite-dimensional random variables which are either Gaussians or functions of Gaussians. To that end, we start by defining some key terms needed in this analysis.

Definition 5 (Pseudo-Lipschitz continuity)**.** For a given $p \geq 1$, a map $\mathbf{g} : \mathbb{R}^{1\times d} \to \mathbb{R}^{1\times r}$ is called pseudo-Lipschitz of order p if for any $\mathbf{r}_1, \mathbf{r}_2 \in \mathbb{R}^d$ we have,

$$\|\mathbf{g}(\mathbf{r}_1) - \mathbf{g}(\mathbf{r}_2)\| \leq C\|\mathbf{r}_1 - \mathbf{r}_2\|\left(1 + \|\mathbf{r}_1\|^{p-1} + \|\mathbf{r}_2\|^{p-1}\right)$$

Definition 6 (Empirical convergence of rows of a matrix sequence)**.** Consider a matrix-sequence $\{\mathbf{X}^{(N)}\}_{N=1}^{\infty}$ with $\mathbf{X}^{(N)} \in \mathbb{R}^{N\times d}$. For a finite $p \geq 1$, let $X \in (\mathbb{R}^d, \mathcal{R}^d)$ be a $\mathcal{R}^d$-measurable random variable with bounded moment $\mathbb{E}\|X\|_p^p < \infty$. We say the rows of matrix sequence $\{\mathbf{X}^{(N)}\}$ *converge empirically to X with p^{th} order moments* if for all pseudo-Lipschitz continuous functions $f(\cdot)$ of order p,

$$\lim_{N\to\infty} \frac{1}{N}\sum_{n=1}^{N} f(\mathbf{X}_{n:}^{(N)}) = \mathbb{E}[f(X)] \quad \text{a.s.} \tag{B.5}$$

Note that the sequence $\{\mathbf{X}^{(N)}\}$ could be random or deterministic. If it is random, however, then the quantities on the left hand side are random sums and the almost sure convergence must take this randomness into account as well.

The above convergence is equivalent to requiring weak convergence as well as convergence of the p^{th} moment, of the empirical distribution $\frac{1}{N}\sum_{n=1}^{N}\delta_{\mathbf{X}_{n:}^{(N)}}$ of the rows of $\mathbf{X}^{(N)}$ to X. This is also referred to convergence in the Wasserstein-p metric [157, Chap. 6].

In the case of $p = 2$, the condition is equivalent to requiring (B.5) to hold for all continuously bounded functions f as well as for all $f_q(\boldsymbol{x}) = \boldsymbol{x}^{\mathsf{T}}\boldsymbol{Q}\boldsymbol{x}$ for all positive definite matrices $\boldsymbol{Q}$.

Definition 7 (Uniform Lipschitz continuity)**.** For a positive definite matrix $\boldsymbol{M}$, the map $\phi(\mathbf{r};\boldsymbol{M}) : \mathbb{R}^d \to \mathbb{R}^d$ is said to be uniformly Lipschitz continuous in $\mathbf{r}$ at $\boldsymbol{M} = \overline{\boldsymbol{M}}$ if there exist non-negative constants L_1, L_2 and L_3 such that for all $\mathbf{r} \in \mathbb{R}^d$

$$\|\phi(\mathbf{r}_1;\boldsymbol{M}_0) - \phi(\mathbf{r}_2;\boldsymbol{M}_0)\| \leq L_1\|\mathbf{r}_1 - \mathbf{r}_2\|$$

$$\|\phi(\mathbf{r};\boldsymbol{M}_1) - \phi(\mathbf{r};\boldsymbol{M}_2)\| \leq L_2(1 + \|\mathbf{r}\|)\rho(\boldsymbol{M}_1,\boldsymbol{M}_2)$$

for all $\boldsymbol{M}_i$ such that $\rho(\boldsymbol{M}_i,\overline{\boldsymbol{M}}) < L_3$ where ρ is a metric on the cone of positive semidefinite matrices.

We are now ready to prove Theorem 9.

B.3 Proof of Theorem 9

The proof of Theorem 9 is a special case of a more general result on multi-layer recursions given in Theorem 12. This result is stated in B.4, and proved in B.5. The rest of this section identifies certain relevant quantities from Theorem 9 in order to apply Theorem 12.

Consider the SVD given of weight matrices $\mathbf{W}_\ell$ of the network given by,

$$\mathbf{W}_\ell = \mathbf{V}_\ell \text{diag}(\mathbf{S}_\ell)\mathbf{V}_\ell - 1$$

as explained in Section 4 of the main paper. We analyze Algo. 4 using *transformed* versions of the true signals $\mathbf{Z}_\ell^0$ and input errors $\mathbf{R}_\ell^\pm - \mathbf{Z}_\ell^0$ to the denoisers $\mathbf{G}_\ell^\pm$. For $\ell = 0, 2, \ldots L-2$, define

$$\mathbf{q}_\ell^0 = \mathbf{Z}_\ell^0 \qquad \mathbf{q}_{\ell+1}^0 = \mathbf{V}_{\ell+1}^\top \mathbf{Z}_{\ell+1}^0 \tag{B.6a}$$

$$\mathbf{p}_\ell^0 = \mathbf{V}_\ell \mathbf{Z}_\ell^0 \qquad \mathbf{p}_{\ell+1}^0 = \mathbf{Z}_{\ell+1}^0 \tag{B.6b}$$

which are depicted in Fig. B.1 (TOP). Similarly, define the following *transformed* versions of errors in the inputs $\mathbf{R}_\ell^\pm$ to the denoisers $\mathbf{G}_\ell^\pm$

$$\mathbf{q}_\ell^- = \mathbf{R}_\ell^- - \mathbf{Z}_\ell^0 \qquad \mathbf{q}_{\ell+1}^- = \mathbf{V}_{\ell+1}^\top(\mathbf{R}_{\ell+1}^- - \mathbf{Z}_{\ell+1}^0) \tag{B.7a}$$

$$\mathbf{p}_\ell^+ = \boldsymbol{V}_\ell(\mathbf{R}_\ell^+ - \mathbf{Z}_\ell^0) \qquad \mathbf{p}_{\ell+1}^+ = \mathbf{R}_{\ell+1}^+ - \mathbf{Z}_{\ell+1}^0 \tag{B.7b}$$

These quantities are depicted as inputs to function blocks $\mathbf{f}_\ell^\pm$ in Fig. B.1 (MIDDLE). Define perturbation variables $\mathbf{w}_\ell$ as

$$\mathbf{w}_0 = \mathbf{Z}_0^0, \quad \mathbf{w}_L = (\mathbf{Y}, \Xi_L), \quad \mathbf{w}_\ell = \Xi_\ell, \qquad \forall\, \ell \in L_{\text{even}} \tag{B.8a}$$

$$\mathbf{w}_\ell = (\mathbf{S}_\ell, \overline{\mathbf{B}}_\ell, \Xi_\ell), \qquad \forall\, \ell \in L_{\text{odd}} \tag{B.8b}$$

Finally, we define $\mathbf{q}_\ell^+$ and $\mathbf{p}_\ell^-$ for $\ell = 1, 2, \ldots, L-1$ as

$$\mathbf{q}_\ell^+ = \mathbf{f}_\ell^+(\mathbf{p}_{\ell-1}^0, \mathbf{p}_{\ell-1}^+, \mathbf{q}_\ell^-, \mathbf{w}_\ell, \Omega_\ell) \tag{B.9a}$$

$$\mathbf{p}_{\ell-1}^- = \mathbf{f}_\ell^-(\mathbf{p}_{\ell-1}^0, \mathbf{p}_{\ell-1}^+, \mathbf{q}_\ell^-, \mathbf{w}_\ell, \Omega_\ell), \tag{B.9b}$$

which are outputs of function blocks in Fig. B.1 (MIDDLE). Similarly, define the quantities $\mathbf{q}_0^+ = \mathbf{f}_0^+(\mathbf{q}_0^-, \mathbf{Z}_0, \Omega_0)$ and $\mathbf{p}_{L-1}^- = \mathbf{f}_L^+(\mathbf{p}_{L-1}^0, \mathbf{p}_{L-1}^+, \mathbf{Y}, \Omega_L)$.

Lemma 7. *Algorithm 4 is a special case of Algorithm 10 with the definitions* $\{\mathbf{q}^0_\ell, \mathbf{p}^0_\ell, \mathbf{q}^\pm_\ell, \mathbf{p}^\pm_\ell\}_{\ell=0}^{L-1}$ *given in equations* (B.6),(B.7)*, and* (B.9)*, functions* $\mathbf{f}^\pm_\ell$ *are row-wise extensions of* $f^\pm_\ell$ *defined using equations* (B.4) *and* (B.3)*.*

Lemma 8. *Assumptions 5 and 6 required for applying Theorem 12 are satisfied by the conditions in Theorem 9.*

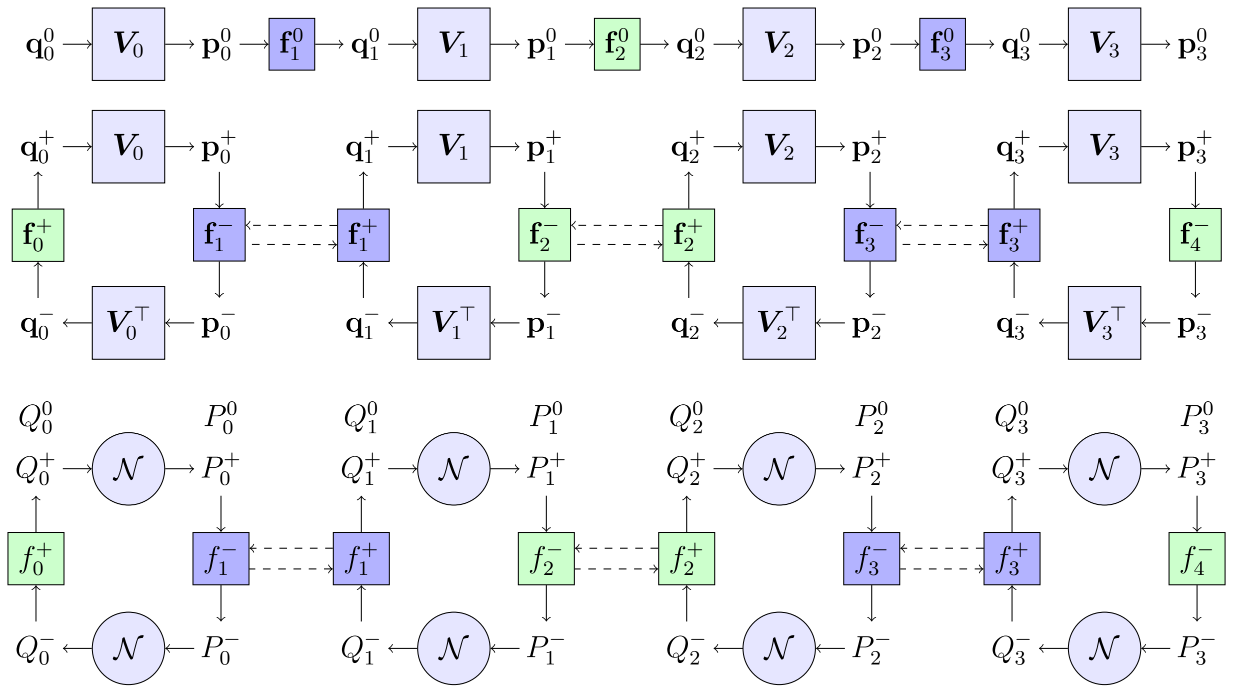
Proof. The proofs of the above lemmas are identical to the case of $d = 1$, which was shown in [114]. For details see [114, Appendix F]. $\square$

B.4 General Multi-Layer Recursions

To analyze Algorithm 4, we consider a more general class of recursions as given in Algorithm 10 and depicted in Fig. B.1. The Gen-ML recursions generates (i) a set of *true matrices* $\boldsymbol{q}^0_\ell$ and $\boldsymbol{p}^0_\ell$ and (ii) *iterated matrices* $\mathbf{q}^\pm_{k\ell}$ and $\mathbf{p}^\pm_{k\ell}$. Each of these matrices have the same number of columns, denoted by d.

The true matrices are generated by a single forward pass, whereas the iterated matrices are generated via a sequence of forward and backward passes through a multi-layer system. In proving the State Evolution for the ML-Mat-VAMP algorithm (Algo. 4, one would then associate the terms $\mathbf{q}^\pm_{k\ell}$ and $\mathbf{p}^\pm_{k\ell}$ with certain error quantities in the ML-Mat-VAMP recursions. To account for the effect of the parameters $\mathbf{\Gamma}^\pm_{k\ell}$ and $\mathbf{\Lambda}^\pm_{k\ell}$ in ML-Mat-VAMP, the Gen-ML algorithm describes the parameter updates through a sequence of *parameter lists* $\Upsilon^\pm_{k\ell}$. The parameter lists are ordered lists of parameters that accumulate as the algorithm progresses. The true and iterated matrices from Algorithm 10 are depicted in the signal flow graphs on the (TOP) and (MIDDLE) panel of Fig. B.1 respectively. The iteration index k for the iterated vectors $\boldsymbol{q}_{k\ell}, \boldsymbol{p}_{k\ell}$ has been dropped for simplifying notation.

The functions $\mathbf{f}^0_\ell(\cdot)$ that produce the true matrices $\boldsymbol{q}^0_\ell, \boldsymbol{p}^0_\ell$ are called *initial matrix functions* and use the initial parameter list Υ^-_{01}. The functions $\mathbf{f}^\pm_{k\ell}(\cdot)$ that produce the matrices $\mathbf{q}^+_{k\ell}$ and



$\mathbf{q}_0^0$ $\boldsymbol{V}_0$ $\mathbf{p}_0^0$ $\mathbf{f}_1^0$ $\mathbf{q}_1^0$ $\boldsymbol{V}_1$ $\mathbf{p}_1^0$ $\mathbf{f}_2^0$ $\mathbf{q}_2^0$ $\boldsymbol{V}_2$ $\mathbf{p}_2^0$ $\mathbf{f}_3^0$ $\mathbf{q}_3^0$ $\boldsymbol{V}_3$ $\mathbf{p}_3^0$
$\mathbf{q}_0^+$ $\boldsymbol{V}_0$ $\mathbf{p}_0^+$ $\mathbf{q}_1^+$ $\boldsymbol{V}_1$ $\mathbf{p}_1^+$ $\mathbf{q}_2^+$ $\boldsymbol{V}_2$ $\mathbf{p}_2^+$ $\mathbf{q}_3^+$ $\boldsymbol{V}_3$ $\mathbf{p}_3^+$
$\mathbf{f}_0^+$ $\mathbf{f}_1^-$ $\mathbf{f}_1^+$ $\mathbf{f}_2^-$ $\mathbf{f}_2^+$ $\mathbf{f}_3^-$ $\mathbf{f}_3^+$ $\mathbf{f}_4^-$
$\mathbf{q}_0^-$ $\boldsymbol{V}_0^\top$ $\mathbf{p}_0^-$ $\mathbf{q}_1^-$ $\boldsymbol{V}_1^\top$ $\mathbf{p}_1^-$ $\mathbf{q}_2^-$ $\boldsymbol{V}_2^\top$ $\mathbf{p}_2^-$ $\mathbf{q}_3^-$ $\boldsymbol{V}_3^\top$ $\mathbf{p}_3^-$
Q_0^0 P_0^0 Q_1^0 P_1^0 Q_2^0 P_2^0 Q_3^0 P_3^0
Q_0^+ $\mathcal{N}$ P_0^+ Q_1^+ $\mathcal{N}$ P_1^+ Q_2^+ $\mathcal{N}$ P_2^+ Q_3^+ $\mathcal{N}$ P_3^+
f_0^+ f_1^- f_1^+ f_2^- f_2^+ f_3^- f_3^+ f_4^-
Q_0^- $\mathcal{N}$ P_0^- Q_1^- $\mathcal{N}$ P_1^- Q_2^- $\mathcal{N}$ P_2^- Q_3^- $\mathcal{N}$ P_3^-

Figure B.1: (TOP) The equations (4.1) with equivalent quantities defined in (B.6), and $\mathbf{f}_\ell^0$ defined using (B.2).
(MIDDLE) The Gen-ML-Mat recursions in Algorithm 10. These are also equivalent to ML-Mat-VAMP recursions from Algorithm 4 (See Lemma 7) if $\mathbf{q}^\pm, \mathbf{p}^\pm$ are as defined as in equations (B.7) and (B.9), and $\mathbf{f}_\ell^\pm$ given by equations (B.4) and (B.3).
(BOTTOM) Quantities in the GEN-ML-SE recursions. These are also equivalent to ML-Mat-VAMP SE recursions from Algorithm 9 (See Lemma 7)
The iteration indices k have been dropped for notational simplicity.

$\mathbf{p}_{k\ell}^-$ are called the *matrix update functions* and use parameter lists $\Upsilon_{kl}^\pm$. The initial parameter lists Υ_{01}^- are assumed to be provided. As the algorithm progresses, new parameters $\lambda_{k\ell}^\pm$ are computed and then added to the lists in lines 12, 18, 25 and 31. The matrix update functions $\mathbf{f}_{k\ell}^\pm(\cdot)$ may depend on any sets of parameters accumulated in the parameter list. In lines 11, 17, 24 and 30, the new parameters $\lambda_{k\ell}^\pm$ are computed by: (1) computing average values $\mu_{k\ell}^\pm$ of *row-wise* functions $\boldsymbol{\varphi}_{k\ell}^\pm(\cdot)$; and (2) taking functions $T_{k\ell}^\pm(\cdot)$ of the average values $\mu_{k\ell}^\pm$. Since the average values $\mu_{k\ell}^\pm$ represent statistics on the rows of $\boldsymbol{\varphi}_{k\ell}^\pm(\cdot)$, we will call $\boldsymbol{\varphi}_{k\ell}^\pm(\cdot)$ the *parameter statistic functions*. We will call the $T_{k\ell}^\pm(\cdot)$ the *parameter update functions*. The functions $\mathbf{f}_\ell^0, \mathbf{f}_{k\ell}^\pm, \boldsymbol{\varphi}_\ell^\pm$ also take as input some perturbation vectors $\boldsymbol{w}_\ell$.

Similar to the analysis of the ML-Mat-VAMP Algorithm, we consider the following large-system limit (LSL) analysis of Gen-ML. Specifically, we consider a sequence of runs of the recursions indexed by N. For each N, let $N_\ell = N_\ell(N)$ be the dimension of the matrix signals $\mathbf{p}_\ell^\pm$ and $\mathbf{q}_\ell^\pm$ as we assume that $\lim\limits_{N\to\infty} \frac{N_\ell}{N} = \beta_\ell \in (0,\infty)$ is a constant so that N_ℓ scales linearly with N. Note however that the number of columns of each of the matrices $\{\mathbf{q}_\ell^0, \mathbf{p}_\ell^0, \mathbf{q}_{k\ell}^\pm, \mathbf{p}_{k\ell}^\pm\}$ is equal to a finite integer $d > 0$, which remains fixed for all N. We then make the following assumptions. See B.2 for an overview of empirical convergence of sequences which we use in the assumptions described below.

Assumption 5. For vectors in the Gen-ML Algorithm (Algorithm 10), we assume:

(a) The matrices $\mathbf{V}_\ell$ are Haar distributed on the set of $N_\ell \times N_\ell$ orthogonal matrices and are independent from one another and from the matrices $\boldsymbol{q}_0^0$, $\mathbf{q}_{0\ell}^-$, perturbation variables $\mathbf{w}_\ell$.

(b) The rows of the initial matrices $\mathbf{q}_{0\ell}^-$, and perturbation variables $\mathbf{w}_\ell$ converge jointly empirically with limits,

$$\mathbf{q}_{0\ell}^- \overset{2}{\Rightarrow} Q_{0\ell}^-, \quad \mathbf{w}_\ell \overset{2}{\Rightarrow} W_\ell, \tag{B.10}$$

where $Q_{0\ell}^-$ are random vectors in $\mathbb{R}^{1\times d}$ such that $(Q_{00}^-, \cdots, Q_{0,L-1}^-)$ is jointly Gaussian. For $\ell = 0, \dots, L-1$, the random variables $W_\ell, P_{\ell-1}^0$ and $Q_{0\ell}^-$ are all independent. We also assume that the initial parameter list converges as

$$\lim_{N\to\infty} \Upsilon_{01}^-(N) \xrightarrow{a.s.} \overline{\Upsilon}_{01}^-, \tag{B.11}$$

to some list $\overline{\Upsilon}_{01}^-$. The limit (B.11) means that every element in the list $\lambda(N) \in \Upsilon_{01}^-(N)$ converges to a limit $\lambda(N) \to \overline{\lambda} \in \overline{\Upsilon}_{01}^-$ as $N \to \infty$ almost surely.

(c) The *matrix update functions* $\mathbf{f}_{k\ell}^\pm(\cdot)$ and *parameter update functions* $\boldsymbol{\varphi}_{k\ell}^\pm(\cdot)$ act row-wise. For e.g., in the kth forward pass, at stage ℓ, we assume that for each output row n,

$$\left[\mathbf{f}_{k\ell}^+(\mathbf{p}_{\ell-1}^0, \mathbf{p}_{k,\ell-1}^+, \mathbf{q}_{k\ell}^-, \mathbf{w}_\ell, \Upsilon_{k\ell}^+)\right]_{n:} = f_{k\ell}^+(\mathbf{p}_{\ell-1,n:}^0, \mathbf{p}_{k,\ell-1,n:}^+, \mathbf{q}_{k\ell,n:}^-, \mathbf{w}_{\ell,n:}, \Upsilon_{k\ell}^+)$$

$$\left[\boldsymbol{\varphi}_{k\ell}^+(\mathbf{p}_{\ell-1}^0, \mathbf{p}_{k,\ell-1}^+, \mathbf{q}_{k\ell}^-, \mathbf{w}_\ell, \Upsilon_{k\ell}^+)\right]_{n:} = \varphi_{k\ell}^+(\mathbf{p}_{\ell-1,n:}^0, \mathbf{p}_{k,\ell-1,n:}^+, \mathbf{q}_{k\ell,n:}^-, \mathbf{w}_{\ell,n:}, \Upsilon_{k\ell}^+),$$

for some $\mathbb{R}^{1\times d}$-valued functions $f^+_{k\ell}(\cdot)$ and $\varphi^+_{k\ell}(\cdot)$. Similar definitions apply in the reverse directions and for the initial vector functions $\mathbf{f}^0_\ell(\cdot)$. We will call $f^\pm_{k\ell}(\cdot)$ the *matrix update row-wise functions* and $\varphi^\pm_{k\ell}(\cdot)$ the *parameter update row-wise functions.*

Next we define a set of *deterministic* constants $\{\mathbf{K}^+_{k\ell}, \boldsymbol{\tau}^-_{k\ell}, \overline{\mu}^\pm_{k\ell}, \overline{\Upsilon}^\pm_{kl}, \boldsymbol{\tau}^0_\ell\}$ and $\mathbb{R}^{1\times d}$-valued random vectors $\{Q^0_\ell, P^0_\ell, Q^\pm_{k\ell}, P^\pm_\ell\}$ which are recursively defined through Algorithm 11, which we call the *Gen-ML-Mat State Evolution* (SE). These recursions in Algorithm closely mirror those in the Gen-ML-Mat algorithm (Algorithm 10). The matrices $\mathbf{q}^\pm_{k\ell}$ and $\mathbf{p}^\pm_{k\ell}$ are replaced by random vectors $Q^\pm_{k\ell}$ and $P^\pm_{k\ell}$; the matrix and parameter update functions $\mathbf{f}^\pm_{k\ell}(\cdot)$ and $\boldsymbol{\varphi}^\pm_{k\ell}(\cdot)$ are replaced by their row-wise functions $f^\pm_{k\ell}(\cdot)$ and $\varphi^\pm_{k\ell}(\cdot)$; and the parameters $\lambda^\pm_{k\ell}$ are replaced by their limits $\overline{\lambda}^\pm_{k\ell}$. We refer to $\{Q^0_\ell, P^0_\ell\}$ as *true random vectors* and $\{Q^\pm_{k\ell}, P^\pm_{kl}\}$ as *iterated random vectors.* The signal flow graph for the true and iterated random variables in Algorithm 11 is given in the (BOTTOM) panel of Fig. B.1. The iteration index k for the iterated random variables $\{Q^\pm_{k\ell}, P^\pm_{kl}\}$ to simplify notation.

We also assume the following about the behaviour of row-wise functions around the quantities defined in Algorithm 11. The iteration index k has been dropped for simplifying notation.

Assumption 6. For row-wise functions f, φ and parameter update functions T we assume:

(a) $T^\pm_{k\ell}(\mu^\pm_{k\ell}, \cdot)$ are continuous at $\mu^\pm_{k\ell} = \overline{\mu}^\pm_{k\ell}$

(b) $f^+_{k\ell}(p^0_{\ell-1}, p^+_{k,\ell-1}, q^-_{k\ell}, w_\ell, \Upsilon^+_{k\ell})$, $\frac{\partial f^+_{k\ell}}{\partial q^-_{k\ell}}(p^0_{\ell-1}, p^+_{k,\ell-1}, q^-_{k\ell}, w_\ell, \Upsilon^+_{k\ell})$ and $\varphi^+_{k\ell}(p^0_{\ell-1}, p^+_{k,\ell-1}, q^-_{k\ell}, w_\ell, \Upsilon^+_{k,\ell-1})$ are uniformly Lipschitz continuous in $(p^0_{\ell-1}, p^+_{k,\ell-1}, q^-_{k\ell}, w_\ell)$ at $\Upsilon^+_{k\ell} = \overline{\Upsilon}^+_{k\ell}$, $\Upsilon^+_{k,\ell-1} = \overline{\Upsilon}^+_{k,\ell-1}$.
Similarly, $f^-_{k+1,\ell}(p^0_{\ell-1}, p^+_{k,\ell-1}, q^-_{k+1,\ell}, w_\ell, \Upsilon^-_{k\ell})$, $\frac{\partial f^-_{k\ell}}{\partial p^+_{k,\ell-1}}(p^0_{\ell-1}, p^+_{k,\ell-1}, q^-_{k+1,\ell}, w_\ell, \Upsilon^-_{k\ell})$, and $\varphi^-_{k\ell}(p^0_{\ell-1}, p^+_{k,\ell-1}, q^-_{k+1,\ell}, w_\ell, \Upsilon^-_{k+1,\ell+1})$ are uniformly Lipschitz continuous in $(p^0_{\ell-1}, p^+_{k,\ell-1}, q^-_{k+1,\ell}, w_\ell)$ at $\Upsilon^-_{k\ell} = \overline{\Upsilon}^-_{k\ell}$, $\Upsilon^-_{k+1,\ell+1} = \overline{\Upsilon}^-_{k+1,\ell+1}$.

(c) $f^0_\ell(p^0_{\ell-1}, w_\ell, \Upsilon^-_{01})$ are uniformly Lipschitz continuous in $(p^0_{k,\ell-1}, w_\ell)$ at $\Upsilon^-_{k+1,\ell} = \overline{\Upsilon}^-_{k+1,\ell}$.

(d) Matrix update functions $\mathbf{f}^{\pm}_{k\ell}$ are *asymptotically divergence free* meaning

$$\lim_{N\to\infty}\left\langle \frac{\partial \mathbf{f}^{+}_{k\ell}}{\partial \mathbf{q}^{-}_{k\ell}}(\mathbf{p}^{+}_{k,\ell-1}, \mathbf{q}^{-}_{k\ell}, \mathbf{w}_{\ell}, \overline{\Upsilon}^{+}_{k\ell})\right\rangle = \mathbf{0}, \quad \lim_{N\to\infty}\left\langle \frac{\partial \mathbf{f}^{-}_{k\ell}}{\partial \mathbf{p}^{+}_{k,\ell-1}}(\mathbf{p}^{+}_{k,\ell-1}, \mathbf{q}^{-}_{k+1,\ell}, \mathbf{w}_{\ell}, \overline{\Upsilon}^{-}_{k\ell})\right\rangle = \mathbf{0} \tag{B.12}$$

We are now ready to state the general result regarding the empirical convergence of the true and iterated vectors from Algorithm 10 in terms of random variables defined in Algorithm 11.

Theorem 12. *Consider the iterates of the Gen-ML recursion (Algorithm 10) and the corresponding random variables and parameter limits defined by the SE recursions (Algorithm 11) under Assumptions 5 and 6. Then,*

(a) For any fixed $k \geq 0$ and fixed $\ell = 1, \ldots, L-1$, the parameter list $\Upsilon^{+}_{k\ell}$ converges as

$$\lim_{N\to\infty} \Upsilon^{+}_{k\ell} = \overline{\Upsilon}^{+}_{k\ell} \tag{B.13}$$

almost surely. Also, the rows of $\mathbf{w}_{\ell}$, $\mathbf{p}^{0}_{\ell-1}$, $\mathbf{q}^{0}_{\ell}$, $\mathbf{p}^{+}_{0,\ell-1}, \ldots, \mathbf{p}^{+}_{k,\ell-1}$ and $\mathbf{q}^{\pm}_{0\ell}, \ldots, \mathbf{q}^{\pm}_{k\ell}$ almost surely jointly converge empirically with limits,

$$(\boldsymbol{p}^{0}_{\ell-1}, \boldsymbol{p}^{+}_{i,\ell-1}, \boldsymbol{q}^{-}_{j\ell}, \boldsymbol{q}^{0}_{\ell}, \boldsymbol{q}^{+}_{j\ell}) \xrightarrow{2} (P^{0}_{\ell-1}, P^{+}_{i,\ell-1}, Q^{-}_{j\ell}, Q^{0}_{\ell}, Q^{+}_{j\ell}), \tag{B.14}$$

for all $0 \leq i, j \leq k$, where the variables $P^{0}_{\ell-1}$, $P^{+}_{i,\ell-1}$ and $Q^{-}_{j\ell}$ are zero-mean jointly Gaussian random variables independent of W_{ℓ} and with covariance matrix given by

$$\mathrm{Cov}(P^{0}_{\ell-1}, P^{+}_{i,\ell-1}) = \mathbf{K}^{+}_{i,\ell-1}, \quad \mathbb{E}(Q^{-}_{j\ell})^2 = \boldsymbol{\tau}^{-}_{j\ell}, \quad \mathbb{E}(P^{+\mathsf{T}}_{i,\ell-1} Q^{-}_{j\ell}) = \mathbf{0}, \quad \mathbb{E}(P^{0\mathsf{T}}_{\ell-1} Q^{-}_{j\ell}) = \mathbf{0}, \tag{B.15}$$

*and Q^{0}_{ℓ}, $Q^{+}_{j\ell}$ are the random variable in lines 4, 19,*i.e.,

$$Q^{0}_{\ell} = f^{0}_{\ell}(P^{0}_{\ell-1}, W_{\ell}), \quad Q^{+}_{j\ell} = f^{+}_{j\ell}(P^{0}_{\ell-1}, P^{+}_{j,\ell-1}, Q^{-}_{j\ell}, W_{\ell}, \overline{\Upsilon}^{+}_{j\ell}). \tag{B.16}$$

An identical result holds for $\ell = 0$ with all the variables $\mathbf{p}^{+}_{i,\ell-1}$ and $P^{+}_{i,\ell-1}$ removed.

(b) For any fixed $k \geq 1$ and fixed $\ell = 1, \ldots, L-1$, the parameter lists $\Upsilon^{-}_{k\ell}$ converge as

$$\lim_{N\to\infty} \Upsilon^{-}_{k\ell} = \overline{\Upsilon}^{-}_{k\ell} \tag{B.17}$$

almost surely. Also, the rows of $\mathbf{w}_\ell$, $\mathbf{p}^0_{\ell-1}$, $\mathbf{p}^{\pm}_{0,\ell-1}, \ldots, \mathbf{p}^{\pm}_{k-1,\ell-1}$, and $\mathbf{q}^-_{0\ell}, \ldots, \mathbf{q}^-_{k\ell}$ almost surely jointly converge empirically with limits,

$$(\mathbf{p}^0_{\ell-1}, \mathbf{p}^+_{i,\ell-1}, \mathbf{q}^-_{j\ell}, \mathbf{p}^-_{j,\ell-1}) \overset{2}{\Rightarrow} (P^0_{\ell-1}, P^+_{i,\ell-1}, Q^-_{j\ell}, P^-_{j,\ell-1}), \tag{B.18}$$

for all $0 \le i \le k-1$ and $0 \le j \le k$, where the variables $P^0_{\ell-1}$, $P^+_{i,\ell-1}$ and $Q^-_{j\ell}$ are zero-mean jointly Gaussian random variables independent of W_ℓ and with covariance matrix given by equation (B.15) and $P^-_{j\ell}$ is the random variable in line 32:

$$P^-_{j\ell} = f^-_{j\ell}(P^0_{\ell-1}, P^+_{j-1,\ell-1}, Q^-_{j\ell}, W_\ell, \overline{\Upsilon}^-_{j\ell}). \tag{B.19}$$

An identical result holds for $\ell = L$ with all the variables $\mathbf{q}^-_{j\ell}$ and $Q^-_{j\ell}$ removed.

For $k = 0$, $\Upsilon^-_{01} \to \overline{\Upsilon}^-_{01}$ almost surely, and the rows $\{(\mathbf{w}_{\ell,n:}, \mathbf{p}^0_{\ell-1,n:}, \mathbf{q}^-_{j\ell,n:})\}_{n=1}^N$ empirically converge to independent random variables $(W_\ell, P^0_{\ell-1}, Q^-_{0\ell})$.

Proof. B.5 is dedicated to proving this result. □

B.5 Proof of Theorem 12

The proof proceeds using mathematical induction. It largely mimics the proof for the case of $d = 1$ which were the convergence results in [114, Thm. 5]. However, in the case of $d > 1$, we observe that several quantities which were scalars in proving [114, Thm. 5] are now matrices. Due to the non-commutativity of these matrix quantities, we re-state the whole prove, while modifying the requisite matrix terms appropriately.

B.5.1 Overview of the Induction Sequence

The proof is similar to that of [127, Theorem 4], which provides a SE analysis for VAMP on a single-layer network. The critical challenge here is to extend that proof to multi-layer recursions. Many of the ideas in the two proofs are similar, so we highlight only the key differences between the two.

Similar to the SE analysis of VAMP in [127], we use an induction argument. However, for the multi-layer proof, we must index over both the iteration index k and layer index ℓ. To this end, let $\mathcal{H}^+_{k\ell}$ and $\mathcal{H}^-_{k\ell}$ be the hypotheses:

- $\mathcal{H}^+_{k\ell}$: The hypothesis that Theorem 12(a) is true for a given k and ℓ, where $0 \le \ell \le L-1$.
- $\mathcal{H}^-_{k\ell}$: The hypothesis that Theorem 12(b) is true for a given k and ℓ, where $1 \le \ell \le L$.

We prove these hypotheses by induction via a sequence of implications,

$$\{\mathcal{H}^-_{0\ell}\}_{\ell=1}^{L} \cdots \Rightarrow \mathcal{H}^-_{k1} \Rightarrow \mathcal{H}^+_{k0} \Rightarrow \cdots \Rightarrow \mathcal{H}^+_{k,L-1} \Rightarrow \mathcal{H}^-_{k+1,L} \Rightarrow \cdots \Rightarrow \mathcal{H}^-_{k+1,1} \Rightarrow \cdots, \tag{B.20}$$

beginning with the hypotheses $\{\mathcal{H}^-_{0\ell}\}$ for all $\ell = 1, \ldots, L-1$.

B.5.2 Base Case: Proof of $\{\mathcal{H}^-_{0\ell}\}_{\ell=1}^{L}$

The base case corresponds to the hypotheses $\{\mathcal{H}^-_{0\ell}\}_{\ell=1}^{L}$. Note that Theorem 12(b) states that for $k = 0$, we need $\Upsilon^-_{01} \to \overline{\Upsilon}^-_{01}$ almost surely, and $\{(\mathbf{w}_{\ell,n:}, \boldsymbol{p}^0_{\ell-1,n:}, \mathbf{q}^-_{j\ell,n:})\}_{n=1}^{N}$ empirically converge to independent random variables $(W_\ell, P^0_{\ell-1}, Q^-_{0\ell})$. These follow directly from equations (B.10) and (B.11) in Assumption 1 (a).

B.5.3 Inductive Step: Proof of $\mathcal{H}^+_{k,\ell+1}$

Fix a layer index $\ell = 1, \ldots, L-1$ and an iteration index $k = 0, 1, \ldots$. We show the implication $\cdots \implies \mathcal{H}^+_{k,\ell+1}$ in (B.20). All other implications can be proven similarly using symmetry arguments.

Definition 8 (Induction hypothesis)**.** The hypotheses prior to $\mathcal{H}^+_{k,\ell+1}$ in the sequence (B.20), but not including $\mathcal{H}^+_{k,\ell+1}$, are true.

The inductive step then corresponds to the following result.

Lemma 9. *Under the induction hypothesis, $\mathcal{H}^+_{k,\ell+1}$ holds*

Before proving the inductive step in Lemma 9, we prove two intermediate lemmas. Let us start by defining some notation. Define $\mathbf{P}^{+}_{k\ell} := \left[\mathbf{p}^{+}_{0\ell} \cdots \mathbf{p}^{+}_{k\ell}\right] \in \boldsymbol{R}^{N_\ell \times (k+1)d}$, be a matrix whose column blocks are the first $k+1$ values of the matrix $\mathbf{p}^{+}_{\ell}$. We define the matrices $\mathbf{P}^{-}_{k\ell}$, $\mathbf{Q}^{+}_{k\ell}$ and $\mathbf{Q}^{-}_{k\ell}$ in a similar manner with values of $\mathbf{p}^{-}_{\ell}$, $\mathbf{q}^{+}_{\ell}$ and $\mathbf{q}^{-}_{\ell}$ respectively.

Note that except the initial matrices $\{\mathbf{w}_\ell, \mathbf{q}^{-}_{0\ell}\}_{\ell=1}^{L}$, all later iterates in Algorithm 10 are random due to the randomness of $\boldsymbol{V}_\ell$. Let $\mathfrak{G}^{\pm}_{k\ell}$ denote the collection of random variables associated with the hypotheses, $\mathcal{H}^{\pm}_{k\ell}$. That is, for $\ell = 1, \ldots, L-1$,

$$\mathfrak{G}^{+}_{k\ell} := \left\{\mathbf{w}_\ell, \mathbf{p}^{0}_{\ell-1}, \mathbf{P}^{+}_{k,\ell-1}, \mathbf{q}^{0}_{\ell}, \mathbf{Q}^{-}_{k\ell}, \mathbf{Q}^{+}_{k\ell}\right\}, \qquad \mathfrak{G}^{-}_{k\ell} := \left\{\mathbf{w}_\ell, \mathbf{p}^{0}_{\ell-1}, \mathbf{P}^{+}_{k-1,\ell-1}, \mathbf{q}^{0}_{\ell}, \mathbf{Q}^{-}_{k\ell}, \mathbf{P}^{-}_{k,\ell-1}\right\}.$$

For $\ell = 0$ and $\ell = L$ we set, $\mathfrak{G}^{+}_{k0} := \left\{\mathbf{w}_0, \mathbf{Q}^{-}_{k0}, \mathbf{Q}^{+}_{k0}\right\}$, $\quad \mathfrak{G}^{-}_{kL} := \left\{\mathbf{w}_L, \mathbf{p}^{0}_{L-1}, \mathbf{P}^{+}_{k-1,L-1}, \mathbf{P}^{-}_{k,L-1}\right\}$.

Let $\overline{\mathfrak{G}}^{+}_{k\ell}$ be the sigma algebra generated by the union of all the sets $\mathfrak{G}^{\pm}_{k'\ell'}$ as they have appeared in the sequence (B.20) up to and including the final set $\mathfrak{G}^{+}_{k\ell}$. Thus, the sigma algebra $\overline{\mathfrak{G}}^{+}_{k\ell}$ contains all *information* produced by Algorithm 10 immediately *before* line 20 in layer ℓ of iteration k. Note also that the random variables in Algorithm 11 immediately before defining $P^{+}_{k,\ell}$ in line 20 are all $\overline{\mathfrak{G}}^{+}_{k\ell}$ measurable.

Observe that the matrix $\mathbf{V}_\ell$ in Algorithm 10 appears only during matrix-vector multiplications in lines 20 and 32. If we define the matrices, $\mathbf{A}_{k\ell} := \left[\mathbf{p}^{0}_{\ell}, \mathbf{P}^{+}_{k-1,\ell}\ \mathbf{P}^{-}_{k\ell}\right]$, $\quad \mathbf{B}_{k\ell} := \left[\mathbf{q}^{0}_{\ell}, \mathbf{Q}^{+}_{k-1,\ell}\ \mathbf{Q}^{-}_{k\ell}\right]$, all the matrices in the set $\overline{\mathfrak{G}}^{+}_{k\ell}$ will be unchanged for all matrices $\mathbf{V}_\ell$ satisfying the linear constraints

$$\mathbf{A}_{k\ell} = \mathbf{V}_\ell \mathbf{B}_{k\ell}. \tag{B.21}$$

Hence, the conditional distribution of $\mathbf{V}_\ell$ given $\overline{\mathfrak{G}}^{+}_{k\ell}$ is precisely the uniform distribution on the set of orthogonal matrices satisfying (B.21). The matrices $\mathbf{A}_{k\ell}$ and $\mathbf{B}_{k\ell}$ are of dimensions $N_\ell \times (2k+2)d$. From [127, Lemmas 3,4], this conditional distribution is given by

$$\mathbf{V}_\ell|_{\overline{\mathfrak{G}}^{+}_{k\ell}} \stackrel{d}{=} \mathbf{A}_{k\ell}(\mathbf{A}^{\mathsf{T}}_{k\ell}\mathbf{A}_{k\ell})^{-1}\mathbf{B}^{\mathsf{T}}_{k\ell} + \mathbf{U}_{\mathbf{A}^{\perp}_{k\ell}} \widetilde{\mathbf{V}}_\ell \mathbf{U}^{\mathsf{T}}_{\mathbf{B}^{\perp}_{k\ell}}, \tag{B.22}$$

where $\mathbf{U}_{\mathbf{A}^{\perp}_{k\ell}}$ and $\mathbf{U}_{\mathbf{B}^{\perp}_{k\ell}}$ are $N_\ell \times (N_\ell - (2k+2)d)$ matrices whose columns are an orthonormal basis for Range$(\mathbf{A}_{k\ell})^{\perp}$ and Range$(\mathbf{B}_{k\ell})^{\perp}$. The matrix $\widetilde{\mathbf{V}}_\ell$ is Haar distributed on the set of $(N_\ell - (2k+2)d) \times (N_\ell - (2k+2)d)$ orthogonal matrices and is independent of $\overline{\mathfrak{G}}^{+}_{k\ell}$.

Next, similar to the proof of [127, Thm. 4], we can use (B.22) to write the conditional distribution of $\mathbf{p}_{k\ell}^+$ (from line 20 of Algorithm 10) given $\overline{\mathfrak{G}}_{k\ell}^+$ as a sum of two terms

$$\mathbf{p}_{k\ell}^+|_{\overline{\mathfrak{G}}_{k\ell}^+} = \mathbf{V}_\ell|_{\overline{\mathfrak{G}}_{k\ell}^+}\, \mathbf{q}_{k\ell}^+ \overset{d}{=} \mathbf{p}_{k\ell}^{+\mathrm{det}} + \mathbf{p}_{k\ell}^{+\mathrm{ran}}, \tag{B.23a}$$

$$\mathbf{p}_{k\ell}^{+\mathrm{det}} := \mathbf{A}_{k\ell}(\mathbf{B}_{k\ell}^\intercal \mathbf{B}_{k\ell})^{-1}\mathbf{B}_{k\ell}^\intercal \mathbf{q}_{k\ell}^+ \tag{B.23b}$$

$$\mathbf{p}_{k\ell}^{+\mathrm{ran}} := \mathbf{U}_{\mathbf{B}_k^\perp}\widetilde{\mathbf{V}}_\ell^\intercal \mathbf{U}_{\mathbf{A}_k^\perp}^\intercal \mathbf{q}_{k\ell}^+. \tag{B.23c}$$

where we call $\mathbf{p}_{k\ell}^{+\mathrm{det}}$ the *deterministic* term and $\mathbf{p}_{k\ell}^{+\mathrm{ran}}$ the *random* term. The next two lemmas characterize the limiting distributions of the deterministic and random terms.

Lemma 10. *Under the induction hypothesis, the rows of the "deterministic" term $\mathbf{p}_{k\ell}^{+\mathrm{det}}$ along with the rows of the matrices in $\overline{\mathfrak{G}}_{k\ell}^+$ converge empirically. In addition, there exists constant $d \times d$ matrices $\beta_{0\ell}^+, \ldots, \beta_{k-1,\ell}^+$ such that*

$$\mathbf{p}_{k\ell}^{+\mathrm{det}} \overset{2}{\Rightarrow} P_{k\ell}^{+\mathrm{det}} := P_\ell^0 \beta_\ell^0 + \sum_{i=0}^{k-1} P_{i\ell}^+ \beta_{i\ell}, \tag{B.24}$$

where $P_{k\ell}^{+\mathrm{det}} \in \mathbb{R}^{1\times d}$ is the limiting random vector for the rows of $\mathbf{p}_{k\ell}^{\mathrm{det}}$.

Proof. The proof is similar that of [127, Lem. 6], but we go over the details as there are some important differences in the multi-layer matrix case. Define $\widetilde{\mathbf{P}}_{k-1,\ell}^+ = \left[\mathbf{p}_\ell^0,\ \mathbf{P}_{k-1,\ell}^+\right], \widetilde{\mathbf{Q}}_{k-1,\ell}^+ = \left[\mathbf{q}_\ell^0,\ \mathbf{Q}_{k-1,\ell}^+\right]$, which are the matrices in $\mathbb{R}^{N_\ell \times (k+1)d}$. We can then write $\mathbf{A}_{k\ell}$ and $\mathbf{B}_{k\ell}$ from (B.21) as

$$\mathbf{A}_{k\ell} := \left[\widetilde{\mathbf{P}}_{k-1,\ell}^+\ \mathbf{P}_{k\ell}^-\right], \quad \mathbf{B}_{k\ell} := \left[\widetilde{\mathbf{Q}}_{k-1,\ell}^+\ \mathbf{Q}_{k\ell}^-\right], \tag{B.25}$$

We first evaluate the asymptotic values of various terms in (B.23b). By definition of $\mathbf{B}_{k\ell}$ in (B.25),

$$\mathbf{B}_{k\ell}^\intercal \mathbf{B}_{k\ell} = \begin{bmatrix} (\widetilde{\mathbf{Q}}_{k-1,\ell}^+)^\intercal \widetilde{\mathbf{Q}}_{k-1,\ell}^+ & (\widetilde{\mathbf{Q}}_{k-1,\ell}^+)^\intercal \mathbf{Q}_{k\ell}^- \\ (\mathbf{Q}_{k\ell}^-)^\intercal \widetilde{\mathbf{Q}}_{k-1,\ell}^+ & (\mathbf{Q}_{k\ell}^-)^\intercal \mathbf{Q}_{k\ell}^- \end{bmatrix}$$

We can then evaluate the asymptotic values of these terms as follows: For $0 \le i, j \le k-1$

the asymptotic value of the $(i+2, j+2)^{\rm nd}$ $d \times d$ block of the matrix $(\widetilde{\mathbf{Q}}_{k-1,\ell}^{+})^{\mathsf{T}}\widetilde{\mathbf{Q}}_{k-1,\ell}^{+}$ is

$$\begin{aligned}\lim_{N\to\infty} \tfrac{1}{N_\ell}\left[(\widetilde{\mathbf{Q}}_{k-1,\ell}^{+})^{\mathsf{T}}\widetilde{\mathbf{Q}}_{k-1,\ell}^{+}\right]_{i+2,j+2} &\overset{(a)}{=} \lim_{N\to\infty}\frac{1}{N_\ell}(\mathbf{q}_{i\ell}^{+})^{\mathsf{T}}\mathbf{q}_{j\ell}^{+} \\ &= \lim_{N\to\infty}\tfrac{1}{N_\ell}\sum_{n=1}^{N_\ell}[\mathbf{q}_{i\ell}^{+}]_{n:}[\mathbf{q}_{j\ell}^{+}]_{n:}^{\mathsf{T}} \overset{(b)}{=} \mathbb{E}\left[Q_{i\ell}^{+\mathsf{T}}Q_{j\ell}^{+}\right]\end{aligned}$$

where (a) follows since the $(i+2)^{\rm th}$ column block of $\widetilde{\mathbf{Q}}_{k-1,\ell}^{+}$ is $\mathbf{q}_{i\ell}^{+}$, and (b) follows due to the empirical convergence assumption in (B.14). Also, since the first column block of $\widetilde{\mathbf{Q}}_{k-1,\ell}^{+}$ is $\mathbf{q}_{\ell}^{0}$, we obtain that

$$\begin{aligned}&\lim_{N_\ell\to\infty}\tfrac{1}{N_\ell}(\widetilde{\mathbf{Q}}_{k-1,\ell}^{+})^{\mathsf{T}}\widetilde{\mathbf{Q}}_{k-1,\ell}^{+} = \mathbf{R}_{k-1,\ell}^{+} \qquad \text{and} \\ &\lim_{N_\ell\to\infty}\tfrac{1}{N_\ell}(\mathbf{Q}_{k\ell}^{-})^{\mathsf{T}}\mathbf{Q}_{k\ell}^{-} = \mathbf{R}_{k\ell}^{-},\end{aligned} \tag{B.26}$$

where $\mathbf{R}_{k-1,\ell}^{+} \in \mathbb{R}^{(k+1)d\times(k+1)d}$ is the covariance matrix of $\left[Q_\ell^0\ Q_{0\ell}^{+}\ \ldots\ Q_{k-1,\ell}^{+}\right]$, and $\mathbf{R}_{k\ell}^{-} \in \mathbb{R}^{(k+1)d\times(k+1)d}$ is the covariance matrix of $\left[Q_{0\ell}^{-}\ Q_{1\ell}^{-}\ \ldots\ Q_{k\ell}^{-}\right]$. For the matrix $(\widetilde{\mathbf{Q}}_{k-1,\ell}^{+})^{\mathsf{T}}\mathbf{Q}_{k\ell}^{-}$, first observe that the limit of the divergence free condition (B.12) implies

$$\mathbb{E}\left[\frac{\partial f_{i\ell}^{+}(P_{i,\ell-1}^{+}, Q_{i\ell}^{-}, W_\ell, \overline{\Upsilon}_{i\ell})}{\partial Q_{i\ell}^{-}}\right] = \lim_{N_\ell\to\infty}\left\langle \frac{\partial \mathbf{f}_{i\ell}^{+}(\mathbf{p}_{i,\ell-1}^{+}, \mathbf{q}_{i\ell}^{-}, \mathbf{w}_\ell, \overline{\Upsilon}_{i\ell}^{+})}{\partial \mathbf{q}_{i\ell}^{-}}\right\rangle = \mathbf{0}, \tag{B.27}$$

for any i. Also, by the induction hypothesis $\mathcal{H}_{k\ell}^{+}$,

$$\mathbb{E}(P_{i,\ell-1}^{+\mathsf{T}}Q_{j\ell}^{-}) = \mathbf{0}, \quad \mathbb{E}(P_{\ell-1}^{0\mathsf{T}}Q_{j\ell}^{-}) = \mathbf{0}, \tag{B.28}$$

for all $0 \le i, j \le k$. Therefore using (B.16), the cross-terms $\mathbb{E}(Q_{i\ell}^{+\mathsf{T}}Q_{j\ell}^{-})$ are given by

$$\begin{aligned}\mathbb{E}(f_{i\ell}^{+}(P_{\ell-1}^{0}, P_{i,\ell-1}^{+}, Q_{i\ell}^{-}, W_\ell, \overline{\Upsilon}_{i\ell})^{\mathsf{T}}Q_{j\ell}^{-}) &\overset{(a)}{=} \mathbb{E}\left[\tfrac{\partial f_{i\ell}^{+}(P_{\ell-1}^{0}, P_{i,\ell-1}^{+}, Q_{i\ell}^{-}, W_\ell, \overline{\Upsilon}_{i\ell}^{+})}{\partial P_{\ell-1}^{0}}\right]\mathbb{E}(P_{\ell-1}^{0\mathsf{T}}Q_{j\ell}^{-}) \\ &+ \mathbb{E}\left[\tfrac{\partial f_{i\ell}^{+}(P_{\ell-1}^{0}, P_{i,\ell-1}^{+}, Q_{i\ell}^{-}, W_\ell, \overline{\Upsilon}_{i\ell}^{+})}{\partial P_{i,\ell-1}^{+}}\right]\mathbb{E}(P_{i,\ell-1}^{+\mathsf{T}}Q_{j\ell}^{-}) \\ &+ \mathbb{E}\left[\tfrac{\partial f_{i\ell}^{+}(P_{\ell-1}^{0}, P_{i,\ell-1}^{+}, Q_{i\ell}^{-}, W_\ell, \overline{\Upsilon}_{i\ell}^{+})}{\partial Q_{i\ell}^{-}}\right]\mathbb{E}(Q_{i\ell}^{-\mathsf{T}}Q_{j\ell}^{-}) \overset{(b)}{=} \mathbf{0},\end{aligned} \tag{B.29}$$

(a) follows from a multivariate version of Stein's Lemma [**?**, eqn.(2)]; and (b) follows from (B.27), and (B.28). Consequently,

$$\lim_{N_\ell\to\infty}\tfrac{1}{N_\ell}\mathbf{B}_{k\ell}^{\mathsf{T}}\mathbf{B}_{k\ell} = \begin{bmatrix}\mathbf{R}_{k-1,\ell}^{+} & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_{k\ell}^{-}\end{bmatrix}, \quad \text{and} \lim_{N_\ell\to\infty}\tfrac{1}{N_\ell}\mathbf{B}_{k\ell}^{\mathsf{T}}\mathbf{q}_{k\ell}^{+} = \begin{bmatrix}\mathbf{b}_{k\ell}^{+} \\ \mathbf{0}\end{bmatrix}, \tag{B.30}$$

where $\mathbf{b}_{k\ell}^{+} := \left[\mathbb{E}(Q_{0\ell}^{+\mathsf{T}}Q_{k\ell}^{+})\ \mathbb{E}(Q_{1\ell}^{+\mathsf{T}}Q_{k\ell}^{+})\ \cdots\ \mathbb{E}(Q_{k-1,\ell}^{+\mathsf{T}}Q_{k\ell}^{+})\right]^{\mathsf{T}}$, is the matrix of correlations. We again have $\mathbf{0}$ in the second term because $\mathbb{E}[Q_{i\ell}^{+\mathsf{T}} Q_{j\ell}^{-}] = \mathbf{0}$ for all $0 \le i, j \le k$. Hence we have

$$\lim_{N_\ell \to \infty} (\mathbf{B}_{k\ell}^{\mathsf{T}} \mathbf{B}_{k\ell})^{-1} \mathbf{B}_{k\ell}^{\mathsf{T}} \mathbf{q}_{k\ell}^{+} = \begin{bmatrix} \boldsymbol{\beta}_{k\ell}^{+} \\ \mathbf{0} \end{bmatrix}, \quad \boldsymbol{\beta}_{k\ell}^{+} := \begin{bmatrix} \mathbf{R}_{k-1,\ell}^{+} \end{bmatrix}^{-1} \mathbf{b}_{k\ell}^{+}. \tag{B.31}$$

Therefore, $\mathbf{p}_{k\ell}^{+\text{det}}$ equals

$$\begin{aligned} \mathbf{A}_{k\ell} (\mathbf{B}_{k\ell}^{\mathsf{T}} \mathbf{B}_{k\ell})^{-1} \mathbf{B}_{k\ell}^{\mathsf{T}} \mathbf{q}_{k\ell}^{+} &= \begin{bmatrix} \widetilde{\mathbf{P}}_{k-1,\ell}^{+} \ \mathbf{P}_{k,\ell}^{-} \end{bmatrix} \begin{bmatrix} \boldsymbol{\beta}_{k\ell}^{+} \\ \mathbf{0} \end{bmatrix} + O\left(\tfrac{1}{N_\ell}\right) \\ &= \mathbf{p}_\ell^0 \beta_\ell^0 + \sum_{i=0}^{k-1} \mathbf{p}_{i\ell}^{+} \beta_{i\ell}^{+} + O\left(\tfrac{1}{N_\ell}\right), \end{aligned} \tag{B.32}$$

where β_ℓ^0 and $\beta_{i\ell}^{+}$ are $d \times d$ block matrices of $\boldsymbol{\beta}_{k\ell}^{+}$ and the term $O(\frac{1}{N_\ell})$ means a matrix sequence, $\boldsymbol{\varphi}(N) \in \boldsymbol{R}^{N_\ell}$ such that $\lim_{N\to\infty} \frac{1}{N} \|\boldsymbol{\varphi}(N)\|^2 = 0$. A continuity argument then shows the empirical convergence (B.24). $\square$

Lemma 11. *Under the induction hypothesis, the components of the "random" term* $\mathbf{p}_{k\ell}^{+\text{ran}}$ *along with the components of the vectors in* $\overline{\mathfrak{G}}_{k\ell}^{+}$ *almost surely converge empirically. The components of* $\mathbf{p}_{k\ell}^{+\text{ran}}$ *converge as*

$$\mathbf{p}_{k\ell}^{+\text{ran}} \overset{2}{\Rightarrow} U_{k\ell}, \tag{B.33}$$

where $U_{k\ell}$ *is a zero mean Gaussian random vector in* $\mathbb{R}^{1\times d}$ *independent of the limiting random variables corresponding to the variables in* $\overline{\mathfrak{G}}_{k\ell}^{+}$.

Proof. The proof is identical to that of [127, Lemmas 7,8]. $\square$

We are now ready to prove Lemma 9.

Proof of Lemma 9. Using the partition (B.23a) and Lemmas 10 and 11, we see that the components of the vector sequences in $\overline{\mathfrak{G}}_{k\ell}^{+}$ along with $\mathbf{p}_{k\ell}^{+}$ almost surely converge jointly empirically, where the components of $\mathbf{p}_{k\ell}^{+}$ have the limit

$$\mathbf{p}_{k\ell}^{+} = \mathbf{p}_{k\ell}^{\text{det}} + \mathbf{p}_{k\ell}^{\text{ran}} \overset{2}{\Rightarrow} P_\ell^0 \beta_\ell^0 + \sum_{i=0}^{k-1} P_{i\ell}^{+} \beta_{i\ell}^{+} + U_{k\ell} =: P_{k\ell}^{+}. \tag{B.34}$$

Note that the above Wasserstein-2 convergence can be shown using the same arguments involved in showing that if $X_N|\mathcal{F} \overset{d}{\Longrightarrow} X|\mathcal{F}$, and $Y_N|\mathcal{F} \overset{d}{\Longrightarrow} c$, then $(X_N, Y_N)|\mathcal{F} \overset{d}{\Longrightarrow}$

$(X, c)|\mathcal{F}$ for some constant c and sigma-algebra $\mathcal{F}$.

We first establish the Gaussianity of $P^+_{k\ell}$. Observe that by the induction hypothesis, $\mathcal{H}^-_{k,\ell+1}$ holds whereby $(P^0_\ell, P^+_{0\ell}, \ldots, P^+_{k-1,\ell}, Q^-_{0,\ell+1}, \ldots, Q^-_{k,\ell+1})$, is jointly Gaussian. Since U_k is Gaussian and independent of $(P^0_\ell, P^+_{0\ell}, \ldots, P^+_{k-1,\ell}, Q^-_{0,\ell+1}, \ldots, Q^-_{k,\ell+1})$, we can conclude from (B.34) that $(P^0_\ell, P^+_{0\ell}, \ldots, P^+_{k-1,\ell}, P^+_{k\ell}, Q^-_{0,\ell+1}, \ldots, Q^-_{k,\ell+1})$ is jointly Gaussian.

We now need to prove the correlations of this jointly Gaussian random vector are as claimed by $\mathcal{H}^+_{k,\ell+1}$. Since $\mathcal{H}^-_{k,\ell+1}$ is true, we know that (B.15) is true for all $i = 0, \ldots, k-1$ and $j = 0, \ldots, k$ and $\ell = \ell + 1$. Hence, we need only to prove the additional identity for $i = k$, namely the equations: $\mathrm{Cov}(P^0_\ell, P^+_{k\ell})^2 = \mathbf{K}^+_{k\ell}$ and $\mathbb{E}(P^+_{k\ell} Q^-_{j,\ell+1}) = 0$. First observe that

$$\mathbb{E}(P^{+\mathsf{T}}_{k\ell} P^+_{k\ell})^2 \overset{(a)}{=} \lim_{N_\ell \to \infty} \tfrac{1}{N_\ell} \mathbf{p}^{+\mathsf{T}}_{k\ell} \mathbf{p}^+_{k\ell} \overset{(b)}{=} \lim_{N_\ell \to \infty} \tfrac{1}{N_\ell} \mathbf{q}^{+\mathsf{T}}_{k\ell} \mathbf{q}^+_{k\ell} \overset{(c)}{=} \mathbb{E}\left(Q^{+\mathsf{T}}_{k\ell} Q^+_{k\ell}\right)^2$$

where (a) follows from the fact that the rows of $\mathbf{p}^+_{k\ell}$ converge empirically to $P^+_{k\ell}$; (b) follows from line 20 in Algorithm 10 and the fact that $\mathbf{V}_\ell$ is orthogonal; and (c) follows from the fact that the rows of $\mathbf{q}^+_{k\ell}$ converge empirically to $Q^+_{k\ell}$ from hypothesis $\mathcal{H}^+_{k,\ell}$. Since $\mathbf{p}^0_\ell = \mathbf{V}_\ell \mathbf{q}^0$, we similarly obtain that $\mathbb{E}(P^{0\mathsf{T}}_\ell P^+_{k\ell}) = \mathbb{E}(Q^{0\mathsf{T}}_\ell Q^+_{k\ell})$, $\quad \mathbb{E}(P^{0\mathsf{T}}_\ell P^0_\ell) = \mathbb{E}(Q^{0\mathsf{T}}_\ell Q^0_\ell)$, from which we conclude

$$\mathrm{Cov}(P^0_\ell, P^+_{k\ell}) = \mathrm{Cov}(Q^0_\ell, Q^+_{k\ell}) =: \mathbf{K}^+_{k\ell}, \tag{B.35}$$

where the last step follows from the definition of $\mathbf{K}^+_{k\ell}$ in line 20 of Algorithm 11. Finally, we observe that for $0 \le j \le k$

$$\mathbb{E}(P^{+\mathsf{T}}_{k\ell} Q^-_{j,\ell+1}) \overset{(a)}{=} \beta^{0\mathsf{T}}_\ell \mathbb{E}(P^{0\mathsf{T}}_\ell Q^-_{j,\ell+1}) + \sum_{i=0}^{k-1} \beta^{+\mathsf{T}}_{i\ell} \mathbb{E}(P^{+\mathsf{T}}_{i\ell} Q^-_{j,\ell+1}) + \mathbb{E}(U^\mathsf{T}_{k\ell} Q^-_{j,\ell+1}) \overset{(b)}{=} \mathbf{0}, \tag{B.36}$$

where (a) follows from (B.34) and, in (b), we used the fact that $\mathbb{E}(P^{0\mathsf{T}}_\ell Q^-_{j,\ell+1}) = \mathbf{0}$ and $\mathbb{E}(P^{+\mathsf{T}}_{i\ell} Q^-_{j,\ell+1}) = \mathbf{0}$ since (B.15) is true for $i \le k$–1 corresponding to $\mathcal{H}^-_{k,\ell+1}$ and $\mathbb{E}(U^\mathsf{T}_{k\ell} Q^-_{j,\ell+1}) = \mathbf{0}$ since $U_{k\ell}$ is independent of $\overline{\mathfrak{G}}^+_{k\ell}$, and $Q^-_{j,\ell+1}$ is $\overline{\mathfrak{G}}^+_{k\ell}$ measurable. Thus, with (B.35) and (B.36), we have proven all the correlations in (B.15) corresponding to $\mathcal{H}^+_{k,\ell+1}$.

Next, we prove the convergence of the parameter lists $\Upsilon^+_{k,\ell+1}$ to $\overline{\Upsilon}^+_{k,\ell+1}$. Since $\Upsilon^+_{k\ell} \to \overline{\Upsilon}^+_{k\ell}$ due to hypothesis $\mathcal{H}^+_{k\ell}$, and $\varphi^+_{k,\ell+1}(\cdot)$ is uniformly Lipschitz continuous, we have that

$\lim_{N\to\infty}\mu^+_{k,\ell+1}$ from line 17 in Algorithm 10 converges almost surely as

$$\lim_{N\to\infty}\left\langle \boldsymbol{\varphi}^+_{k,\ell+1}(\mathbf{p}^0_\ell,\mathbf{p}^+_{k\ell},\mathbf{q}^-_{k,\ell+1},\mathbf{w}_{\ell+1},\overline{\Upsilon}^+_{k\ell})\right\rangle = \mathbb{E}\left[\varphi^+_{k,\ell+1}(P^0_\ell,P^+_{k\ell},Q^-_{k,\ell+1},W_{\ell+1},\overline{\Upsilon}^+_{k\ell})\right] = \overline{\mu}^+_{k,\ell+1}, \tag{B.37}$$

where $\overline{\mu}^+_{k,\ell+1}$ is the value in line 17 in Algorithm 11. Since $T^+_{k,\ell+1}(\cdot)$ is continuous, we have that $\lambda^+_{k,\ell+1}$ in line 18 in Algorithm 10 converges as $\lim_{N\to\infty}\lambda^+_{k,\ell+1} = T^+_{k,\ell+1}(\overline{\mu}^+_{k,\ell+1},\overline{\Upsilon}^+_{k\ell}) =: \overline{\lambda}^+_{k,\ell+1}$, from line 18 in Algorithm 11. Therefore, we have the limit

$$\lim_{N\to\infty}\Upsilon^+_{k,\ell+1} = \lim_{N\to\infty}(\Upsilon^+_{k,\ell},\lambda^+_{k,\ell+1}) = (\overline{\Upsilon}^+_{k,\ell},\overline{\lambda}^+_{k,\ell+1}) = \overline{\Upsilon}^+_{k,\ell+1}, \tag{B.38}$$

which proves the convergence of the parameter lists stated in $\mathcal{H}^+_{k,\ell+1}$. Finally, using (B.38), the empirical convergence of the matrix sequences $\mathbf{p}^0_\ell$, $\mathbf{p}^+_{k\ell}$ and $\mathbf{q}^-_{k,\ell+1}$ and the uniform Lipschitz continuity of the update function $f^+_{k,\ell+1}(\cdot)$ we obtain that $\mathbf{q}^+_{k,\ell+1}$ equals

$$\mathbf{f}^+_{k,\ell+1}(\mathbf{p}^0_\ell,\mathbf{p}^-_{k\ell},\mathbf{q}^-_{k,\ell+1},\mathbf{w}_{\ell+1},\Upsilon^+_{k,\ell+1}) \overset{2}{\Rightarrow} f^+_{k,\ell+1}(P^0_\ell,P^-_{k\ell},Q^-_{k,\ell+1},W_{\ell+1},\overline{\Upsilon}^+_{k,\ell+1}) =: Q^+_{k,\ell+1},$$

which proves the claim (B.16) for $\mathcal{H}^+_{k,\ell+1}$. This completes the proof. □

An overview of the iterates in Algorithm 10 is depicted in (TOP) and (MIDDLE) of Figure B.1. Theorem 12 shows that the rows of the iterates of Algorithm 10 converge empirically with 2nd order moments to random variables defined in Algorithm 11. The random variables defined in Algo. 11 are depicted in Figure B.1 (BOTTOM).

Algorithm 10 General Multi-Layer Matrix (Gen-ML-Mat) Recursion

```
Require: Initial matrix functions $\{\mathbf{f}^0_\ell\}$. Matrix update functions $\{\mathbf{f}^\pm_{k\ell}(\cdot)\}$. Parameter statistic functions $\{\boldsymbol{\varphi}^\pm_{k\ell}(\cdot)\}$. Parameter update functions $\{T^\pm_{k\ell}(\cdot)\}$. Orthogonal matrices $\{\mathbf{V}_\ell\}$. Perturbation variables $\{\mathbf{w}^\pm_\ell\}$. Initial matrices $\{\mathbf{q}^-_{0\ell}\}$. Initial parameter list $\Upsilon^-_{01}$.
1: // Initial Pass
2: $\mathbf{q}^0_0 = \mathbf{f}^0_0(\mathbf{w}_0), \quad \mathbf{p}^0_0 = \mathbf{V}_0\mathbf{q}^0_0$
3: for $\ell = 1, \ldots, L-1$ do
4:     $\mathbf{q}^0_\ell = \mathbf{f}^0_\ell(\mathbf{p}^0_{\ell-1}, \mathbf{w}_\ell, \Upsilon^-_{01})$
5:     $\mathbf{p}^0_\ell = \mathbf{V}_\ell\mathbf{q}^0_\ell$
6: end for
7:
8: for $k = 0, 1, \ldots$ do
9:     // Forward Pass
10:    $\lambda^+_{k0} = T^+_{k0}(\mu^+_{k0}, \Upsilon^-_{0k})$
11:    $\mu^+_{k0} = \langle \boldsymbol{\varphi}^+_{k0}(\mathbf{q}^-_{k0}, \mathbf{w}_0, \Upsilon^-_{0k}) \rangle$
12:    $\Upsilon^+_{k0} = (\Upsilon^-_{k1}, \lambda^+_{k0})$
13:    $\mathbf{q}^+_{k0} = \mathbf{f}^+_{k0}(\mathbf{q}^-_{k0}, \mathbf{w}_0, \Upsilon^+_{k0})$
14:    $\mathbf{p}^+_{k0} = \mathbf{V}_0\mathbf{q}^+_{k0}$
15:    for $\ell = 1, \ldots, L-1$ do
16:        $\lambda^+_{k\ell} = T^+_{k\ell}(\mu^+_{k\ell}, \Upsilon^+_{k,\ell-1})$
17:        $\mu^+_{k\ell} = \langle \boldsymbol{\varphi}^+_{k\ell}(\mathbf{p}^0_{\ell-1}, \mathbf{p}^+_{k,\ell-1}, \mathbf{q}^-_{k\ell}, \mathbf{w}_\ell, \Upsilon^+_{k,\ell-1}) \rangle$
18:        $\Upsilon^+_{k\ell} = (\Upsilon^+_{k,\ell-1}, \lambda^+_{k\ell})$
19:        $\mathbf{q}^+_{k\ell} = \mathbf{f}^+_{k\ell}(\mathbf{p}^0_{\ell-1}, \mathbf{p}^+_{k,\ell-1}, \mathbf{q}^-_{k\ell}, \mathbf{w}_\ell, \Upsilon^+_{k\ell})$
20:        $\mathbf{p}^+_{k\ell} = \mathbf{V}_\ell\mathbf{q}^+_{k\ell}$
21:    end for

22:    // Backward Pass
23:    $\lambda^-_{k+1,L} = T^-_{kL}(\mu^-_{kL}, \Upsilon^+_{k,L-1})$
24:    $\mu^-_{kL} = \langle \boldsymbol{\varphi}^-_{kL}(\mathbf{p}^+_{k,L-1}, \mathbf{w}_L, \Upsilon^+_{k,L-1}) \rangle$
25:    $\Upsilon^-_{k+1,L} = (\Upsilon^+_{k,L-1}, \lambda^+_{k+1,L})$
26:    $\mathbf{p}^-_{k+1,L-1} = \mathbf{f}^-_{kL}(\mathbf{p}^0_{L-1}, \mathbf{p}^+_{k,L-1}, \mathbf{w}_L, \Upsilon^-_{k+1,L})$
27:    $\mathbf{q}^-_{k+1,L-1} = \mathbf{V}^{\mathsf{T}}_{L-1}\mathbf{p}_{k+1,L-1}$
28:    for $\ell = L-1, \ldots, 1$ do
29:        $\lambda^-_{k+1,\ell} = T^-_{k\ell}(\mu^-_{k\ell}, \Upsilon^-_{k+1,\ell+1})$
30:        $\mu^-_{k\ell} = \langle \boldsymbol{\varphi}^-_{k\ell}(\mathbf{p}^0_{\ell-1}, \mathbf{p}^+_{k,\ell-1}, \mathbf{q}^-_{k+1,\ell}, \mathbf{w}_\ell, \Upsilon^-_{k+1,\ell+1}) \rangle$
31:        $\Upsilon^-_{k+1,\ell} = (\Upsilon^-_{k+1,\ell+1}, \lambda^-_{k+1,\ell})$
32:        $\mathbf{p}^-_{k+1,\ell-1} = \mathbf{f}^-_{k\ell}(\mathbf{p}^0_{\ell-1}, \mathbf{p}^+_{k,\ell-1}, \mathbf{q}^-_{k+1,\ell}, \mathbf{w}_\ell, \Upsilon^-_{k+1,\ell})$
33:        $\mathbf{q}^-_{k+1,\ell-1} = \mathbf{V}^{\mathsf{T}}_{\ell-1}\mathbf{p}^-_{k+1,\ell-1}$
34:    end for
35: end for
```

Algorithm 11 Gen-ML-Mat State Evolution (SE)

```
Require: Matrix update row-wise functions $f^0_\ell(\cdot)$ and $f^\pm_{k\ell}(\cdot)$, parameter statistic row-wise functions $\varphi^\pm_{k\ell}(\cdot)$, parameter update functions $T^\pm_{k\ell}(\cdot)$, initial parameter list limit: $\overline{\Upsilon}^-_{01}$, initial random variables $W_\ell$, $Q^-_{0\ell}$, $\ell = 0, \ldots, L-1$.
 1: // Initial pass
 2: $Q^0_0 = f^0_0(W_0, \overline{\Upsilon}^-_{01}), \quad P^0_0 \sim \mathcal{N}(0, \tau^0_0), \quad \tau^0_0 = \mathbb{E}(Q^0_0)^2$
 3: for $\ell = 1, \ldots, L-1$ do
 4:     $Q^0_\ell = f^0_\ell(P^0_{\ell-1}, W_\ell, \overline{\Upsilon}^-_{01})$
 5:     $P^0_\ell \sim \mathcal{N}(0, \tau^0_\ell), \quad \tau^0_\ell = \mathrm{Cov}(Q^0_\ell)$
 6: end for
 7:
 8: for $k = 0, 1, \ldots$ do
 9:     // Forward Pass
10:     $\overline{\lambda}^+_{k0} = T^+_{k0}(\overline{\mu}^+_{k0}, \overline{\Upsilon}^-_{0k})$
11:     $\overline{\mu}^+_{k0} = \mathbb{E}(\varphi^+_{k0}(Q^-_{k0}, W_0, \overline{\Upsilon}^-_{0k}))$
12:     $\overline{\Upsilon}^+_{k0} = (\overline{\Upsilon}^-_{k1}, \overline{\lambda}^+_{k0})$
13:     $Q^+_{k0} = f^+_{k0}(Q^-_{k0}, W_0, \overline{\Upsilon}^+_{k0})$
14:     $(P^0_0, P^+_{k0}) \sim \mathcal{N}(\mathbf{0}, \mathbf{K}^+_{k0}), \quad \mathbf{K}^+_{k0} = \mathrm{Cov}(Q^0_0, Q^+_{k0})$
15:     for $\ell = 1, \ldots, L-1$ do
16:         $\overline{\lambda}^+_{k\ell} = T^+_{k\ell}(\overline{\mu}^+_{k\ell}, \overline{\Upsilon}^+_{k,\ell-1})$
17:         $\overline{\mu}^+_{k\ell} = \mathbb{E}(\varphi^+_{k\ell}(P^0_{\ell-1}, P^+_{k,\ell-1}, Q^-_{k\ell}, W_\ell, \overline{\Upsilon}^+_{k,\ell-1}))$
18:         $\overline{\Upsilon}^+_{k\ell} = (\overline{\Upsilon}^+_{k,\ell-1}, \overline{\lambda}^+_{k\ell})$
19:         $Q^+_{k\ell} = f^+_{k\ell}(P^0_{\ell-1}, P^+_{k,\ell-1}, Q^-_{k\ell}, W_\ell, \overline{\Upsilon}^+_{k\ell})$
20:         $(P^0_\ell, P^+_{k\ell}) \sim \mathcal{N}(\mathbf{0}, \mathbf{K}^+_{k\ell}), \quad \mathbf{K}^+_{k\ell} = \mathrm{Cov}(Q^0_\ell, Q^+_{k\ell})$
21:     end for

22:     // Backward Pass
23:     $\overline{\lambda}^-_{k+1,L} = T^-_{kL}(\overline{\mu}^-_{kL}, \overline{\Upsilon}^+_{k,L-1})$
24:     $\overline{\mu}^-_{kL} = \mathbb{E}(\varphi^-_{kL}(P^0_{L-1}, P^+_{k,L-1}, W_L, \overline{\Upsilon}^+_{k,L-1}))$
25:     $\overline{\Upsilon}^-_{k+1,L} = (\overline{\Upsilon}^+_{k,L-1}, \overline{\lambda}^+_{k+1,L})$
26:     $P^-_{k+1,L-1} = f^-_{kL}(P^0_{L-1}, P^+_{k,L-1}, W_L, \overline{\Upsilon}^-_{k+1,L})$
27:     $Q^-_{k+1,L-1} \sim \mathcal{N}(0, \tau^-_{k+1,L-1}), \quad \tau^-_{k+1,L-1} = \mathrm{Cov}(P^-_{k+1,L-1})$
28:     for $\ell = L-1, \ldots, 1$ do
29:         $\overline{\lambda}^-_{k+1,\ell} = T^-_{k\ell}(\overline{\mu}^-_{k\ell}, \overline{\Upsilon}^-_{k+1,\ell+1})$
30:         $\overline{\mu}^-_{k\ell} = \mathbb{E}(\varphi^-_{k\ell}(P^0_{\ell-1}, P^+_{k,\ell-1}, Q^-_{k+1,\ell}, W_\ell, \overline{\Upsilon}^-_{k+1,\ell+1}))$
31:         $\overline{\Upsilon}^-_{k+1,\ell} = (\overline{\Upsilon}^-_{k+1,\ell+1}, \overline{\lambda}^-_{k+1,\ell})$
32:         $P^-_{k+1,\ell-1} = f^-_{k\ell}(P^0_{\ell-1}, P^+_{k,\ell-1}, Q^-_{k+1,\ell}, W_\ell, \overline{\Upsilon}^-_{k+1,\ell})$
33:         $Q^-_{k+1,\ell-1} \sim \mathcal{N}(0, \tau^-_{k+1,\ell-1}), \quad \tau^-_{k+1,\ell-1} = \mathrm{Cov}(P^-_{k+1,\ell-1})$
34:     end for
35: end for
```

Appendix C

Proofs from Chapter 5

C.1 ML-VAMP Denoisers Details

Related to $\mathbf{S}_{\text{mp}}$ and $\mathbf{s}_{\text{mp}}$ from equation (5.11), we need to define two quantities $\mathbf{s}_{\text{mp}}^{+} \in \mathbb{R}^N$ and $\mathbf{s}_{\text{mp}}^{-} \in \mathbb{R}^p$ that are zero-padded versions of the singular values $\mathbf{s}_{\text{mp}}$, so that for $n > \underset{\{}{\min} N, p\}$, we set $s_{\text{mp},n}^{\pm} = 0$. Observe that $(\mathbf{s}_{\text{mp}}^{+})^2$ are eigenvalues of $\mathbf{U}\mathbf{U}^{\mathsf{T}}$ whereas $(\mathbf{s}_{\text{mp}}^{-})^2$ are eigenvalues of $\mathbf{U}^{\mathsf{T}}\mathbf{U}$. Since $\mathbf{s}_{\text{mp}}$ empirically converges to S_{mp} as given in (5.12), the vector $\mathbf{s}_{\text{mp}}^{+}$ empirically converges to random variable S_{mp}^{+} whereas the vector $\mathbf{s}_{\text{mp}}^{-}$ empirically converges to random variable S_{mp}^{-}, where a mass is placed at 0 appropriately. Specifically, S_{mp}^{+} has a point mass of $(1-\beta)_{+}\delta_{\{0\}}$ when $\beta < 1$, whereas S_{mp}^{-} has a point mass of $(1-\frac{1}{\beta})_{+}\delta_{\{0\}}$, when $\beta > 1$. In Appendix 2.9 (eqn. (2.19)), we provide the densities over positive parts of S_{mp}^{+} and S_{mp}^{-}.

A key property of our analysis will be that the non-linear functions (5.20) and the denoisers $\mathbf{g}_{\ell}^{\pm}(\cdot)$ have simple forms.

Non-linear functions $\phi_\ell(\cdot)$: The non-linear functions all act *componentwise*. For example, for $\phi_1(\cdot)$ in (5.20), we have

$$\mathbf{z}_1 = \phi_1(\mathbf{p}_0, \mathbf{s}_{\text{tr}}) = \text{diag}(\mathbf{s}_{\text{tr}})\mathbf{p}_0 \Longleftrightarrow z_{1,n} = \phi_1(p_{0,n}, s_{\text{tr},n}),$$

where $\phi_1(\cdot)$ is the scalar-valued function,

$$\phi_1(p_0, s) = sp_0. \tag{C.1}$$

Similarly, for $\phi_2(\cdot)$,

$$\mathbf{z}_2 = \phi_2(\mathbf{p}_1, \mathbf{s}^+_{\rm mp}) \iff z_{2,n} = \phi_2(\overline{p}_{1,n}, s^+_{{\rm mp},n}), \quad n < N$$

where $\overline{\mathbf{p}}_1 \in \mathbb{R}^N$ is the zero-padded version of $\mathbf{p}_1$, and

$$\phi_2(p_1, s) = s\, p_1. \tag{C.2}$$

Finally, the function $\phi_3(\cdot)$ in (5.20) acts componentwise with

$$\phi_3(p_2, d) = \phi_{\rm out}(p_2, d). \tag{C.3}$$

<u>Input denoiser $\mathbf{g}^+_0(\cdot)$:</u> Since $F_0(\mathbf{z}_0) = F_{\rm in}(\mathbf{z}_0)$, and $F_{\rm in}(\cdot)$ given in (5.6), the denoiser (5.25a) acts *componentwise* in that,

$$\widehat{\mathbf{z}}_0 = \mathbf{g}^+_0(\mathbf{r}^-_0, \gamma^-_0) \iff \widehat{z}_{0,n} = g^+_0(r^-_{0,n}, \gamma^-_0),$$

where $g^+_0(\cdot)$ is the scalar-valued function,

$$g^+_0(r^-_0, \gamma^-_0) := \operatorname*{argmin}_{z_0}\; f_{\rm in}(z_0) + \frac{\gamma^-_0}{2}(z_0 - r^-_0)^2. \tag{C.4}$$

Thus, the vector optimization in (5.25a) reduces to a set of scalar optimizations (C.4) on each component.

<u>Output denoiser $\mathbf{g}^-_3(\cdot)$:</u> The output penalty $F_3(\mathbf{p}_2, \mathbf{y}) = F_{\rm out}(\mathbf{p}_2, \mathbf{y})$ where $F_{\rm out}(\mathbf{p}_2, \mathbf{y})$ has the separable form (5.6). Thus, similar to the case of $\mathbf{g}_0(\cdot)$, the denoiser $\mathbf{g}_3(\cdot)$ in (5.25b) also acts componentwise with the function,

$$g^-_3(r^+_2, \gamma^+_2, y) := \operatorname*{argmin}_{p_2}\; f_{\rm out}(p_2, y) + \tfrac{\gamma^+_2}{2}(p_2 - r^+_2)^2. \tag{C.5}$$

<u>Linear denoiser $\mathbf{g}^\pm_1(\cdot)$:</u> The expressions for both denoisers $g^\pm_1$ and $g^\pm_2$ are very similar and can be explained together. The penalty $F_1(\cdot)$ restricts $\mathbf{z}_1 = \mathbf{S}_{\rm tr}\mathbf{p}_0$, where $\mathbf{S}_{\rm trace}$ is a square matrix. Hence, for $\ell = 1$, the minimization in (5.27) is given by,

$$\widehat{\mathbf{p}}_0 := \operatorname*{argmin}_{\mathbf{p}_0}\; \tfrac{\gamma^+_0}{2}\|\mathbf{p}_0 - \mathbf{r}^+_0\|^2 + \tfrac{\gamma^-_1}{2}\|\mathbf{S}_{\rm tr}\mathbf{p}_0 - \mathbf{r}^-_1\|^2, \tag{C.6}$$

and $\widehat{\mathbf{z}}_1 = \mathbf{S}_{\rm tr}\widehat{\mathbf{p}}_0$. This is a simple quadratic minimization and the components of $\widehat{\mathbf{p}}_0$ and $\widehat{\mathbf{z}}_1$

are given by

$$\widehat{\boldsymbol{p}}_{0,n} = g_1^-(r_{0,n}^+, r_{1,n}^-, \gamma_0^+, \gamma_1^-, s_{\text{tr},n})$$

$$\widehat{\boldsymbol{z}}_{1,n} = g_1^+(r_{0,n}^+, r_{1,n}^-, \gamma_0^+, \gamma_1^-, s_{\text{tr},n}),$$

where

$$g_1^-(r_0^+, r_1^-, \gamma_0^+, \gamma_1^-, s) := \frac{\gamma_0^+ r_0^+ + s\gamma_1^- r_1^-}{\gamma_0^+ + s^2\gamma_1^-} \tag{C.7a}$$

$$g_1^+(r_0^+, r_1^-, \gamma_0^+, \gamma_1^-, s) := \frac{s(\gamma_0^+ r_0^+ + s\gamma_1^- r_1^-)}{\gamma_0^+ + s^2\gamma_1^-} \tag{C.7b}$$

Linear denoiser $\mathbf{g}_2^\pm(\cdot)$: This denoiser is identical to the case $\mathbf{g}_1^\pm(\cdot)$ in that we need to impose the linear constraint $\mathbf{z}_2 = \mathbf{S}_{\text{mp}}\mathbf{p}_1$. However $\mathbf{S}_{\text{mp}}$ is in general a rectangular matrix and the two resulting cases of $\beta \lesseqgtr 1$ needs to be treated separately.

Recall the definitions of vectors $\mathbf{s}_{\text{mp}}^+$ and $\mathbf{s}_{\text{mp}}^-$ at the beginning of this section. Then, for $\ell = 2$, with the penalty $F_2(\mathbf{p}_1, \mathbf{z}_2) = \delta_{\{\mathbf{z}_2 = \mathbf{S}_{\text{mp}}\mathbf{p}_1\}}$, the solution to (5.27) has components,

$$\widehat{\boldsymbol{p}}_{1,n} = g_2^-(r_{1,n}^+, r_{2,n}^-, \gamma_1^+, \gamma_2^+, s_{\text{mp},n}^-) \tag{C.8a}$$

$$\widehat{\boldsymbol{z}}_{2,n} = g_2^+(r_{1,n}^+, r_{2,n}^-, \gamma_1^+, \gamma_2^+, s_{\text{mp},n}^+), \tag{C.8b}$$

with the identical functions $g_2^- = g_1^-$ and $g_2^+ = g_1^+$ as given by (C.7a) and (C.7b). Note that in (C.8a), $n = 1, \ldots, p$ and in (C.8b), $n = 1, \ldots, N$.

C.2 State Evolution Analysis of ML-VAMP

A key property of the ML-VAMP algorithm is that its performance in the LSL can be exactly described by a *scalar equivalent system.* In the scalar equivalent system, the vector-valued outputs of the algorithm are replaced by scalar random variables representing the typical behavior of the components of the vectors in the large-scale-limit (LSL). Each of the random variables are described by a set of parameters, where the parameters are given by a set of deterministic equations called the *state evolution* or SE.

The SE for the general ML-VAMP algorithm are derived in [112] and the special case

Algorithm 12 SE for ML-VAMP for GLM Learning

```
// Initial
Initialize $\overline{\gamma}_{0\ell}^- = \gamma_{0\ell}^-$ from Algorithm 5.
$Q_{0\ell}^- \sim \mathcal{N}(0, \tau_{0\ell}^-)$ for some $\tau_{0\ell}^- > 0$ for $\ell = 0, 1, 2$
$Z_0^0 = W^0$
for $\ell = 0, \ldots, L-1$ do
    $P_\ell^0 = \mathcal{N}(0, \tau_\ell^0), \quad \tau_\ell^0 = \mathrm{var}(Z_\ell^0)$
    $Z_{\ell+1}^0 = \phi_{\ell+1}(P_\ell^0, \Xi_{\ell+1})$
end for

for $k = 0, 1, \ldots$ do
    // Forward Pass
    for $\ell = 0, \ldots, L-1$ do
        if $\ell = 0$ then
            $R_{k0}^- = Z_\ell^0 + Q_{k0}^-$
            $\widehat{Z}_{k0} = g_0^+(R_{k0}^-, \overline{\gamma}_{k0}^-)$
        else
            $R_{k,\ell-1}^+ = P_{\ell-1}^0 + P_{k,\ell-1}^+,\ R_{k\ell}^- = Z_\ell^0 + Q_{k\ell}^-$
            $\widehat{Z}_{k\ell} = g_\ell^+(R_{k,\ell-1}^+, R_{k\ell}^-, \overline{\gamma}_{k,\ell-1}^+, \overline{\gamma}_{k\ell}^-, \Xi_\ell)$
        end if
        $\overline{\alpha}_{k\ell}^+ = \mathbb{E}\partial \widehat{Z}_{k\ell} / \partial Q_{k\ell}^-$
        $Q_{k\ell}^+ = \dfrac{\widehat{Z}_{k\ell} - Z_\ell^0 - \overline{\alpha}_{k\ell}^+ Q_{k\ell}^-}{1 - \overline{\alpha}_{k\ell}^+}$
        $\overline{\gamma}_{k\ell}^+ = (\frac{1}{\overline{\alpha}_{k\ell}^+} - 1)\overline{\gamma}_{k\ell}^-$
        $(P_\ell^0, P_{k\ell}^+) \sim \mathcal{N}(0, \mathbf{K}_{k\ell}^+),\ \mathbf{K}_{k\ell}^+ = \mathrm{cov}(Z_\ell^0, Q_{k\ell}^+)$
    end for

    // Backward Pass
    for $\ell = L, \ldots, 1$ do
        if $\ell = L$ then
            $R_{k,L-1}^+ = P_{L-1}^0 + P_{k,L-1}^+$
            $\widehat{P}_{k,L-1} = g_L^-(R_{k,L-1}^+, \overline{\gamma}_{k,L-1}^+, Z_L^0)$
        else
            $R_{k,\ell-1}^+ = P_{\ell-1}^0 + P_{k,\ell-1}^+,\ R_{k+1,\ell}^- = Z_\ell^0 + Q_{k+1,\ell}^-$
            $\widehat{P}_{k,\ell-1} = g_\ell^-(R_{k,\ell-1}^+, R_{k+1,\ell}^-, \overline{\gamma}_{k,\ell-1}^+, \overline{\gamma}_{k+1,\ell}^-, \Xi_\ell)$
        end if
        $\overline{\alpha}_{k,\ell-1}^- = \mathbb{E}\partial \widehat{P}_{k,\ell-1} / \partial P_{k,\ell-1}^+$
        $P_{k+1,\ell-1}^- = \dfrac{\widehat{P}_{k,\ell-1} - P_{\ell-1}^0 - \overline{\alpha}_{k,\ell-1}^- P_{k,\ell-1}^+}{1 - \overline{\alpha}_{k,\ell-1}^-}$
        $\overline{\gamma}_{k+1,\ell-1}^- = (\frac{1}{\overline{\alpha}_{k,\ell-1}^-} - 1)\overline{\gamma}_{k,\ell-1}^+$
        $Q_{k+1,\ell-1}^- \sim \mathcal{N}(0, \tau_{k+1,\ell-1}^-),\ \tau_{k,\ell-1}^- = \mathbb{E}(P_{k+1,\ell-1}^-)^2$
    end for
end for
```

of the updates for ML-VAMP for GLM learning are shown in Algorithm 12 with details of functions $\mathbf{g}_\ell^\pm$ in Appendix C.1. We see that the SE updates in Algorithm 12 parallel those in the ML-VAMP algorithm Algo. 5, except that vector quantities such as $\widehat{\mathbf{z}}_{k\ell}$, $\widehat{\mathbf{p}}_{k\ell}$, $\mathbf{r}_{k\ell}^+$ and $\mathbf{r}_{k\ell}^-$ are replaced by scalar random variables such as $\widehat{Z}_{k\ell}$, $\widehat{P}_{k\ell}$, $R_{k\ell}^+$ and $R_{k\ell}^-$. Each of these random variables are described by the deterministic parameters such as $\mathbf{K}_{k\ell} \in \mathbb{R}^{2\times 2}_{\succ 0}$, and τ_ℓ^0, $\tau_{k\ell}^- \in \mathbb{R}_+$.

The updates in the section labeled as "Initial", provide the scalar equivalent model for the true system (5.18). In these updates, Ξ_ℓ represent the limits of the vectors $\boldsymbol{\xi}_\ell$ in (5.19). That is,

$$\Xi_1 := S_{\rm tr}, \quad \Xi_2 := S_{\rm mp}^+, \quad \Xi_3 := D. \tag{C.9}$$

Due to assumptions in Section 5.2, we have that the components of $\boldsymbol{\xi}_\ell$ converge empirically as,

$$\lim_{N\to\infty} \{\xi_{\ell,i}\} \overset{PL(2)}{=} \Xi_\ell, \tag{C.10}$$

So, the Ξ_ℓ represent the asymptotic distribution of the components of the vectors $\boldsymbol{\xi}_\ell$.

The updates in sections labeled "Forward pass" and "Backward pass" in the SE equations in Algorithm 12 parallel those in Algorithm 5. The key quantities in these SE equations are the error variables,

$$\mathbf{p}_{k\ell}^+ := \mathbf{r}_{k\ell}^+ - \mathbf{p}_\ell^0, \quad \mathbf{q}_{k\ell}^- := \mathbf{r}_{k\ell}^- - \mathbf{z}_\ell^0,$$

which represent the errors of the estimates to the inputs of the denoisers. We will also be interested in their transforms,

$$\mathbf{q}_{k\ell}^+ = \mathbf{V}_\ell^\mathsf{T} \mathbf{p}_{k,\ell+1}^+, \quad \mathbf{p}_{k\ell}^- = \mathbf{V}_\ell \mathbf{q}_{k\ell}^-.$$

The following Theorem is an adapted version of the main result from [112] to the iterates of Algorithms 5 and 12.

Theorem 13. *Consider the outputs of the ML-VAMP for GLM Learning Algorithm under the assumptions of Section 5.2. Assume the denoisers satisfy the continuity conditions in*

Assumption 1. Also, assume that the outputs of the SE satisfy

$$\overline{\alpha}^{\pm}_{k\ell} \in (0,1),$$

for all k and ℓ. Suppose Algo. 5 is initialized so that the following convergence holds

$$\lim_{N\to\infty} \{\mathbf{r}^{-}_{0\ell} - \mathbf{z}^{0}_{\ell}\} \overset{PL(2)}{=} Q^{-}_{0\ell}$$

where $(Q^{-}_{00}, Q^{-}_{01}, Q^{-}_{02})$ are independent zero-mean Gaussians, independent of $\{\Xi_\ell\}$. Then,

(a) For any fixed iteration $k \geq 0$ in the forward direction and layer $\ell = 1, \ldots, L-1$, we have that, almost surely,

$$\lim_{N\to\infty} (\gamma^{+}_{k,\ell-1}, \gamma^{-}_{k\ell}) = (\overline{\gamma}^{+}_{k,\ell-1}, \overline{\gamma}^{-}_{k\ell}), \quad \text{and,} \tag{C.11}$$

$$\lim_{N\to\infty} \{(\widehat{\mathbf{z}}^{+}_{k\ell}, \mathbf{z}^{0}_{\ell}, \mathbf{p}^{0}_{\ell-1}, \mathbf{r}^{+}_{k,\ell-1}, \mathbf{r}^{-}_{\ell})\} \overset{PL(2)}{=} (\widehat{Z}^{+}_{k\ell}, Z^{0}_{\ell}, P^{0}_{\ell-1}, R^{+}_{k,\ell-1}, R^{-}_{\ell}) \tag{C.12}$$

where the variables on the right-hand side are from the SE equations (C.11) *and* (C.12) *are the outputs of the SE equations in Algorithm 12. Similar equations hold for $\ell = 0$ with the appropriate variables removed.*

(b) Similarly, in the reverse direction, For any fixed iteration $k \geq 0$ and layer $\ell = 1, \ldots, L-2$, we have that, almost surely,

$$\lim_{N\to\infty} (\gamma^{+}_{k,\ell-1}, \gamma^{-}_{k+1,\ell}) = (\overline{\gamma}^{+}_{k,\ell-1}, \overline{\gamma}^{-}_{k+1,\ell}), \quad \text{and} \tag{C.13}$$

$$\lim_{N\to\infty} \{(\widehat{\mathbf{p}}^{+}_{k+1,\ell-1}, \mathbf{z}^{0}_{\ell}, \mathbf{p}^{0}_{\ell-1}, \mathbf{r}^{+}_{k,\ell-1}, \mathbf{r}^{-}_{k+1,\ell})\} \overset{PL(2)}{=} (\widehat{P}^{+}_{k+1,\ell-1}, Z^{0}_{\ell}, P^{0}_{\ell-1}, R^{+}_{k,\ell-1}, R^{-}_{k+1,\ell}). \tag{C.14}$$

Furthermore, $(P^{0}_{\ell-1}, P^{+}_{k\ell-1})$ and $Q^{-}_{k\ell}$ are independent.

Proof. This is a direct application of Theorem 3 from [114] to the iterations of Algorithm 5. The convergence result in [114] requires the uniform Lipschitz continuity of functions $\mathbf{g}^{\pm}_{\ell}(\cdot)$. Assumption 1 provides the required uniform Lipschitz continuity assumption on $\mathbf{g}^{+}_{0}(\cdot)$ and $\mathbf{g}^{-}_{3}(\cdot)$. For the "middle" layers, $\ell = 1, 2$, the denoisers $\mathbf{g}^{\pm}_{\ell}(\cdot)$ are linear and the uniform continuity assumption is valid since we have made the additional assumption that the terms

$\mathbf{s}_{\rm tr}$ and $\mathbf{s}_{\rm mp}$ are bounded almost surely. □

A key use of the Theorem is to compute asymptotic empirical limits. Specifically, for a componentwise function $\psi(\cdot)$, let $\langle\psi(\mathbf{x})\rangle$ denotes the average $\frac{1}{N}\sum_{n=1}^{N}\psi(x_n)$ The above theorem then states that for any componentwise pseudo-Lipschitz function $\psi(\cdot)$ of order 2, as $N\to\infty$, we have the following two properties

$$\begin{aligned}
&\lim_{N\to\infty}\left\langle\psi(\widehat{\mathbf{z}}^{+}_{k\ell},\mathbf{z}^{0}_{\ell},\mathbf{p}^{0}_{\ell-1},\mathbf{r}^{+}_{k,\ell-1},\mathbf{r}^{-}_{\ell})\right\rangle \\
&\qquad=\mathbb{E}\,\psi(\widehat{Z}^{+}_{k\ell},Z^{0}_{\ell},P^{0}_{\ell-1},R^{+}_{k,\ell-1},R^{-}_{\ell}) \\
&\lim_{N\to\infty}\left\langle\psi(\widehat{\mathbf{p}}^{+}_{k+1,\ell-1},\mathbf{z}^{0}_{\ell},\mathbf{p}^{0}_{\ell-1},\mathbf{r}^{+}_{k,\ell-1},\mathbf{r}^{-}_{k+1,\ell)})\right\rangle \\
&\qquad=\mathbb{E}\,\psi(\widehat{P}^{+}_{k+1,\ell-1},Z^{0}_{\ell},P^{0}_{\ell-1},R^{+}_{k,\ell-1},R^{-}_{k+1,\ell}).
\end{aligned}$$

That is, we can compute the empirical average over components with the expected value of the random variable limit. This convergence is key to proving Theorem 10.

C.3 Empirical Convergence of Fixed Points

A consequence of Assumption 2 is that we can take the limit $k\to\infty$ of the random variables in the SE algorithm. Specifically, let $\mathbf{x}_k=\mathbf{x}_k(N)$ be any set of d outputs from the ML-VAMP for GLM Learning Algorithm under the assumptions of Theorem 13. Under Assumption 2, for each N, there exists a vector

$$\mathbf{x}(N)=\lim_{k\to\infty}\mathbf{x}_k(N), \tag{C.15}$$

representing the limit over k. For each k, Theorem 13 shows there also exists a random vector limit,

$$\lim_{N\to\infty}\{\mathbf{x}_{k,i}(N)\}\overset{PL(2)}{=}X_k, \tag{C.16}$$

representing the limit over N. The following proposition shows that we can take the limits of the random variables X_k.

Proposition 1. *Consider the outputs of the ML-VAMP for GLM Learning Algorithm under* *the assumptions of Theorem 13 and Assumption 2. Let* $\mathbf{x}_k = \mathbf{x}_k(N)$ *be any set of* d *outputs from the algorithm and let* $\mathbf{x}(N)$ *be its limit from* (C.15) *and let* X_k *be the random variable limit* (C.16)*. Then, there exists a random variable* $X \in \boldsymbol{R}^d$ *such that, for any pseudo-Lipschitz continuous* $f : \boldsymbol{R}^d \to \boldsymbol{R}$*,*

$$\lim_{k\to\infty} \mathbb{E}f(X_k) = \mathbb{E}f(X) = \lim_{N\to\infty} \frac{1}{N}\sum_{i=1}^{N} f(x_i(N)). \tag{C.17}$$

In addition, the SE parameter limits $\overline{\alpha}^{\pm}_{k\ell}$ *and* $\overline{\gamma}^{\pm}_{k\ell}$ *converge to limits,*

$$\lim_{k\to\infty} \overline{\alpha}^{\pm}_{k\ell} = \overline{\alpha}^{\pm}_{\ell}, \quad \lim_{k\to\infty} \overline{\gamma}^{\pm}_{k\ell} = \overline{\gamma}^{\pm}_{\ell}. \tag{C.18}$$

The proposition shows that, under the convergence assumption, Assumption 2, we can take the limits as $k \to \infty$ of the random variables from the SE. To prove the proposition we first need the following simple lemma.

Lemma 12. *If* α_N *and* $\beta_k \in \boldsymbol{R}$ *are sequences such that*

$$\lim_{k\to\infty}\lim_{N\to\infty} |\alpha_N - \beta_k| = 0, \tag{C.19}$$

then, there exists a constant C *such that,*

$$\lim_{N\to\infty} \alpha_N = \lim_{k\to\infty} \beta_k = C. \tag{C.20}$$

In particular, the two limits in (C.20) *exist.*

Proof. For any $\epsilon > 0$, the limit (C.19) implies that there exists a $k_\epsilon (\uparrow \infty \text{ as } \epsilon \downarrow 0)$ such that for all $k > k_\epsilon$,

$$\lim_{N\to\infty} |\alpha_N - \beta_k| < \epsilon,$$

from which we can conclude,

$$\liminf_{N\to\infty} \alpha_N > \beta_k - \epsilon$$

for all $k > k_\epsilon$. Therefore,

$$\liminf_{N\to\infty} \alpha_N \geq \sup_{k\geq k_\epsilon} \beta_k - \epsilon.$$

Since this is true for all $\epsilon > 0$, it follows that

$$\liminf_{N\to\infty} \alpha_N \geq \limsup_{k\to\infty} \beta_k. \tag{C.21}$$

Similarly, $\limsup_{N\to\infty} \alpha_N \leq \inf_{k>k_\epsilon} \beta_k + \epsilon$, whereby

$$\limsup_{N\to\infty} \alpha_N \leq \liminf_{k\to\infty} \beta_k. \tag{C.22}$$

Equations (C.21) and (C.22) together show that the limits in (C.20) exists and are equal. $\square$

Proof of Proposition 1 Let $f : \boldsymbol{R}^d \to \boldsymbol{R}$ be any pseudo-Lipschitz function of order 2, and define,

$$\alpha_N = \frac{1}{N}\sum_{i=1}^{N} f(x_i(N)), \quad \beta_k = \mathbb{E}f(X_k). \tag{C.23}$$

Their difference can be written as,

$$\alpha_N - \beta_k = A_{N,k} + B_{N,k}, \tag{C.24}$$

where

$$A_{N,k} := \frac{1}{N}\sum_{i=1}^{N} f(\mathbf{x}_i(N)) - f(\mathbf{x}_{k,i}(N)), \tag{C.25}$$

$$B_{N,k} := \frac{1}{N}\sum_{i=1}^{N} f(\mathbf{x}_{k,i}(N)) - \mathbb{E}f(X_k). \tag{C.26}$$

Since $\{x_{k,i}(N)\}$ converges $PL(2)$ to X_k, we have,

$$\lim_{N\to\infty} B_{N,k} = 0. \tag{C.27}$$

For the term $A_{N,k}$,

$$\begin{aligned}
|A_{N,k}| &\overset{\text{(a)}}{\leq} \lim_{N\to\infty} \frac{1}{N}\sum_{i=1}^{N} |f(\mathbf{x}_i(N)) - f(\mathbf{x}_{k,i}(N))| \\
&\overset{\text{(b)}}{\leq} \lim_{N\to\infty} \frac{C}{N}\sum_{i=1}^{N} a_{ki}(N)(1 + a_{ki}(N)) \\
&\overset{\text{(c)}}{\leq} C \lim_{N\to\infty} \sqrt{\frac{1}{N}\sum_{i=1}^{N} a_{ki}^2(N) + \frac{1}{N}\sum_{i=1}^{N} a_{ki}^2(N)} \\
&= C \lim_{N\to\infty} \epsilon_k(N)(1 + \epsilon_k(N)),
\end{aligned} \tag{C.28}$$

where (a) follows from applying the triangle inequality to the definition of $A_{N,k}$ in (C.25); (b) follows from the definition of pseudo-Lipschitz continuity in Definition 1, $C > 0$ is the Lipschitz contant and

$$a_{ki}(N) := \|\mathbf{x}_{k,i}(N) - \mathbf{x}_i(N)\|_2,$$

and (c) follows from the RMS-AM inequality:

$$\left(\frac{1}{N}\sum_{i=1}^{N} a_{ki}(N)\right)^2 \leq \frac{1}{N}\sum_{i=1}^{N} a_{ki}^2(N) =: \epsilon_k^2(N).$$

By (5.29), we have that,

$$\lim_{k\to\infty}\lim_{N\to\infty} \epsilon_k(N) = 0.$$

Hence, from (C.28), it follows that,

$$\lim_{k\to\infty}\lim_{N\to\infty} A_{N,k} = 0. \tag{C.29}$$

Substituting (C.27) and (C.29) into (C.24) show that α_N and β_k satisfy (C.19). Therefore, applying Lemma 12 we have that for any pseudo-Lipschitz function $f(\cdot)$, there exists a limit $\Phi(f)$ such that,

$$\lim_{N\to\infty} \frac{1}{N}\sum_{i=1}^{N} f(x_i(N)) = \lim_{k\to\infty} \mathbb{E}f(X_k) = \Phi(f). \tag{C.30}$$

In particular, the two limits in (C.30) exists. When restricted to the continuous, bounded functions with the $\|f\|_\infty$ norm, it is easy verified that $\Phi(f)$ is a positive, linear, bounded function of f, with $\Phi(1) = 1$. Therefore, by the Riesz representation theorem, there exists a random variable X such that $\Phi(f) = \mathbb{E}f(X)$. This fact in combination with (C.30) shows (C.17).

It remains to prove the parameter limits in (C.18). We prove the result for the parameter $\overline{\alpha}_{k\ell}^{+}$. The proof for the other parameters are similar. Using Stein's lemma, it is shown in [112] that

$$\overline{\alpha}_{k\ell}^{+} = \frac{\mathbb{E}(\widehat{Z}_{k\ell} Q_{k\ell}^{-})}{\mathbb{E}(Q_\ell^{-})^2}. \tag{C.31}$$

Since the numerator and denominator of (C.31) are $PL(2)$ functions we have that the limit,

$$\overline{\alpha}_\ell^+ := \lim_{k\to\infty} \overline{\alpha}_{k\ell}^+ = \lim_{k\to\infty} \frac{\mathbb{E}(\widehat{Z}_{k\ell} Q_{k\ell}^-)}{\mathbb{E}(Q_{k\ell}^-)^2}$$
$$= \frac{\mathbb{E}(\widehat{Z}_\ell Q_\ell^-)}{\mathbb{E}(Q_\ell^-)^2}, \tag{C.32}$$

where $\widehat{Z}_\ell$ and Q_ℓ^- are the limits of $\widehat{Z}_{k\ell}$ and $Q_{k\ell}^-$. This completes the proof. □

C.4 Proof of Theorem 10

From Assumption 2, we know that for every N, every group of vectors $\mathbf{x}_k$ converge to limits, $\mathbf{x} := \lim_{k\to\infty} \mathbf{x}_k$. The parameters, $\gamma_{k\ell}^\pm$, also converge to limits $\overline{\gamma}_\ell^\pm := \lim_{k\to\infty} \gamma_{k\ell}^\pm$ for all ℓ. By the continuity assumptions on the functions $\mathbf{g}_\ell^\pm(\cdot)$, the limits $\mathbf{x}$ and $\overline{\gamma}_\ell^\pm$ are fixed points of the algorithms.

A proof similar to that in [114] shows that the fixed points $\widehat{\mathbf{z}}_\ell$ and $\widehat{\mathbf{p}}_\ell$ satisfy the KKT condition of the constrained optimization (5.22). This proves part (a).

The estimate $\widehat{\mathbf{w}}$ is the limit,

$$\widehat{\mathbf{w}} = \widehat{\mathbf{z}}_0 = \lim_{k\to\infty} \widehat{\mathbf{z}}_{k0}.$$

Also, the true parameter is $\mathbf{z}_0^0 = \mathbf{w}^0$. By Proposition 1, we have that the $PL(2)$ limits of these variables are

$$\lim_{N\to\infty} \{(\widehat{\mathbf{w}}, \mathbf{w}_0)\} \overset{PL(2)}{=} (\widehat{W}, W_0) := (\widehat{Z}_0, Z_0^0).$$

From line 15 of the SE Algorithm 12, we have

$$\widehat{W} = \widehat{Z}_0 = g_0^+(R_0^-, \overline{\gamma}_0^-) = \text{prox}_{f_{\text{in}}/\overline{\gamma}_0^-}(W^0 + Q_0^-).$$

This proves part (b).

To prove part (c), we use the limit

$$\lim_{N\to\infty} \{p_{0,n}^0, \widehat{p}_{0,n}\} \overset{PL(2)}{=} (P_0^0, \widehat{P}_0). \tag{C.33}$$

Since the fixed points are critical points of the constrained optimization (5.22), $\widehat{\mathbf{p}}_0 = \mathbf{V}_0 \widehat{\mathbf{w}}$.

We also have $\mathbf{p}_0^0 = \mathbf{V}_0\mathbf{w}^0$. Therefore,

$$\begin{aligned}\left[z_{\rm ts}^{(N)}\ \widehat{z}_{\rm ts}^{(N)}\right] &:= \mathbf{u}^{\mathsf{T}}\,\mathrm{Diag}(\mathbf{s}_{\rm ts})\mathbf{V}_0[\mathbf{w}^0\ \widehat{\mathbf{w}}] \\ &= \mathbf{u}^{\mathsf{T}}\,\mathrm{Diag}(\mathbf{s}_{\rm ts})[\mathbf{p}_0^0\ \widehat{\mathbf{p}}_0]. \end{aligned} \tag{C.34}$$

Here, (N) in the subscript denotes the dependence on N. Since $\mathbf{u} \sim \mathcal{N}(0, \frac{1}{p}\mathbf{I})$, $[z_{\rm ts}^{(N)}\ \widehat{z}_{\rm ts}^{(N)}]$ is a zero-mean bivariate Gaussian with covariance matrix

$$\mathbf{M}^{(N)} = \tfrac{1}{p}\sum_{n=1}^{p}\begin{bmatrix} s_{{\rm ts},n}^2 p_{0,n}^0 p_{0,n}^0 & s_{{\rm ts},n}^2 p_{0,n}^0 \widehat{p}_{0,n} \\ s_{{\rm ts},n}^2 p_{0,n}^0 \widehat{p}_{0,n} & s_{{\rm ts},n}^2 \widehat{p}_{0,n}\widehat{p}_{0,n}\end{bmatrix}$$

The empirical convergence (C.33) yields the following limit,

$$\lim_{N\to\infty}\mathbf{M}^{(N)} = \mathbf{M} := \mathbb{E}\, S_{\rm ts}^2 \begin{bmatrix} P_0^0 P_0^0 & P_0^0 \widehat{P}_0 \\ P_0^0 \widehat{P}_0 & \widehat{P}_0 \widehat{P}_0 \end{bmatrix}. \tag{C.35}$$

It suffices to show that the distribution of $[z_{\rm ts}^{(N)}\ \widehat{z}_{\rm ts}^{(N)}]$ converges to the distribution of $[Z_{\rm ts}\ \widehat{Z}_{\rm ts}]$ in the Wasserstein-2 metric as $N \to \infty$. (See the discussion in Appendix 2.6.1 on the equivalence of convergence in Wasserstein-2 metric and PL(2) convergence.)

Now, Wassestein-2 distance between between two probability measures ν_1 and ν_2 is defined as

$$W_2(\nu_1,\nu_2) = \left(\inf_{\gamma\in\Gamma}\mathbb{E}_\gamma \left\|X_1 - X_2\right\|^2\right)^{1/2}, \tag{C.36}$$

where Γ is the set of probability distributions on the product space with marginals consistent with ν_1 and ν_2. For Gaussian measures $\nu_1 = \mathcal{N}(\mathbf{0},\Sigma_1)$ and $\nu_2 = \mathcal{N}(\mathbf{0},\Sigma_2)$ we have [52]

$$W_2^2(\nu_1,\nu_2) = \mathrm{trace}(\Sigma_1 - 2(\Sigma_1^{1/2}\Sigma_2\Sigma_1^{1/2})^{1/2} + \Sigma_2).$$

Therefore, for Gaussian distributions $\nu_1^{(N)} = \mathcal{N}(\mathbf{0},\mathbf{M}^{(N)})$, and $\nu_2 = \mathcal{N}(\mathbf{0},\mathbf{M})$, the convergence (C.35) implies $W_2(\nu_1^{(N)},\nu_2) \to 0$, i.e., convergence in Wasserstein-2 distance. Hence,

$$(z_{\rm ts}^{(N)}, \widehat{z}_{\rm ts}^{(N)}) \xrightarrow{W_2} (Z_{\rm ts}, \widehat{Z}_{\rm ts}) \sim \mathcal{N}(0,\mathbf{M}),$$

where $\mathbf{M}$ is the covariance matrix in (C.35). Hence the convergence holds in the PL(2) sense (see discussion in Appendix 2.6.1 on the equivalence of convergence in W_2 and PL(2) convergence).

Hence the asymptotic generalization error (5.17) is

$$\begin{aligned}\mathcal{E}_{\rm ts} &:= \lim_{N\to\infty} \mathbb{E} f_{\rm ts}(\widehat{y}_{\rm ts}, y_{\rm ts}) \\ &\overset{(a)}{=} \lim_{N\to\infty} \mathbb{E} f_{\rm ts}(\phi_{\rm out}(z_{\rm ts}^{(N)}, D), \phi(\widehat{z}_{\rm ts}^{(N)})) \\ &\overset{(b)}{=} \mathbb{E} f_{\rm ts}(\phi_{\rm out}(Z_{\rm ts}, D), \phi(\widehat{Z}_{\rm ts})), \end{aligned} \tag{C.37}$$

where (a) follows from (5.3); and step (b) follows from continuity assumption in Assumption 1(b) along with the definition of PL(2) convergence in Def. 3. This proves part (c).

C.5 Formula for M

For the special cases in the next Appendix, it is useful to derive expressions for the entries the covariance matrix $\mathbf{M}$ in (C.35). For the term m_{11},

$$m_{11} = \mathbb{E}\, S_{\rm ts}^2 (P_0^0)^2 = \mathbb{E}\, S_{\rm ts}^2 \mathbb{E}(P_0^0)^2 = \mathbb{E}\, S_{\rm ts}^2 \cdot k_{11}, \tag{C.38}$$

where we have used the fact that $P_0^0 \perp\!\!\!\perp (S_{\rm ts}, S_{\rm trace})$. Next, $m_{12} = \mathbb{E}\, S_{\rm ts}^2\ P_0^0 \widehat{P}_0$. where,

$$\begin{aligned}\widehat{P}_0 &= g_1^-(P_0^0 + P_0^+, Z_1^0 + Q_1^-, \overline{\gamma}_0^+, \overline{\gamma}_1^-, S_{\rm tr}^-) \\ &= \frac{\overline{\gamma}_0^+ P_0^+ + S_{\rm tr}\overline{\gamma}_1^- Q_1^-}{\overline{\gamma}_0^+ + S_{\rm tr}^2 \overline{\gamma}_1^-} + P_0^0, \end{aligned} \tag{C.39}$$

where (P_0^0, P_0^+, Q_0^-) are independent of $(S_{\rm trace}, S_{\rm ts})$. Hence,

$$\begin{aligned} m_{12} &= \mathbb{E}\, S_{\rm ts}^2 \cdot \mathbb{E}(P_0^0)^2 + \mathbb{E} \frac{S_{\rm ts}^2 \overline{\gamma}_0^+}{\overline{\gamma}_0^+ + S_{\rm tr}^2 \overline{\gamma}_1^-} \mathbb{E}[P_0^0 P_0^+] \\ &= m_{11} + \mathbb{E}\left(\frac{S_{\rm ts}^2 \overline{\gamma}_0^+}{\overline{\gamma}_0^+ + S_{\rm trace}^2 \overline{\gamma}_1^-}\right) \cdot k_{12}, \end{aligned} \tag{C.40}$$

since $\mathbb{E}[P_0^0 Q_1^-] = 0$ and $\mathbf{K}_0^+$ is the covariance matrix of (P_0^0, P_0^+) from line 23.

Finally for m_{22} we have,

$$\begin{aligned}
m_{22} &= \mathbb{E}\, S_{\text{ts}}^2 \widehat{P}_0 \widehat{P}_0 \\
&= \mathbb{E}\left(\frac{S_{\text{ts}}\overline{\gamma}_0^+}{\overline{\gamma}_0^+ + S_{\text{tr}}^2\overline{\gamma}_1^-}\right)^2 \mathbb{E}(P_0^+)^2 \\
&+ \mathbb{E}\left(\frac{S_{\text{ts}}S_{\text{trace}}\overline{\gamma}_1^-}{\overline{\gamma}_0^+ + S_{\text{tr}}^2\overline{\gamma}_1^-}\right)^2 \mathbb{E}(Q_1^-)^2 \\
&+ \mathbb{E}\, S_{\text{ts}}^2 \mathbb{E}(P_0^0)^2 + 2\mathbb{E}\frac{\overline{\gamma}_0^+ S_{\text{ts}}^2}{\overline{\gamma}_0^+ + \overline{\gamma}_1^- S_{\text{trace}}^2} \cdot \mathbb{E}\, P_0^0 P_0^+ \\
&= k_{22}\mathbb{E}\left(\frac{S_{\text{ts}}\overline{\gamma}_0^+}{\overline{\gamma}_0^+ + S_{\text{tr}}^2\overline{\gamma}_1^-}\right)^2 \\
&+ \tau_1^- \mathbb{E}\left(\frac{S_{\text{ts}}S_{\text{trace}}\overline{\gamma}_1^-}{\overline{\gamma}_0^+ + S_{\text{tr}}^2\overline{\gamma}_1^-}\right)^2 - m_{11} + 2m_{12}. \qquad \text{(C.41)}
\end{aligned}$$

C.6 Special Cases

C.6.1 Linear Output with Square Error

In this section we examine a few special cases of the GLM problem (5.2). We first consider a linear output with additive Gaussian noise and a squared error training and test loss. Specifically, consider the model,

$$\mathbf{y} = \mathbf{X}\mathbf{w}^0 + \mathbf{d} \qquad \text{(C.42)}$$

We consider estimates of $\mathbf{w}^0$ such that:

$$\widehat{\mathbf{w}} = \underset{\mathbf{w}}{\operatorname{argmin}}\ \tfrac{1}{2}\left\|\mathbf{y} - \mathbf{X}\mathbf{w}\right\|^2 + \tfrac{\lambda}{2\beta}\left\|\mathbf{w}\right\|^2 \qquad \text{(C.43)}$$

The factor β is added above since the two terms scale with a ratio of β. It does not change analysis. Consider the ML-VAMP GLM learning algorithm applied to this problem. The following corollary follows from the Main result in Theorem 10.

Corollary 1 (Squared error)**.** *For linear regression,* i.e., $\phi(t) = t$, $\phi_{\text{out}}(t, d) = t + d$, $f_{\text{ts}}(y, \widehat{y}) = (y_{\text{ts}} - \widehat{y}_{\text{ts}})^2$, $F_{\text{out}}(\boldsymbol{p}_2) = \frac{1}{N}\left\|\boldsymbol{y} - \boldsymbol{p}_2\right\|^2$, we have

$$\mathcal{E}_{\text{ts}}^{\mathsf{LR}} = \mathbb{E}\left(\tfrac{\overline{\gamma}_0^+ S_{\text{ts}}}{\overline{\gamma}_0^+ + S_{\text{trace}}^2\overline{\gamma}_1^-}\right)^2 k_{22} + \mathbb{E}\left(\tfrac{\overline{\gamma}_1^- S_{\text{trace}} S_{\text{ts}}}{\overline{\gamma}_0^+ + S_{\text{trace}}^2\overline{\gamma}_1^-}\right)^2 \tau_1^- + \sigma_d^2.$$

The quantities k_{22}, $\tau_1^-, \overline{\gamma}_0^+, \overline{\gamma}_1^-$ depend on the choice of regularizer λ and the covariance between features.

Proof. This follows directly from the following observation:

$$\begin{aligned}\mathcal{E}_{\text{ts}}^{\mathsf{SLR}} &= \mathbb{E}(Z_{\text{ts}} + D - \widehat{Z}_{\text{ts}})^2 = \mathbb{E}(Z_{\text{ts}} - \widehat{Z}_{\text{ts}})^2 + \mathbb{E}\, D^2 \\ &= m_{11} + m_{22} - 2m_{12} + \sigma_d^2.\end{aligned}$$

Substituting equation (C.41) proves the claim. □

C.6.2 Ridge Regression with i.i.d. Covariates

We next the special case when the input features are independent, i.e., (C.43) where rows of $\mathbf{X}$ corresponding to the training data has i.i.d Gaussian features with covariance $\mathbf{P}_{\text{train}} = \frac{\sigma_{\text{tr}}^2}{p}\mathbf{I}$ and $S_{\text{tr}} = \sigma_{\text{tr}}$.

Although the solution to (C.43) exists in closed form $(\mathbf{X}^{\mathsf{T}}\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^{\mathsf{T}}\mathbf{y}$, we can study the effect of the regularization parameter λ on the generalization error $\mathcal{E}_{\text{ts}}$ as detailed in the result below.

Corollary 2. *Consider the ridge regression problem* (C.43) *with regularization parameter* $\lambda > 0$*. For the squared loss* i.e., $f_{\text{ts}}(y, \widehat{y}) = (y - \widehat{y})^2$, i.i.d Gaussian features without train-test mismatch, i.e., $S_{\text{trace}} = S_{\text{ts}} = \sigma_{\text{trace}}$, the generalization error $\mathcal{E}_{\text{ts}}^{\mathsf{RR}}$ is given by Corollary 1, with constants

$$k_{22} = \mathbb{V}\,(W^0), \qquad \overline{\gamma}_0^+ = \lambda/\beta,$$

$$\overline{\gamma}_1^- = \begin{cases} \frac{1}{G} - \frac{\lambda}{\sigma_{\text{trace}}^2} & \beta < 1 \\ \frac{\frac{\lambda}{\sigma_{\text{trace}}^2\beta}\left(\frac{1}{G} - \frac{\lambda}{\sigma_{\text{trace}}^2\beta}\right)}{\frac{\beta-1}{G} + \frac{\lambda}{\sigma_{\text{trace}}^2\beta}} & \beta > 1 \end{cases}$$

where $G = G_{\text{mp}}(-\frac{\lambda}{\sigma_{\text{tr}}^2\beta})$, with G_{mp} given in Appendix 2.9, and $\tau_1^- = \mathbb{E}(P_1^-)^2$ where P_1^- is given in equation (C.55) in the proof.

Proof of Corollary 2. We are interested in identifying the following constants appearing in

Corollary 1:

$$\mathbf{K}_0^+, \tau_1^-, \overline{\gamma}_0^+, \overline{\gamma}_1^-.$$

These quantities are obtained as fixed points of the State Evolution Equations in Algo. 12. We explain below how to obtain expressions for these constants. Since these are fixed points we ignore the subscript k corresponding to the iteration number in Algo. 12.

In the case of problem (C.43), the maps $\mathrm{prox}_{f_{\mathrm{in}}}$ and $\mathrm{prox}_{f_{\mathrm{out}}}$, i.e., g_0^+ and g_3^- respectively, can be expressed as closed-form formulae. This leads to simplification of the SE equations as explained below.

We start by looking at the *forward pass* (finding quantities with superscript '+') of Algorithm 12 for different layers and then the *backward pass* (finding quantities with superscript '-') to get the parameters $\{\mathbf{K}_\ell^+, \tau_\ell^-, \overline{\alpha}_\ell^\pm, \overline{\gamma}_\ell^\pm\}$ for $\ell = 0, 1, 2$.

To begin with, notice that $f_{\mathrm{in}}(w) = \frac{\lambda}{2}w^2$, and therefore the denoiser $g_0^+(\cdot)$ in (C.4) is simply,

$$g_0^+(r_0^-, \gamma_0^-) = \tfrac{\gamma_0^-}{\gamma_0^- + \lambda/\beta} r_0^-, \quad \text{and} \quad \tfrac{\partial g_0^+}{\partial r_0^-} = \tfrac{\gamma_0^-}{\gamma_0^- + \lambda/\beta}$$

Using the random variable R_0^- and substituting in the expression of the denoiser to get $\widehat{Z}_0$, we can now calculate $\overline{\alpha}_0^+$ using lines 20 and 22,

$$\overline{\alpha}_0^+ = \tfrac{\overline{\gamma}_0^-}{\overline{\gamma}_0^- + \lambda/\beta}, \qquad \overline{\gamma}_0^+ = \lambda/\beta. \tag{C.44}$$

Similarly, we have $f_{\mathrm{out}}(p_2) = \frac{1}{2}(p_2 - y)^2$, whereby the output denoiser $g_3^-(\cdot)$ in the last layer for ridge regression is given by,

$$g_3^-(r_2^+, \gamma_2^+, y) = \frac{\gamma_2^+ r_2^+ + y}{\gamma_2^+ + 1}. \tag{C.45}$$

By substituting this denoiser in line 30 of the algorithm we get $\widehat{P}_2^-$ and thus, following the lines 35-38 of the algorithm we have

$$\overline{\alpha}_2^- = \tfrac{\overline{\gamma}_2^+}{\overline{\gamma}_2^+ + 1}, \qquad \text{whereby} \quad \overline{\gamma}_2^- = 1. \tag{C.46}$$

Having identified these constants $\overline{\alpha}_0^+, \overline{\gamma}_0^+, \overline{\alpha}_2^-, \overline{\gamma}_2^-$, we will now sequentially identify the quantities

$$(\overline{\alpha}_0^+, \overline{\gamma}_0^+) \to \mathbf{K}_0^+ \to (\overline{\alpha}_1^+, \overline{\gamma}_1^+) \to \mathbf{K}_1^+ \to (\overline{\alpha}_2^+, \overline{\gamma}_2^+) \to \mathbf{K}_2^+$$

in the forward pass, and then the quantities

$$\tau_0^- \leftarrow (\overline{\alpha}_0^-, \overline{\gamma}_0^-) \leftarrow \tau_1^- \leftarrow (\overline{\alpha}_1^-, \overline{\gamma}_1^-) \leftarrow \tau_2^- \leftarrow (\overline{\alpha}_2^-, \overline{\gamma}_2^-)$$

in the backward pass.

We also note that we have

$$\overline{\alpha}_\ell^+ + \overline{\alpha}_\ell^- = 1 \tag{C.47}$$

Forward Pass: Observe that $\mathbf{K}_0^+ = \text{Cov}(Z_0, Q_0^+)$. Now, from line 21, on simplification we get $Q_0^+ = -W_0^0$ whereby,

$$\mathbf{K}_0^+ = \text{var}(W^0) \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}. \tag{C.48}$$

Notice that from line 23, the pair (P_0^0, P_0^+) is jointly Gaussian with covariance matrix $\mathbf{K}_0^+$. But the above equation means that $P_0^+ = -P_0^0$, whereby $R_0^+ = 0$ from line 17.

Now, the linear denoiser $g_1^+(\cdot)$ is defined as in (C.7a). Note that since we are considering i.i.d Gaussian features for this problem, the random variable S_{trace} in this layer is a constant σ_{trace}. Therefore, similar to layer $\ell = 0$ by evaluating lines 17-23 of the algorithm we get $Q_1^+ = -Z_1^0$, whereby

$$\overline{\alpha}_1^+ = \frac{\sigma_{\text{trace}}^2 \overline{\gamma}_1^-}{\overline{\gamma}_0^+ + \sigma_{\text{trace}}^2 \overline{\gamma}_1^-}, \ \overline{\gamma}_1^+ = \frac{\overline{\gamma}_0^+}{\sigma_{\text{trace}}^2} = \frac{\lambda}{\sigma_{\text{trace}}^2 \beta}, \ \mathbf{K}_1^+ = \sigma_{\text{trace}}^2 \mathbf{K}_0^+. \tag{C.49}$$

Observe that this means

$$P_1^+ = -P_1^0. \tag{C.50}$$

Backward Pass: Since $Y = \phi_{\text{out}}(P_2^0, D) = P_2^0 + D$, line 36 of algorithm on simplification yields $P_2^- = D$, whereby we can get τ_2^-,

$$\tau_2^- = \mathbb{E}(P_2^-)^2 = \mathbb{E}[D^2] = \sigma_d^2. \tag{C.51}$$

Next, to calculate the terms $(\overline{\alpha}_1^-, \overline{\gamma}_1^-)$, we use the decoiser g_2^- defined in (C.7a) for line 33 of the algorithm to get $\widehat{P}_1$.

$$\widehat{P}_1 = \frac{\overline{\gamma}_1^+ R_1^+ + S_{\rm mp}^- \overline{\gamma}_2^- R_2^-}{\overline{\gamma}_1^+ + (S_{\rm mp}^-)^2 \overline{\gamma}_2^-} = \frac{S_{\rm mp}^-(S_{\rm mp}^+ P_1^0 + Q_2^-)}{\overline{\gamma}_1^+ + (S_{\rm mp}^-)^2}, \tag{C.52}$$

where we have used $\overline{\gamma}_2^- = 1$, $R_1^+ = P_1^0 + P_1^+ = 0$ due to (C.50), and $R_2^- = Z_2^0 + Q_2^- = S_{\rm mp}^+ P_1^0 + Q_2^-$ from lines 17, 32 and 4 respectively.

Then, we calculate $\overline{\alpha}_1^-$ and $\overline{\gamma}_1^-$ as $\overline{\alpha}_1^- = \mathbb{E}\frac{\partial g_2^-}{\partial P_1^+} = \mathbb{E}\frac{\overline{\gamma}_1^+}{\overline{\gamma}_1^+ + (S_{\rm mp}^-)^2}$. This gives,

$$\overline{\alpha}_1^- = \begin{cases} \frac{\lambda}{\sigma_{\rm trace}^2 \beta} G & \beta < 1 \\ (1 - \frac{1}{\beta}) + \frac{1}{\beta}\frac{\lambda}{\sigma_{\rm trace}^2 \beta} G & \beta \geq 1 \end{cases} \tag{C.53}$$

Here, in the overparameterized case ($\beta > 1$), the denoiser g_2^- outputs R_1^+ with probability $1 - \frac{1}{\beta}$ and $\frac{\lambda}{\sigma_{\rm trace}^2 \beta} G$ with probability $\frac{1}{\beta}$.

Next, from line 37 we get,

$$\overline{\gamma}_1^- = (\tfrac{1}{\overline{\alpha}_1^-} - 1)\overline{\gamma}_1^+ = \begin{cases} \frac{1}{G} - \frac{\lambda}{\sigma_{\rm trace}^2 \beta} & \beta < 1 \\ \frac{\frac{\lambda}{\sigma_{\rm trace}^2 \beta}(\frac{1}{G} - \frac{\lambda}{\sigma_{\rm trace}^2 \beta})}{\frac{\beta - 1}{G} + \frac{\lambda}{\sigma_{\rm trace}^2 \beta}} & \beta > 1 \end{cases} \tag{C.54}$$

Now from line 36 and equation (C.47) we get,

$$\begin{aligned} \overline{\alpha}_1^+ P_1^- &= \widehat{P}_1 - P_1^0 - \overline{\alpha}_1^- P_1^+ \overset{\text{(a)}}{=} \widehat{P}_1 - \overline{\alpha}_1^+ P_1^0 \\ &\overset{\text{(b)}}{=} \underbrace{\left(\frac{S_{\rm mp}^- S_{\rm mp}^+}{\frac{\lambda}{\sigma_{\rm trace}^2 \beta} + (S_{\rm mp}^-)^2} - \overline{\alpha}_1^+ \right)}_{A} P_1^0 + \underbrace{\frac{S_{\rm mp}^-}{\frac{\lambda}{\sigma_{\rm trace}^2 \beta} + (S_{\rm mp}^-)^2}}_{B} Q_2^- \end{aligned} \tag{C.55}$$

where (a) follows from (C.50) and (C.47), and (b) follows from (C.52). From this one can obtain $\tau_1^- = \mathbb{E}(P_1^-)^2$ which can be calculated using the knowledge that P_1^0, Q_2^- are independent Gaussian with covariances $\mathbb{E}(P_1^0)^2 = \sigma_{\rm trace}^2 \mathbb{V}(W^0)$, $\mathbb{E}(Q_2^-)^2 = \sigma_d^2$. Further, P_1^0, Q_2^- are independent of $(S_{\rm mp}^+, S_{\rm mp}^-)$.

Observe that by (C.55) we have

$$\tau_1^- = \frac{1}{(\overline{\alpha}_1^+)^2}\left(\mathbb{E}(A^2)\sigma_{\rm trace}^2 \mathbb{V}(W^0) + \mathbb{E}(B^2)\sigma_d^2 \right). \tag{C.56}$$

with some simplification we get

$$\mathbb{E}(A^2) = (\frac{\lambda}{\sigma_{\text{tr}}^2\beta})^2 G' - (\frac{\lambda}{\sigma_{\text{tr}}^2\beta}G)^2, \tag{C.57a}$$

$$\mathbb{E}(B^2) = G - \frac{\lambda}{\sigma_{\text{tr}}^2\beta}G', \tag{C.57b}$$

where $G = G_{\text{mp}}(-\frac{\lambda}{\sigma_{\text{tr}}^2\beta})$, with G_{mp} given in Appendix 2.9, and G' is the derivative of G_{mp} calculated at $-\frac{\lambda}{\sigma_{\text{tr}}^2\beta}$.

Now consider the **under-parametrized** case ($\beta < 1$):

Let $u = -\frac{\lambda}{\sigma_{\text{tr}}^2\beta}$ and $z = G_{\text{mp}}(u)$. In this case we have

$$\overline{\alpha}_1^+ = 1 - \frac{\lambda}{\sigma_{\text{tr}}^2\beta}G = 1 + uz. \tag{C.58}$$

Note that,

$$\begin{aligned} G_{\text{mp}}^{-1}(z) = u \quad &\overset{\text{(a)}}{\Rightarrow} R_{\text{mp}}(z) + \frac{1}{z} = u \\ &\overset{\text{(b)}}{\Rightarrow} \frac{1}{1-\beta z} + \frac{1}{z} = u, \end{aligned} \tag{C.59a}$$

where $R_{\text{mp}}(.)$ is the R-transform defined in [151] and (a) follows from the relationship between the R- and Stieltjes-transform and (b) follows from the fact that for Marchenko-Pastur distribution we have $R_{\text{mp}}(z) = \frac{1}{1-z\beta}$. Therefore,

$$\begin{aligned} &G_{\text{mp}}(\frac{1}{1-\beta z} + \frac{1}{z}) = z \\ &\Rightarrow G'_{\text{mp}}(\frac{1}{1-\beta z} + \frac{1}{z}) = G' = \frac{1}{\frac{\beta}{(1-\beta z)^2} - \frac{1}{z^2}}. \end{aligned} \tag{C.60}$$

For the **over-parametrized** case ($\beta > 1$) we have:

$$\overline{\alpha}_1^+ = \tfrac{1}{\beta}(1 + \tfrac{\lambda}{\sigma_{\text{trace}}^2\beta}G) = \frac{1-uz}{\beta}. \tag{C.61}$$

In this case, as mentioned in Appendix 2.9 and following the results from [151], the measure μ_β scales with β and thus $R_{\text{mp}}(z) = \frac{\beta}{1-z}$. Therefore, similar to (C.59a), z satisfies

$$\frac{\beta}{1-z} + \frac{1}{z} = u \quad \Rightarrow G' = \frac{1}{\frac{\beta}{(1-z)^2} - \frac{1}{z^2}}. \tag{C.62}$$

Now τ_1^- can be calculated as follows:

$$\tau_1^- = \eta^2\Big(u^2z^2\sigma_{\text{tr}}^2 var(W^0)(\kappa-1) + \sigma_d^2 z(uz\kappa+1)\Big) \tag{C.63}$$

where

$$\eta = \begin{cases} \frac{1}{(1+uz)} & \beta < 1 \\ \frac{\beta}{(1-uz)} & \beta \geq 1 \end{cases}, \quad \kappa = \begin{cases} \frac{(1-\beta z)^2}{\beta z^2-(1-\beta z)^2} & \beta < 1 \\ \frac{(1-z)^2}{\beta z^2-(1-z)^2} & \beta \geq 1 \end{cases} \tag{C.64}$$

and z is the solution to the fixed points

$$\begin{cases} \frac{1}{1-\beta z} + \frac{1}{z} = u & \beta < 1 \\ \frac{\beta}{1-z} + \frac{1}{z} = u & \beta \geq 1 \end{cases}. \tag{C.65}$$

□

C.6.3 Ridgeless Linear Regression

Here we consider the case of Ridge regression (C.43) when $\lambda \to 0^+$. Note that the solution to the problem (C.43) is $(\mathbf{X}^\mathsf{T}\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\mathsf{T}\mathbf{y}$ remains unique since $\lambda > 0$. The following result was stated in [55], and can be recovered using our methodology. Note however, that we calculate the generalization error whereas they have calculated the squared error, whereby we obtain an additional additive factor of σ_d^2. The result explains the double-descent phenomenon for Ridgeless linear regression.

Corollary 3. *For ridgeless linear regression, we have*

$$\lim_{\lambda\to 0^+} \mathcal{E}_{\mathrm{ts}}^{\mathsf{RR}} = \begin{cases} \frac{1}{1-\beta}\sigma_d^2 & \beta < 1 \\ \frac{\beta}{\beta-1}\sigma_d^2 + (1-\frac{1}{\beta})\sigma_{\mathrm{trace}}^2 \mathbb{V}\left(W^0\right) & \beta \geq 1 \end{cases}$$

Proof of Corollary 3. We calculate the parameters $\overline{\gamma}_0^+, \overline{\gamma}_1^-$, k_{22} and τ_1^- when $\lambda \to 0^+$. Before starting off, we note that

$$G_0 := \lim_{z\to 0^+} G_{\mathrm{mp}}(-z) = \begin{cases} \frac{\beta}{1-\beta} & \beta < 1 \\ \frac{\beta}{\beta-1} & \beta > 1 \end{cases}, \tag{C.66}$$

as described in Appendix 2.9. Following the derivations in Corollary 2, we have

$$\overline{\gamma}_0^+ = \lambda/\beta, \quad k_{22} = \mathbb{V}\left(W^0\right) \tag{C.67}$$

Now for $\lambda \to 0^+$, we have

$$1 - \overline{\alpha}_1^- = \begin{cases} 1 & \beta < 1 \\ \frac{1}{\beta} & \beta \geq 1 \end{cases}, \quad \overline{\gamma}_1^- = \begin{cases} \frac{1}{G_0} = \frac{1-\beta}{\beta} & \beta < 1 \\ \frac{\lambda}{(\beta-1)\sigma_{\text{trace}}^2 \beta} & \beta > 1 \end{cases}, \tag{C.68}$$

Using this in simplifying (C.55) for $\lambda \to 0^+$, we get

$$\tau_1^- = \mathbb{E}(P_1^-)^2 = \begin{cases} \sigma_d^2 G_0 & \beta < 1 \\ \beta \sigma_d^2 G_0 + \sigma_{\text{trace}}^2 \mathbb{V}(W^0)(\beta - 1) & \beta \geq 1 \end{cases}$$

where during the evaluation of $\mathbb{E}\left(\frac{S_{\text{mp}}^-}{\overline{\gamma}_1^+ + (S_{\text{mp}}^-)^2}\right)^2$, for the case of $\beta > 1$, we need to account for the point mass at 0 for S_{mp}^- with weight $1 - \frac{1}{\beta}$.

Next, notice that

$$a := \frac{\overline{\gamma}_0^+ \sigma_{\text{trace}}}{\overline{\gamma}_0^+ + \overline{\gamma}_1^- \sigma_{\text{trace}}^2} = \begin{cases} 0 & \beta < 1 \\ (1 - \frac{1}{\beta})\sigma_{\text{trace}} & \beta \geq 1 \end{cases},$$

and,

$$b := \frac{\overline{\gamma}_1^- \sigma_{\text{trace}}^2}{\overline{\gamma}_0^+ + \overline{\gamma}_1^- \sigma_{\text{trace}}^2} = \begin{cases} 1 & \beta < 1 \\ \frac{1}{\beta} & \beta \geq 1 \end{cases},$$

Thus applying Corollary 1, we get

$$\begin{aligned} \mathcal{E}_{\text{ts}}^{\mathsf{RR}} &= a^2 k_{22} + b^2 \tau_1^- + \sigma_d^2 \\ &= \begin{cases} \frac{1}{1-\beta}\sigma_d^2 & \beta < 1 \\ \frac{\beta}{\beta-1}\sigma_d^2 + (1 - \frac{1}{\beta})\sigma_{\text{trace}}^2 \mathbb{V}(W^0) & \beta \geq 1 \end{cases} \end{aligned}$$

This proves the claim. □

C.6.4 Train-Test Mismatch

Observe that our formulation allows for analyzing the effect of mismatch in the training and test distribution. One can consider arbitrary joint distributions over $(S_{\text{trace}}, S_{\text{ts}})$ that model the mismatch between training and test features. Here we give a simple example which highlights the effect of this mismatch.

Definition 9 (Bernoulli ε-mismatch)**.** $(S_{\text{ts}}, S_{\text{trace}})$ has a bivariate Bernoulli distribution with

- $\Pr\{S_{\text{trace}}=S_{\text{ts}}=0\} = \mathbb{P}\{S_{\text{trace}}=S_{\text{ts}}=1\} = (1-\varepsilon)/2$
- $\Pr\{S_{\text{trace}}=0, S_{\text{ts}}=1\} = \mathbb{P}\{S_{\text{trace}}=1, S_{\text{ts}}=0\} = \varepsilon/2$

Notice that the marginal distribution of the S_{trace} in the Bernoulli $\varepsilon-$mismatch model is such that $\mathbb{P}(S_{\text{trace}} \neq 0) = \frac{1}{2}$. Hence half of the features extracted by the matrix V_0 are relevant during training. Similarly, $\mathbb{P}(S_{\text{ts}} \neq 0) = \frac{1}{2}$. However the features spanned by the test data do not exactly overlap with the features captured in the training data, and the fraction of features common to both the training and test data is $1-\varepsilon$. Hence for $\varepsilon = 0$, there is no training-test mismatch, whereas for $\varepsilon = 1$ there is a complete mismatch.

The following result shows that the generalization error increases linearly with the mismatch parameter ε.

Corollary 4 (Mismatch)**.** *Consider the problem of Linear Regression* (C.43) *under the conditions of Corollary 1. Additionally suppose we have Bernoulli ε-mismatch between the training and test distributions. Then*

$$\mathcal{E}_{\text{ts}} = \tfrac{k_{22}}{2}((1-\varepsilon)\gamma^{*2} + \varepsilon) + \tfrac{\tau_1^-}{2}(1-\gamma^*)(1-\varepsilon) + \sigma_d^2,$$

where $\gamma^* := \frac{\overline{\gamma}_0^+}{\overline{\gamma}_0^+ + \overline{\gamma}_1^-}$. *The terms $k_{22}, \tau_1^-, \gamma^*$ are independent of ε.*

Proof. This follows directly by calculating the expectations of the terms in Corollary 1, with the joint distribution of $(S_{\text{trace}}, S_{\text{ts}})$ given in Definition 9. $\square$

The quantities k_{22} and τ_1^- in the result above can be calculated similar to the derivation in the proof of Corollary 2 and can in general depend on the regularization parameter λ and overparameterization parameter β.

C.6.5 Logistic Regression

The precise analysis for the special case of regularized logistic regression estimator with i.i.d Gaussian features is provided in [136]. Consider the logistic regression model,

$$\mathbb{P}(y_i = 1|\mathbf{x}_i) := \rho(\mathbf{x}_i^\mathsf{T}\mathbf{w}) \quad \text{for } i = 1, \cdots, N$$

where $\rho(x) = \frac{1}{1+e^{-x}}$ is the standard logistic function.

In this problem we consider estimates of $\mathbf{w}^0$ such that

$$\widehat{\mathbf{w}} := \underset{\mathbf{w}}{\text{argmin}} \mathbf{1}^\mathsf{T} \log(1 + e^{\mathbf{Xw}}) - \mathbf{y}^\mathsf{T}\mathbf{Xw} + F_{\text{in}}(\mathbf{w}).$$

where F_{in} is the regularization function. This is a special case of optimization problem (5.2) where

$$F_{\text{out}}(\mathbf{y}, \mathbf{Xw}) = \mathbf{1}^\mathsf{T} \log(1 + e^{\mathbf{Xw}}) - \mathbf{y}^\mathsf{T}\mathbf{Xw}. \tag{C.69}$$

Similar to the linear regression model, using the ML-VAMP GLM learning algorithm, we can characterize the generalization error for this model with quantities $\mathbf{K}_0^+, \tau_1^-, \overline{\gamma}_0^+, \overline{\gamma}_1^-$ given by algorithm 12. We note that in this case, the output non-linearity is

$$\phi_{\text{out}}(p_2, d) = \mathbb{1}_{\{\rho(p_2)>d\}} \tag{C.70}$$

where $d \sim \text{Unif}(0, 1)$. Also, the denoisers g_0^+, and g_3^- can be derived as the proximal operators of F_{in}, and F_{out} defined in (5.25).

C.6.6 Support Vector Machines

The asymptotic generalization error for support vector machine (SVM) is provided in [29]. Our model can also handle SVMs. Similar to logistic regression, SVM finds a linear classifier using the hinge loss instead of logistic loss. Assuming the class labels are $y = \pm 1$ the hinge loss is

$$\ell_{\text{hinge}}(y, \hat{y}) = \max(0, 1 - y\hat{y}). \tag{C.71}$$

Therefore, if we take

$$F_{\text{out}}(\mathbf{y}, \mathbf{X}\mathbf{w}) = \sum_i \max(0, 1 - y_i \mathbf{X}_i \mathbf{w}), \tag{C.72}$$

where $\mathbf{X}_i$ is the i^{th} row of the data matrix, the ML-VAMP algorithm for GLMs finds the SVM classifier. The algorithm would have proximal map of hinge loss and our theory provides exact predictions for the estimation and prediction error of SVM.

As with all other models considered in this work, the true underlying data generating model could be anything that can be represented by the graphical model in Figure 5.1, e.g. logistic or probit model, and our theory is able to exactly predict the error when SVM is applied to learn such linear classifiers in the large system limit.

Bibliography

[1] Madhu S Advani and Andrew M Saxe. High-dimensional dynamics of generalization error in neural networks. *arXiv preprint arXiv:1710.03667*, 2017.

[2] Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. In *Advances in Neural Information Processing Systems*, pages 6155–6166, 2019.

[3] Luca Ambrogioni, Umut Gu clu, Ya gmur Gu cluturk, Max Hinne, Eric Maris, and Marcel A. J. van Gerven. Wasserstein Variational Inference. *arXiv:1805.11284 [cs, stat]*, May 2018.

[4] Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. *arXiv preprint arXiv:1901.08584*, 2019.

[5] Benjamin Aubin, Antoine Maillard, Florent Krzakala, Nicolas Macris, Lenka Zdeborová, et al. The committee machine: Computational to statistical gaps in learning a two-layers neural network. In *Advances in Neural Information Processing Systems*, pages 3223–3234, 2018.

[6] Jean Barbier, Mohamad Dia, Nicolas Macris, and Florent Krzakala. The mutual information in random linear estimation. In *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 625–632. IEEE, 2016.

[7] Jean Barbier, Florent Krzakala, Nicolas Macris, Léo Miolane, and Lenka Zdeborová. Optimal errors and phase transitions in high-dimensional generalized linear models. *Proc. Nat. Acad. Sci.*, 116(12):5451–5460, 2019.

[8] Jean Barbier, Nicolas Macris, Mohamad Dia, and Florent Krzakala. Mutual information and optimality of approximate message-passing in random linear estimation. *arXiv preprint arXiv:1701.05823*, 2017.

[9] Peter L Bartlett, Philip M Long, Gábor Lugosi, and Alexander Tsigler. Benign overfitting in linear regression. *arXiv preprint arXiv:1906.11300*, 2019.

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

[11] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proc. National Academy of Sciences*, 116(32):15849–15854, 2019.

[12] Mikhail Belkin, Daniel Hsu, and Ji Xu. Two models of double descent for weak features. *arXiv preprint arXiv:1903.07571*, 2019.

[13] Mikhail Belkin, Siyuan Ma, and Soumik Mandal. To understand deep learning we need to understand kernel learning. *arXiv preprint arXiv:1802.01396*, 2018.

[14] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester. Image inpainting. In *Proc. ACM Conf. Computer Graphics and Interactive Techniques*, pages 417–424, 2000.

[15] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational Inference: A Review for Statisticians. *Journal of the American Statistical Association*, 112(518):859–877, April 2017.

[16] Ashish Bora, Ajil Jalal, Eric Price, and Alexandros G Dimakis. Compressed sensing using generative models. *Proc. ICML*, 2017.

[17] Mark Borgerding, Philip Schniter, and Sundeep Rangan. AMP-inspired deep networks for sparse linear inverse problems. *IEEE Trans. Signal Processing*, 65(16):4293–4308, 2017.

[18] Evan Byrne, Antoine Chatalic, Rémi Gribonval, and Philip Schniter. Sketched clustering via hybrid approximate message passing. *IEEE Transactions on Signal Processing*, 67(17):4556–4569, 2019.

[19] Burak Cakmak, Ole Winther, and Bernard H Fleury. S-AMP: Approximate message passing for general matrix ensembles. In *Proc. IEEE ITW*, 2014.

[20] Francesco Caltagirone, Florent Krzakala, and Lenka Zdeborová. On Convergence of Approximate Message Passing. *2014 IEEE International Symposium on Information Theory*, pages 1812–1816, June 2014.

[21] A. Chambolle, R. A. De Vore, Nam-Yong Lee, and B. J. Lucier. Nonlinear wavelet image processing: Variational problems, compression, and noise removal through wavelet shrinkage. *IEEE Transactions on Image Processing*, 7(3):319–335, March 1998.

[22] J. H. Rick Chang, Chun-Liang Li, Barnabas Poczos, B. V. K. Vijaya Kumar, and Aswin C. Sankaranarayanan. One network to solve them all—solving linear inverse problems using deep projection models. In *IEEE Int. Conf. Computer Vision*, pages 5889–5898. IEEE, 2017.

[23] Xiang Cheng, Niladri S Chatterji, Yasin Abbasi-Yadkori, Peter L Bartlett, and Michael I Jordan. Sharp convergence rates for langevin dynamics in the nonconvex setting. *arXiv preprint arXiv:1805.01648*, 2018.

[24] Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surfaces of multilayer networks. In *Artificial Intelligence and Statistics*, pages 192–204, 2015.

[25] Shane F Cotter, Bhaskar D Rao, Kjersti Engan, and Kenneth Kreutz-Delgado. Sparse solutions to linear inverse problems with multiple measurement vectors. *IEEE Transactions on Signal Processing*, 53(7):2477–2488, 2005.

[26] Amit Daniely. Sgd learns the conjugate kernel class of the network. In *Advances in Neural Information Processing Systems*, pages 2422–2430, 2017.

[27] Amit Daniely, Roy Frostig, and Yoram Singer. Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In *Advances In Neural Information Processing Systems*, pages 2253–2261, 2016.

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

[29] Zeyu Deng, Abla Kammoun, and Christos Thrampoulidis. A model of double descent for high-dimensional binary linear classification. *arXiv preprint arXiv:1911.05822*, 2019.

[30] Dongning Guo and S. Verdu. Randomly spread CDMA: Asymptotics via statistical physics. *IEEE Transactions on Information Theory*, 51(6):1983–2010, June 2005.

[31] D. L. Donoho, A. Maleki, and A. Montanari. Message-passing algorithms for compressed sensing. *Proc. Nat. Acad. Sci.*, 106(45):18914–18919, Nov. 2009.

[32] D. L. Donoho, A. Maleki, and A. Montanari. Message passing algorithms for compressed sensing. In *Proc. Inform. Theory Workshop*, pages 1–5, 2010.

[33] David L. Donoho, Arian Maleki, and Andrea Montanari. Message Passing Algorithms for Compressed Sensing. *Proceedings of the National Academy of Sciences*, 106(45):18914–18919, November 2009.

[34] David L Donoho, Arian Maleki, and Andrea Montanari. Message-passing algorithms for compressed sensing. *Proc. National Academy of Sciences*, 106(45):18914–18919, 2009.

[35] David L. Donoho, Arian Maleki, and Andrea Montanari. Message Passing Algorithms for Compressed Sensing: I. Motivation and Construction. *arXiv:0911.4219 [cs, math]*, November 2009.

[36] Simon S Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. *arXiv preprint arXiv:1810.02054*, 2018.

[37] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Olivier Mastropietro, Alex Lamb, Martin Arjovsky, and Aaron Courville. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016.

[38] Yonina C. Eldar and Gitta Kutyniok. *Compressed Sensing: Theory and Applications*. Cambridge Univ. Press, June 2012.

[39] Melikasadat Emami, Mojtaba Sahraee-Ardakan, Parthe Pandit, Sundeep Rangan, and Alyson Fletcher. Generalization error of generalized linear models in high dimensions. In *International Conference on Machine Learning*, pages 2892–2901. PMLR, 2020.

[40] Alyson K Fletcher, Parthe Pandit, Sundeep Rangan, Subrata Sarkar, and Philip Schniter. Plug-in estimation in high-dimensional linear inverse problems: A rigorous analysis. In *Advances in Neural Information Processing Systems*, pages 7440–7449, 2018.

[41] Alyson K Fletcher, Sundeep Rangan, and P. Schniter. Inference in deep networks in high dimensions. *arXiv:1706.06549*, 2017.

[42] Alyson K Fletcher, Sundeep Rangan, and P. Schniter. Inference in deep networks in high dimensions. *Proc. IEEE Int. Symp. Information Theory*, 2018.

[43] Alyson K. Fletcher, Sundeep Rangan, and Philip Schniter. Inference in Deep Networks in High Dimensions. In *2018 IEEE International Symposium on Information Theory (ISIT)*, pages 1884–1888, Vail, CO, June 2018. IEEE.

[44] Alyson K. Fletcher, Mojtaba Sahraee-Ardakan, Sundeep Rangan, and Philip Schniter. Expectation Consistent Approximate Inference: Generalizations and Convergence. *arXiv:1602.07795 [cs, math, stat]*, February 2016.

[45] Alyson K. Fletcher, Mojtaba Sahraee-Ardakan, Sundeep Rangan, and Philip Schniter. Expectation consistent approximate inference: Generalizations and convergence. In *Proc. IEEE Int. Symp. Information Theory*, pages 190–194, 2016.

[46] Alyson K Fletcher and Philip Schniter. Learning and free energies for vector approximate message passing. In *IEEE Int. Conf. Acoustics, Speech and Signal Processing*, pages 4247–4251, 2017.

[47] Marylou Gabrié, Andre Manoel, Clément Luneau, Jean Barbier, Nicolas Macris, Florent Krzakala, and Lenka Zdeborová. Entropy and mutual information in models of deep neural networks. In *Proc. NIPS*, 2018.

[48] R. Gallager. Low-density parity-check codes. *IRE Transactions on Information Theory*, 8(1):21–28, January 1962.

[49] Rong Ge, Chi Jin, and Yi Zheng. No spurious local minima in nonconvex low rank problems: A unified geometric analysis. In *International Conference on Machine Learning*, pages 1233–1242. PMLR, 2017.

[50] Cédric Gerbelot, Alia Abbara, and Florent Krzakala. Asymptotic errors for convex penalized linear regression beyond gaussian matrices. *arXiv preprint arXiv:2002.04372*, 2020.

[51] Raja Giryes, Guillermo Sapiro, and Alex M Bronstein. Deep neural networks with random Gaussian weights: A universal classification strategy? *IEEE Trans. Signal Processing*, 64(13):3444–3457, 2016.

[52] Clark R Givens, Rae Michael Shortt, et al. A class of wasserstein metrics for probability distributions. *The Michigan Mathematical Journal*, 31(2):231–240, 1984.

[53] Paul Hand and Vladislav Voroninski. Global guarantees for enforcing deep generative priors by empirical risk. *arXiv:1705.07576*, 2017.

[54] Boris Hanin and David Rolnick. How to start training: The effect of initialization and architecture. In *Advances in Neural Information Processing Systems*, pages 571–581, 2018.

[55] Trevor Hastie, Andrea Montanari, Saharon Rosset, and Ryan J Tibshirani. Surprises in high-dimensional ridgeless least squares interpolation. *arXiv preprint arXiv:1903.08560*, 2019.

[56] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. 2017.

[57] W Keith Hastings. Monte carlo sampling methods using markov chains and their applications. 1970.

[58] Bingsheng He, Han Liu, Juwei Lu, and Xiaoming Yuan. Application of the strictly contractive peaceman-rachford splitting method to multi-block separable convex programming. In *Splitting Methods in Communication, Imaging, Science, and Eng.* Springer, 2016.

[59] Bingsheng He, Han Liu, Zhaoran Wang, and Xiaoming Yuan. A strictly contractive peaceman–rachford splitting method for convex programming. *SIAM Journal on Optimization*, 24(3):1011–1040, 2014.

[60] Hengtao He, Chao-Kai Wen, and Shi Jin. Generalized Expectation Consistent Signal Recovery for Nonlinear Measurements. *arXiv:1701.04301 [cs, math]*, January 2017.

[61] Hengtao He, Chao-Kai Wen, and Shi Jin. Generalized expectation consistent signal recovery for nonlinear measurements. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pages 2333–2337. IEEE, 2017.

[62] Tom Heskes. Stable Fixed Points of Loopy Belief Propagation Are Local Minima of the Bethe Free Energy. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 359–366. MIT Press, 2003.

[63] Tom Heskes, Manfred Opper, Wim Wiegerinck, Ole Winther, and Onno Zoeter. Approximate inference techniques with expectation constraints. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(11):P11015–P11015, November 2005.

[64] Fumio Hiai and Denes Petz. *The Semicircle Law, Free Random Variables and Entropy (Mathematical Surveys & Monographs)*. American Mathematical Society, Boston, MA, USA, 2006.

[65] Wen Huang, Paul Hand, Reinhard Heckel, and Vladislav Voroninski. A provably convergent scheme for compressive sensing under random generative priors. *arXiv preprint arXiv:1812.04176*, 2018.

[66] Peter J Huber. A robust version of the probability ratio test. *The Annals of Mathematical Statistics*, pages 1753–1758, 1965.

[67] Peter J Huber. *Robust statistical procedures.* SIAM, 1996.

[68] Peter J Huber. *Robust Statistics.* Springer, 2011.

[69] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, pages 8571–8580, 2018.

[70] Adel Javanmard and Andrea Montanari. State evolution for general approximate message passing algorithms, with applications to spatial coupling. *Information and Inference*, 2(2):115–144, 2013.

[71] Yoshiyuki Kabashima. A cdma multiuser detection algorithm on the basis of belief propagation. *Journal of Physics A: Mathematical and General*, 36(43):11111, 2003.

[72] Yoshiyuki Kabashima and Mikko Vehkapera. Signal recovery using expectation consistent approximation for linear observations. *arXiv:1401.5151 [cond-mat]*, January 2014.

[73] Maya Kabkab, Pouya Samangouei, and Rama Chellappa. Task-aware compressed sensing with generative adversarial networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[74] Nicolas Keriven, Anthony Bourrier, Rémi Gribonval, and Patrick Pérez. Sketching for large-scale learning of mixture models. *Information and Inference: A Journal of the IMA*, 7(3):447–508, 2017.

[75] Nicolas Keriven, Nicolas Tremblay, Yann Traonmilin, and Rémi Gribonval. Compressive k-means. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6369–6373. IEEE, 2017.

[76] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[77] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv:1312.6114*, 2013.

[78] Florent Krzakala, Andre Manoel, Eric W Tramel, and Lenka Zdeborová. Variational free energies for compressed sensing. In *Proc. IEEE Int. Symp. Information Theory*, pages 1499–1503, 2014.

[79] Florent Krzakala, Marc Mézard, François Sausset, YF Sun, and Lenka Zdeborová. Statistical-physics-based reconstruction in compressed sensing. *Physical Review X*, 2(2):021005, 2012.

[80] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[81] Qi Lei, Ajil Jalal, Inderjit S Dhillon, and Alexandros G Dimakis. Inverting deep generative models, one layer at a time. *arXiv:1906.07437*, 2019.

[82] Ping Li and Phan-Minh Nguyen. On random deep weight-tied autoencoders: Exact asymptotic analysis, phase transitions, and implications to training. In *Proc. Int. Conf. Learning Research*, 2019.

[83] Dong Liang, Leslie Ying, and Feng Liang. Parallel MRI Acceleration Using M-FOCUSS. In *Proc. International Conference on Bioinformatics and Biomedical Engineering*, pages 1–4. IEEE, 2009.

[84] Ting Liu, Chao-Kai Wen, Shi Jin, and Xiaohu You. Generalized Turbo Signal Recovery for Nonlinear Measurements and Orthogonal Sensing Matrices. *2016 IEEE International Symposium on Information Theory (ISIT)*, pages 2883–2887, July 2016.

[85] David Luenberger. *Optimization by Vector Space Methods*. Wiley Professional Paperback Series. 1969.

[86] Junjie Ma and Li Ping. Orthogonal AMP. *IEEE Access*, 5:2020–2033, 2017.

[87] Junjie Ma, Xiaojun Yuan, and Li Ping. Turbo Compressed Sensing with Partial DFT Sensing Matrix. *IEEE Signal Processing Letters*, 22(2):158–161, February 2015.

[88] Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 5188–5196, 2015.

[89] Andre Manoel, Florent Krzakala, Marc Mézard, and Lenka Zdeborová. Multi-layer generalized linear estimation. In *Proc. IEEE Int. Symp. Information Theory*, pages 2098–2102, 2017.

[90] Andre Manoel, Florent Krzakala, Gaël Varoquaux, Bertrand Thirion, and Lenka Zdeborová. Approximate message-passing for convex optimization with non-separable penalties. *arXiv:1809.06304 [cs, math, stat]*, September 2018.

[91] Andre Manoel, Florent Krzakala, Gaël Varoquaux, Bertrand Thirion, and Lenka Zdeborová. Approximate message-passing for convex optimization with non-separable penalties. *arXiv preprint arXiv:1809.06304*, 2018.

[92] P. McCullagh and J. A. Nelder. *Generalized Linear Models.* Chapman & Hall, 2nd edition, 1989.

[93] Song Mei and Andrea Montanari. The generalization error of random features regression: Precise asymptotics and double descent curve. *arXiv preprint arXiv:1908.05355*, 2019.

[94] Song Mei, Andrea Montanari, and Phan-Minh Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33):E7665–E7671, 2018.

[95] Chris Metzler, Ali Mousavi, and Richard Baraniuk. Learned D-amp: Principled neural network based compressive image recovery. In *Proc. NIPS*, pages 1772–1783, 2017.

[96] Thomas P Minka. Expectation propagation for approximate bayesian inference. In *Proc. UAI*, pages 362–369, 2001.

[97] Dustin G Mixon and Soledad Villar. Sunlayer: Stable denoising with generative networks. *arXiv preprint arXiv:1803.09319*, 2018.

[98] Andrea Montanari. Graphical Models Concepts in Compressed Sensing. *arXiv:1011.4328 [cs, math]*, November 2010.

[99] Andrea Montanari, Feng Ruan, Youngtak Sohn, and Jun Yan. The generalization error of max-margin linear classifiers: High-dimensional asymptotics in the overparametrized regime. *arXiv preprint arXiv:1911.01544*, 2019.

[100] Ali Mousavi, Ankit B Patel, and Richard G Baraniuk. A deep learning approach to structured signal recovery. In *Proc. Allerton Conf. Comm. Control & Comput.*, pages 1336–1343, 2015.

[101] Kevin Murphy, Yair Weiss, and Michael I. Jordan. Loopy Belief Propagation for Approximate Inference: An Empirical Study. *arXiv:1301.6725 [cs]*, January 2013.

[102] Vidya Muthukumar, Kailas Vodrahalli, and Anant Sahai. Harmless interpolation of noisy data in regression. In *2019 IEEE International Symposium on Information Theory (ISIT)*, pages 2299–2303. IEEE, 2019.

[103] Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

[104] Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011.

[105] Sahand N Negahban, Pradeep Ravikumar, Martin J Wainwright, and Bin Yu. A unified framework for high-dimensional analysis of m-estimators with decomposable regularizers. *Statistical Science*, pages 538–557, 2012.

[106] Y. E. Nesterov. A method for solving the convex programming problem with convergence rate O(1/k$\hat{2}$). *Dokl. Akad. Nauk SSSR*, 269:543–547, 1983.

[107] Behnam Neyshabur, Zhiyuan Li, Srinadh Bhojanapalli, Yann LeCun, and Nathan Srebro. Towards understanding the role of over-parametrization in generalization of neural networks. *arXiv preprint arXiv:1805.12076*, 2018.

[108] Roman Novak, Lechao Xiao, Yasaman Bahri, Jaehoon Lee, Greg Yang, Jiri Hron, Daniel A Abolafia, Jeffrey Pennington, and Jascha Sohl-Dickstein. Bayesian deep convolutional networks with many channels are Gaussian processes. *arXiv preprint arXiv:1810.05148*, 2018.

[109] Guillaume Obozinski, Ben Taskar, and Michael Jordan. Multi-task feature selection. *Statistics Department, UC Berkeley, Tech. Rep*, 2(2.2), 2006.

[110] Manfred Opper and Ole Winther. Expectation Consistent Approximate Inference. page 28, 2005.

[111] Manfred Opper and Ole Winther. Expectation consistent approximate inference. *J. Machine Learning Res.*, 6:2177–2204, December 2005.

[112] P. Pandit, M. Sahraee, S. Rangan, and A. K. Fletcher. Asymptotics of MAP inference in deep networks. In *Proc. IEEE Int. Symp. Information Theory*, pages 842–846, 2019.

[113] Parthe Pandit, Mojtaba Sahraee, Sundeep Rangan, and Alyson K. Fletcher. Asymptotics of MAP Inference in Deep Networks. *arXiv:1903.01293 [cs, math, stat]*, March 2019.

[114] Parthe Pandit, Mojtaba Sahraee-Ardakan, Sundeep Rangan, Philip Schniter, and Alyson K Fletcher. Inference with deep generative priors in high dimensions. *arXiv preprint arXiv:1911.03409*, 2019.

[115] Parthe Pandit, Mojtaba Sahraee-Ardakan, Sundeep Rangan, Philip Schniter, and Alyson K Fletcher. Inference with deep generative priors in high dimensions. *IEEE Journal on Selected Areas in Information Theory*, 1(1):336–347, 2020.

[116] Parthe Pandit, Mojtaba Sahraee Ardakan, Sundeep Rangan, Philip Schniter, and Alyson K Fletcher. Matrix inference and estimation in multi-layer models. *Advances in Neural Information Processing Systems*, 33, 2020.

[117] Parthe Pandit, Mojtaba Sahraee Ardakan, Sundeep Rangan, Philip Schniter, and Alyson K Fletcher. Matrix inference and estimation in multi-layer models. *Journal of Statistical Mechanics: Theory and Experiment*, 2022, 2022.

[118] Marcelo Pereyra, Philip Schniter, Emilie Chouzenoux, Jean-Christophe Pesquet, Jean-Yves Tourneret, Alfred Hero, and Steve McLaughlin. A Survey of Stochastic Simulation and Optimization Methods in Signal Processing. *IEEE Journal of Selected Topics in Signal Processing*, 10(2):224–241, March 2016.

[119] Marcelo Pereyra, Philip Schniter, Emilie Chouzenoux, Jean-Christophe Pesquet, Jean-Yves Tourneret, Alfred O Hero, and Steve McLaughlin. A survey of stochastic simulation and optimization methods in signal processing. *IEEE J. Sel. Top. Signal Process.*, 10(2):224–241, 2015.

[120] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989.

[121] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[122] Sundeep Rangan. Generalized Approximate Message Passing for Estimation with Random Linear Mixing. *arXiv:1010.5141 [cs, math]*, October 2010.

[123] Sundeep Rangan. Generalized approximate message passing for estimation with random linear mixing. In *Proc. IEEE Int. Symp. Information Theory*, pages 2168–2172, 2011.

[124] Sundeep Rangan, Alyson K. Fletcher, and Vivek K. Goyal. Asymptotic Analysis of MAP Estimation via the Replica Method and Applications to Compressed Sensing. *IEEE Transactions on Information Theory*, 58(3):1902–1923, March 2012.

[125] Sundeep Rangan, Philip Schniter, and Alyson K. Fletcher. On the convergence of approximate message passing with arbitrary matrices. In *Proc. IEEE Int. Symp. Information Theory*, pages 236–240, July 2014.

[126] Sundeep Rangan, Philip Schniter, and Alyson K. Fletcher. Vector Approximate Message Passing. *arXiv:1610.03082 [cs, math]*, October 2016.

[127] Sundeep Rangan, Philip Schniter, and Alyson K Fletcher. Vector approximate message passing. *IEEE Trans. Information Theory*, 65(10):6664–6684, 2019.

[128] Sundeep Rangan, Philip Schniter, Alyson K. Fletcher, and Subrata Sarkar. On the Convergence of Approximate Message Passing with Arbitrary Matrices. *arXiv:1402.3210 [cs, math]*, February 2014.

[129] Sundeep Rangan, Philip Schniter, Erwin Riegler, Alyson K Fletcher, and Volkan Cevher. Fixed points of generalized approximate message passing with arbitrary matrices. *IEEE Trans. Information Theory*, 62(12):7464–7474, 2016.

[130] Rajesh Ranganath, Jaan Altosaar, Dustin Tran, and David M Blei. Operator Variational Inference. page 11, 2016.

[131] G. Reeves. Additivity of information in multilayer networks via additive Gaussian noise transforms. In *Proc. Allerton Conf. Comm. Control & Comput.*, pages 1064–1070, 2017.

[132] Galen Reeves and Henry D Pfister. The replica-symmetric prediction for compressed sensing with Gaussian matrices is exact. In *Proc. IEEE Int. Symp. Information Theory*, pages 665–669, 2016.

[133] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proc. ICML*, pages 1278–1286, 2014.

[134] Cynthia Rush and Ramji Venkataramanan. Finite-sample analysis of approximate message passing algorithms. *IEEE Trans. Inform. Theory*, 64(11):7264–7286, 2018.

[135] Ruslan Salakhutdinov. Learning deep generative models. *Annual Review of Statistics and Its Application*, 2, 2015.

[136] Fariborz Salehi, Ehsan Abbasi, and Babak Hassibi. The impact of regularization on high-dimensional logistic regression. *arXiv preprint arXiv:1906.03761*, 2019.

[137] Subrata Sarkar, Alyson K Fletcher, Sundeep Rangan, and Philip Schniter. Bilinear recovery using adaptive vector-amp. *IEEE Tran. Signal Processing*, 67(13):3383–3396, 2019.

[138] P. Schniter. Turbo reconstruction of structured sparse signals. In *2010 44th Annual Conference on Information Sciences and Systems (CISS)*, pages 1–6, March 2010.

[139] Philip Schniter, Sundeep Rangan, and Alyson K. Fletcher. Vector approximate message passing for the generalized linear model. In *2016 50th Asilomar Conference on Signals, Systems and Computers*, pages 1525–1529, Pacific Grove, CA, USA, November 2016. IEEE.

[140] Philip Schniter, Sundeep Rangan, and Alyson K Fletcher. Vector approximate message passing for the generalized linear model. In *Proc. Asilomar Conf. Signals, Syst. & Computers*, pages 1525–1529, 2016.

[141] Samuel S Schoenholz, Justin Gilmer, Surya Ganguli, and Jascha Sohl-Dickstein. Deep information propagation. *arXiv preprint arXiv:1611.01232*, 2016.

[142] Viraj Shah and Chinmay Hegde. Solving linear inverse problems using GAN priors: An algorithm with provable guarantees. In *IEEE Int. Conf. Acoustics, Speech and Signal Processing*, pages 4609–4613, 2018.

[143] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.

[144] Keigo Takeuchi. Rigorous Dynamics of Expectation-Propagation-Based Signal Recovery from Unitarily Invariant Measurements. *arXiv:1701.05284 [cs, math]*, January 2017.

[145] Keigo Takeuchi. Rigorous dynamics of expectation-propagation-based signal recovery from unitarily invariant measurements. In *Proc. IEEE Int. Symp. Information Theory*, pages 501–505, 2017.

[146] T. Tanaka. A statistical-mechanics approach to large-system analysis of CDMA multiuser detectors. *IEEE Transactions on Information Theory*, 48(11):2888–2910, November 2002.

[147] Andreas Themelis and Panagiotis Patrinos. Douglas–rachford splitting and admm for nonconvex optimization: Tight convergence results. *SIAM Journal on Optimization*, 30(1):149–181, 2020.

[148] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[149] Volker Tresp. A bayesian committee machine. *Neural computation*, 12(11):2719–2741, 2000.

[150] Subarna Tripathi, Zachary C Lipton, and Truong Q Nguyen. Correction by projection: Denoising images with generative adversarial networks. *arXiv preprint arXiv:1803.04477*, 2018.

[151] Antonia M Tulino, Sergio Verdú, et al. Random matrix theory and wireless communications. *Foundations and Trends® in Communications and Information Theory*, 1(1):1–182, 2004.

[152] George Tzagkarakis, Dimitris Milioris, and Panagiotis Tsakalides. Multiple-measurement Bayesian compressed sensing using GSM priors for DOA estimation. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2610–2613. IEEE, 2010.

[153] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pages 9446–9454, 2018.

[154] Dave Van Veen, Ajil Jalal, Mahdi Soltanolkotabi, Eric Price, Sriram Vishwanath, and Alexandros G Dimakis. Compressed sensing with deep image prior and learned regularization. *arXiv preprint arXiv:1806.06438*, 2018.

[155] Sergio Verdu et al. *Multiuser detection*. Cambridge university press, 1998.

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

[157] Cédric Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

[158] Martin J. Wainwright and Michael I. Jordan. *Graphical Models, Exponential Families, and Variational Inference*, volume 1 of *Foundations and Trends R in Machine Learning*. 2008.

[159] Martin J Wainwright and Michael I Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1–2):1–305, 2008.

[160] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient Langevin dynamics. In *Proc. 28th Int. Conf. Machine Learning*, pages 681–688, 2011.

[161] Jonathan S Yedidia, William T. Freeman, and Yair Weiss. Generalized Belief Propagation. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 689–695. MIT Press, 2001.

[162] Jonathan S Yedidia, William T Freeman, and Yair Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Trans. Information Theory*, 51(7):2282–2312, 2005.

[163] Raymond Yeh, Chen Chen, Teck Yian Lim, Mark Hasegawa-Johnson, and Minh N Do. Semantic image inpainting with perceptual and contextual losses. *arXiv:1607.07539*, 2016.

[164] Xinyang Yi, Constantine Caramanis, and Sujay Sanghavi. Alternating minimization for mixed linear regression. In *International Conference on Machine Learning*, pages 613–621, 2014.

[165] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*, 2015.

[166] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

[167] Justin Ziniel and Philip Schniter. Efficient high-dimensional inference in the multiple measurement vector problem. *IEEE Transactions on Signal Processing*, 61(2):340–354, 2012.

ProQuest Number: 28867277

INFORMATION TO ALL USERS
The quality and completeness of this reproduction is dependent on the quality
and completeness of the copy made available to ProQuest.

Distributed by ProQuest LLC (2021).
Copyright of the Dissertation is held by the Author unless otherwise noted.

This work may be used in accordance with the terms of the Creative Commons license
or other rights statement, as indicated in the copyright statement or in the metadata
associated with this work. Unless otherwise specified in the copyright statement
or the metadata, all rights are reserved by the copyright holder.

This work is protected against unauthorized copying under Title 17,
United States Code and other applicable copyright laws.

Microform Edition where available © ProQuest LLC. No reproduction or digitization
of the Microform Edition is authorized without permission of ProQuest LLC.

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346 USA