Towards an understanding of neural networks : mean-field incursions

Marylou Gabrié

▶ **To cite this version:**

Marylou Gabrié. Towards an understanding of neural networks : mean-field incursions. Mathematical Physics [math-ph]. Université Paris sciences et lettres, 2019. English. NNT : 2019PSLEE035 . tel-02921539

HAL Id: tel-02921539
https://theses.hal.science/tel-02921539v1

Submitted on 25 Aug 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

PSL
UNIVERSITÉ PARIS

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PSL

Préparée à l'École Normale Supérieure

Towards an understanding of neural networks: Mean-field incursions

Soutenue par
Marylou Gabrié
Le 20 septembre 2019

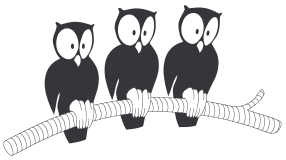
École doctorale n°564
Physique en Île-de-France

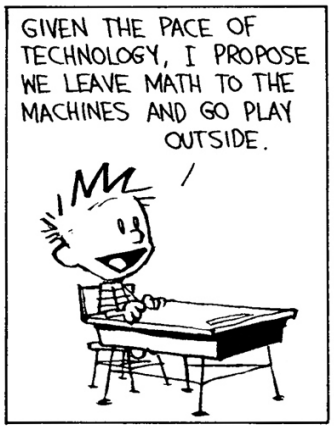
Spécialité
Physique Théorique

Composition du jury :

Giulio Biroli LPENS	*Examinateur, Président du jury*
Florent Krzakala LPENS	*Directeur de thèse*
Yue Lu Harvard University	*Rapporteur*
Matteo Marsili ICTP	*Examinateur*
Manfred Opper TU Berlin	*Rapporteur*
Lenka Zdeborová IPHT	*Invitée*

ENS | PSL





My greatest concern was what to call it. I thought of calling it 'information,' but the word was overly used, so I decided to call it 'uncertainty.' When I discussed it with John von Neumann, he had a better idea. Von Neumann told me, "You should call it entropy, for two reasons. In the first place your uncertainty function has been used in statistical mechanics under that name, so it already has a name. In the second place, and more importantly, no one really knows what entropy really is, so in a debate you will always have the advantage."

Claude Shannon

Contents

Acknowledgements v

Foreword vii

Index of notations and abbreviations xi

I Background 1

1 Machine learning with neural networks and mean-field approximations (Context and motivation) 3
- 1.1 Neural networks for machine learning . . . 3
  - 1.1.1 Supervised learning . . . 3
  - 1.1.2 Unsupervised learning . . . 6
- 1.2 A brief history of mean-field methods for neural networks . . . 9

2 Statistical inference and statistical physics (Fundamental theoretical frameworks) 11
- 2.1 Statistical inference . . . 11
  - 2.1.1 Statistical models representations . . . 11
  - 2.1.2 Some inference questions in neural networks for machine learning . . . 13
  - 2.1.3 Challenges in inference . . . 15
- 2.2 Statistical physics of disordered systems . . . 16

3 Selected overview of mean-field treatments: free energies and algorithms (Techniques) 19
- 3.1 Naive mean-field . . . 19
  - 3.1.1 Variational derivation . . . 19
  - 3.1.2 When does naive mean-field hold true? . . . 20
- 3.2 Thouless Anderson and Palmer equations . . . 21
  - 3.2.1 Outline of the derivation . . . 21
  - 3.2.2 Illustration on the Boltzmann machine and important remarks . . . 22
  - 3.2.3 Generalizing the Georges-Yedidia expansion . . . 23
- 3.3 Belief propagation and approximate message passing . . . 23
  - 3.3.1 Generalized linear model . . . 23
  - 3.3.2 Belief Propagation . . . 24
  - 3.3.3 (Generalized) approximate message passing . . . 25
- 3.4 Replica method . . . 29
  - 3.4.1 Steps of a replica computation . . . 29
  - 3.4.2 Assumptions and relation to other mean-field methods . . . 31
- 3.5 Extensions of interest for this thesis . . . 32
  - 3.5.1 Streaming AMP for online learning . . . 32
  - 3.5.2 A special special case of GAMP: Cal-AMP . . . 32
  - 3.5.3 Algorithms and free energies beyond i.i.d. matrices . . . 33
  - 3.5.4 Model composition . . . 36

II Contributions 37

4 Mean-field inference for (deep) unsupervised learning 39
- 4.1 Mean-field inference in Boltzmann machines . . . 39
  - 4.1.1 Georges-Yedidia expansion for binary Boltzmann machines . . . 39

4.1.2 Georges-Yedidia expansion for generalized Boltzmann machines 40
4.1.3 Application to RBMs and DBMs 44
4.1.4 Adaptive-TAP fixed points for binary BM 46
4.2 Applications to Boltzmann machine learning with hidden units 47
4.2.1 Mean-field likelihood and deterministic training 48
4.2.2 Numerical experiments 50
4.3 Application to Bayesian reconstruction 55
4.3.1 Combining CS and RBM inference 55
4.3.2 Numerical experiments 56
4.4 Perspectives 57

5 Mean-field inference for information theory in deep supervised learning 61
5.1 Mean-field entropy for multi-layer models 61
5.1.1 Extension of the replica formula to multi-layer networks 61
5.1.2 Comparison with non-parametric entropy estimators 64
5.2 Mean-field information trajectories over training of deep networks 65
5.2.1 Tractable deep learning models 66
5.2.2 Training experiments 68
5.3 Further investigation of the information theoretic analysis of deep learning 72
5.3.1 Mutual information in noisy trainings 72
5.3.2 Discussion 74

6 Towards a model for deep Bayesian (online) learning 77
6.1 Cal-AMP revisited 77
6.1.1 Derivation through AMP on vector variables 78
6.1.2 State Evolution for Cal-AMP 82
6.1.3 Online algorithm and analysis 86
6.2 Experimental validation on gain calibration 87
6.2.1 Setting and update functions 87
6.2.2 Offline results 89
6.2.3 Online results 90
6.3 Matrix factorization model and multi-layer networks 93
6.3.1 Constrained matrix factorization 93
6.3.2 Multi-layer vectorized AMP 93
6.3.3 AMP for constrained matrix factorization 94
6.4 Towards the analysis of learning in multi-layer neural networks 97

Conclusion and outlook 99

Appendix 103
A Vector Approximate Message Passing for the GLM 103
B Update functions for constrained matrix factorization 104

Bibliography 107

Acknowledgements

For this dissertation:

For accepting to read and report my work, I warmly thank Manfred Opper and Yue Lu, as well as Matteo Marsili and Giulio Biroli.

Pour leurs suggestions, leurs relectures ou leur 'baby-sitting' pendant la rédaction de ce manuscrit: merci à Adrien, Benjamin, Cédric, Florent, Frédéric, Guilhem, Lauriane, Lenka et Maxence.

Pour ces années :

Un grand merci à Florent tout d'abord, pour m'avoir embarquée sur ce joli sujet, pour tout ce qu'il a pu m'expliquer sur des coin de cahier et de tableau, pour m'avoir écoutée répéter les mêmes présentations tant de fois, pour m'avoir permis de voyager et découvrir autant de lieux et de personnes. Avec Lenka, Guilhem, Léon et Giulio, merci aussi pour leurs encouragements, leur confiance et leurs conseils.

J'ai aussi eu la chance de profiter de la collaboration et de l'enthousisasme des membres du Sphinx-Smile: Francesco, Jean, Alaa, Christian, Thibault, Alejandro, Laura, Alexis, Jonathan, Alia, Antoine, Benjamin, Sebastian, Stefano, Antoine, Bruno, et les nouveaux, merci à vous ! Plus particulièrement, merci aux très experts Eric et Andre avec qui j'ai partagé les projets présentés dans cette thèse.

Je remercie infiniment les membres distinguées du DJFP, Danijela, Lauriane et Sophie, pour leur amitié, et leurs oreilles attentives à mes interrogations sur la thèse et sur la vie. Pour cela je remercie aussi Dimitri, Frédéric, Benoît, Gabriel et Levent.

Finalement merci à Manon, Clara et Mathilde que je sais toujours derrière moi. Merci à mes parents, Pascale et Benoît, de m'avoir transmis la motivation et la curiosité, et de toujours aider mes projets tant qu'ils le peuvent. Merci enfin à Rosalie d'être toujour chat, d'être encore et depuis toujours là.

Foreword

With the continuous improvement of storage techniques, the amount of available data is currently growing exponentially. While it is not humanly feasible to treat all the data created, *machine learning* is one possible response. It is a class of algorithms that allows to automatically infer structure in large data sets. In particular, *deep learning* methods, based on neural networks, have drastically improved performances in key fields of artificial intelligence such as image processing, speech recognition or text mining. A good review of the first successes of this technology published in 2015 is [79]. A few years later, the current state-of-the-art of this very active line of research is difficult to envision globally. However, the complexity of deep neural networks remains an obstacle to the understanding of their great efficiency. Made of many layers, each of which constituted of many neurons with a collection of parameters, the set of variables describing completely a typical neural network is impossible to only visualize. Instead we must consider aggregated quantities to characterize these models and hopefully help and explain the learning process. The first open challenge is therefore to identify the relevant observables to focus on. Often enough, what seems interesting is also what is hard to calculate. In the high-dimensional regime we need to consider, exact analytical solutions are unknown most of the time and numerical computations are ruled out. Hence we need to find ways of approximations that are simultaneously simple enough to be tractable and fine enough to retain interesting features.

In the context where dimensionality is an issue, physicists have experimented that macroscopic behaviors are typically well described by the theoretical limit of infinitely large systems. Under this *thermodynamic* limit, the statistical physics of disordered systems offers powerful frameworks of approximation called *mean-field theories*. Nevertheless, the models studied by physicists usually include assumptions of randomness and (statistical) homogeneities. Whereas in machine learning, interactions between neurons, that are chosen by training algorithms with respect to specific data sets, have no reason a priori to be as regular as mean-field theories assume. Thus, it is not straightforward that methods suited in theoretical physics can be applied to deep learning. In this dissertation we discuss cases where the bridge between these two disciplines can be crossed. Thanks to this approach we propose new learning procedures and contribute to the theoretical characterization of other popular, yet poorly understood, training algorithms.

Organization of the manuscript

The first part of this dissertation covers the basic theoretical concepts that are relevant for the presentation of our original contributions.

In Chapter 1 we wish to clarify the context and motivation of our work by giving a minimal introduction to methods and challenges in deep learning that will be of particular interest to the physicist reader. This Chapter is also the occasion to recall some of the historical connections between learning and mean-field theories. In Chapter 2, we introduce the conceptual frameworks of statistical inference and statistical physics that are underlying our approach. Finally, we present mean-field methods in Chapter 3 with several goals in mind. First, we wish to introduce the different techniques of derivation that will be used in the second part of the dissertation. Our second aim is to provide intuitions on where the approximation lies in mean-field computations, or in other words on what they neglect. Third, we attempt to clarify how the different methods are related, when they are equivalent and when they complement. Lastly, we present some recent advances on which the contributions presented in this dissertation are building.

The second part is divided into three Chapters, presenting different directions of research in the training and analysis of neural networks.

In Chapter 4, we present our publications on the unsupervised learning of Boltzmann machines with hidden units:

[42] Marylou Gabrié, Eric W. Tramel, and Florent Krzakala. Training Restricted Boltzmann Machines via the Thouless-Anderson-Palmer Free Energy. Advances in *Advances in Neural Information Processing Systems 28*, pages 640—-648, jun 2015.

[151] Eric W. Tramel, Marylou Gabrié, Andre Manoel, Francesco Caltagirone, and Florent Krzakala. Deterministic and Generalized Framework for Unsupervised Learning with Restricted Boltzmann Machines. *Physical Review X*, 8(4):041006, oct 2018.

[150] Eric W. Tramel, Andre Manoel, Francesco Caltagirone, Marylou Gabrié, and Florent Krzakala. Inferring sparsity: Compressed sensing using generalized restricted Boltzmann machines. In *2016 IEEE Information Theory Workshop (ITW)*, pages 265–269. IEEE, sep 2016.

[148] Eric W. Tramel, Marylou Gabrié, and Florent Krzakala. Boltzmann.jl, https://github.com/sphinxteam/Boltzmann.jl, 2015.

Our main contributions consist in (i) the derivation of the generalization of the Plefka expansion for mean-field inference to real-valued variables for pairwise models recovering Approximate Message Passing and (ii) the design of a training algorithm for Restricted and Deep Boltzmann Machines that we validate in numerical experiments for both binary and real-valued data sets and for which we provide a public implementation in Julia. We also discuss results obtained with E. W. Tramel and A. Manoel regarding how the mean-field framework allows to improve reconstruction performance in Bayesian inference by using Restricted Boltzmann Machines as priors.

In Chapter 5, we discuss our results around an information theoretic approach to the generalization puzzle in the supervised learning of deep neural networks. While most of our contributions are published in the following reference, we also present here further considerations formulated in collaboration with Léon Bottou.

[40] 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 *Advances in Neural Information Processing Systems 31*, pages 1826—-1836, mar 2018.

[41] Marylou Gabrié, Andre Manoel, Gabriel Samain, and Florent Krzakala. Learning Synthetic Data https://github.com/marylou-gabrie/learning-synthetic-data, 2018.

[83] Andre Manoel, Marylou Gabrié, and Florent Krzakala. Deep Neural Networks Entropy from Replicas https://github.com/sphinxteam/dnner, 2018.

Our first contribution is (i) the derivation of an asymptotic formula for entropies in multi-layer neural networks with rotationally invariant weight matrices and its implementation as a python package. Additionally, (ii) we design a teacher-student scenario allowing arbitrary depth, non-linearities and non-trivial synthetic data distributions, also implemented as a python package, for which the proposed entropy formula is applicable throughout the learning. Lastly, (iii) we discuss via numerical experiments, using the proposed formula and other non-parametric methods of estimation, issues in the considered information theoretic approach to deep learning theory.

Finally in Chapter 6 we discuss unpublished works in signal processing that we believe offer a promising direction to approach an analysis of Bayesian learning in multi-layer neural networks under a teacher-student scenario. More precisely we revisit the calibration-approximation message passing algorithm (Cal-AMP) [130, 131] with the following contributions: (i) we provide an original derivation of the corresponding State Evolution, (ii) we provide a generalization of the algorithm and State Evolution in the case of online learning and (iii) we provide numerical tests of the last two results in the particular case of gain calibration. To conclude, (iv) we explain how this algorithm and analysis can help study the Bayesian learning, under a specific weight constraint, of multi-layer neural networks both for supervised and unsupervised learning.

Additional work not covered in this dissertation

This dissertation does not cover an additional publication, using again mean-field methods yet in the context of Constrained Satisfaction Problems (CSPs), unrelated to deep learning,

[39] Marylou Gabrié, Varsha Dani, Guilhem Semerjian, and Lenka Zdeborová. Phase transitions in the q-coloring of random hypergraphs. *Journal of Physics A: Mathematical and Theoretical*, 50(50), 2017.

In this paper, we characterize the set of solutions for random instances of the specific constrained satisfaction problem of hypergraph q-coloring. The publication briefly reviews the statistical physics approach to random CSPs. Results are obtained with a mean-field method that we do not present in this manuscript: the one-step-replica-symmetry-breaking (1RSB) cavity formalism. From its novel application to hypergraph q-coloring we make the following contributions: (i) we compute the thresholds for various phase transitions undergone by the set of solutions in this problem. Focusing on the case of bicoloring ($q = 2$) studied in previous works, (ii) we find that for certain graph ensembles the colorability threshold is actually not given by the (1RSB) analysis due to an instability that was missed before, (iii) we also show that distinct 1RSB solutions can coexist in the colorable region. Lastly, (iv) we derive asymptotic expansions for phase transition thresholds with respect to parameters of the random graph ensembles.

Index of notations and abbreviations

[N]	Set of integers from 1 to N
$\delta(\cdot)$	Dirac distribution
$\sigma(x) = (1 + e^{-x})^{-1}$	Sigmoid
$\text{relu}(x) = \max(0, x)$	Rectified Linear Unit
$\underline{\underline{X}}$	Matrix
$\underline{x}$	Vector
$\underline{\underline{I}}_N \in \mathbb{R}^{N \times N}$	Identity matrix
$\langle \cdot \rangle$	Boltzmann average
$\text{O}(N) \subset \mathbb{R}^{N \times N}$	Orthogonal ensemble
AMP	Approximate message passing
BP	Belief Propagation
CD	Contrastive Divergence
DAG	Directed Acyclic Graph
DBM	Deep Boltzmann Machine
GAMP	Generalized Approximate message passing
GLM	Generalized Linear Model
i.i.d.	independent identically distributed
r-BP	relaxed Beleif Propagation
RS	Replica Symmetric
RBM	Restricted Boltzmann Machine
SE	State Evolution
SGD	Stochastic Gradient Descent
SK	Sherrington-Kirkpatrick
TAP	Thouless Anderson Palmer
VAE	Variational Autoencoder
VAMP	Vector Approximate message passing

Part I

Background

1 Machine learning with neural networks and mean-field approximations

(Context and motivation)

This chapter provides the fundamental concepts in machine learning that will be used throughout this thesis. A comprehensive reference is [49]. We also take this chapter as an opportunity to introduce the current challenges in the understanding of deep learning. With these first elements of context, we briefly review the historical connections between mean-field methods coming from statistical physics and neural networks used for machine learning.

1.1 Neural networks for machine learning

Machine learning is traditionally divided into three classes of problems: supervised, unsupervised and reinforcement learning. For all of them, the advent of deep learning techniques, relying on deep neural networks, has brought great leaps forward in terms of performance and opened the way to new applications. Nevertheless, the utterly efficient machinery of these algorithms remains full of theoretical puzzles. In this Section we quickly cover key concepts in supervised and unsupervised machine learning that will be useful to understand the context of this thesis and motivate its contributions.

1.1.1 Supervised learning

Learning under supervision

Supervised learning aims at discovering systematic input to output mappings from examples. For instance, classification is a typical supervised learning problem. For example, from a set of pictures of cats and dogs labelled accordingly, the goal is to find a function able to predict in any new picture the species of the displayed pet.

In practice, the *training set* is a collection of P example pairs $\mathcal{D} = \{\underline{x}_{(k)}, \underline{y}_{(k)}\}_{k=1}^{P}$ from an input data space $\mathcal{X} \subseteq \mathbb{R}^N$ and an output data space $\mathcal{Y} \subseteq \mathbb{R}^M$. Formally, they are assumed to be i.i.d. samples from a joint distribution $p(\underline{x}, \underline{y})$. The predictor h is chosen by a training algorithm from a *hypothesis class*, a set of functions from $\mathcal{X}$ to $\mathcal{Y}$, so as to minimize the error on the training set. This error is formalized as the *empirical risk*

$$\hat{\mathcal{R}}(h, \ell, \mathcal{D}) = \frac{1}{P} \sum_{k=1}^{P} \ell(\underline{y}_{(k)}, h(\underline{x}_{(k)})), \tag{1.1}$$

where the definition involves a loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ measuring differences in the output space. This learning objective nevertheless does not guarantee *generalization*, i.e. the ability of the predictor h to be accurate on inputs $\underline{x}$ that are not in the training set. It is a surrogate for the ideal, but unavailable, *population risk*

$$\mathcal{R}(h, \ell) = \mathbb{E}_{\underline{x},\underline{y}} \Big[\ell(\underline{y}, h(\underline{x})) \Big] = \int_{\mathcal{X},\mathcal{Y}} \mathrm{d}\underline{x}\, \mathrm{d}\underline{y}\, p(\underline{x}, \underline{y}) \ell(\underline{y}, h(\underline{x})), \tag{1.2}$$

expressed as an expectation over the joint distribution $p(\underline{y}, \underline{x})$. The different choices of hypothesis classes and training algorithms yield the now crowded zoo of supervised learning algorithms.

Representation ability of deep neural networks

In the context of supervised learning, deep neural networks enter the picture in the quality of parametrized hypothesis class. Let us first quickly recall the simplest network, the *perceptron*. It

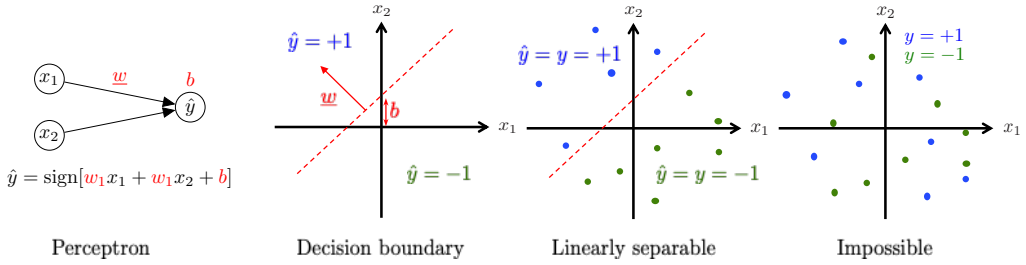


Figure 1.1: Let's assume we wish to classify data points $\underline{x} \in \mathbb{R}^2$ with labels $y = \pm 1$. We choose as an hypothesis class the perceptron sketched on the left with sign activation. For given weight vector $\underline{w}$ and bias b the plane is divided by a decision boundary assigning labels. If the training data are linearly separable, then it is possible to perfectly predict all the labels with the perceptron, otherwise it is impossible.

corresponds to single neuron, so a function from $\mathbb{R}^N$ to $\mathcal{Y} \subset \mathbb{R}$ applying an activation function f to a weighted sum of its inputs shifted by a bias $b \in \mathbb{R}$,

$$\hat{y} = h_{\underline{w},b}(\underline{x}) = f(\underline{w}^\top \underline{x} + b) \tag{1.3}$$

where the weights are collected in the vector $\underline{w} \in \mathbb{R}^N$. From a practical standpoint this very simple model can only solve the classification of linearly separable groups (see figure 1.1). Yet from the point of view of learning theory, it has been the starting point of a rich statistical physics literature as will be discussed in section 1.2.

Combining several neurons into networks defines more complex functions. The universal approximation theorem [30, 61] proves that the following two-layer network architecture can approximate any well-behaved function with a finite number of neurons,

$$\hat{y} = h_\theta(\underline{x}) = \underline{w}^{(2)\top} f(\underline{\underline{W}}^{(1)} \underline{x} + \underline{b}) = \sum_{\alpha=1}^{M} w_\alpha^{(2)} f(\underline{w}_\alpha^{(1)\top} \underline{x} + b_\alpha), \qquad \theta = \{\underline{w}^{(2)}, \underline{\underline{W}}^{(1)}, \underline{b}\} \tag{1.4}$$

for f a bounded, non-constant, continuous scalar function, acting component-wise. In the language of deep learning this network has one hidden layer of M units. Input weight vectors $w_\alpha^{(1)} \in \mathbb{R}^N$ are collected in a weight matrix $\underline{\underline{W}}^{(1)} \in \mathbb{R}^{M \times N}$. Here, and in the following, the notation θ is used as short for the collection of adjustable parameters. The universal approximation theorem is a strong result in terms of representative power of neural networks but it is not constructive. It does not tell us what is the minimal M to approximate a given function, nor what are the optimal values of the parameters $\underline{w}^{(2)}, \underline{\underline{W}}^{(1)}$ and $\underline{b}$. These questions remain unsolved to this date and practice is widely led by empirical considerations.

In applications, neural networks with multiple hidden layers were found more effective. A generic neural network of *depth* L is the function

$$\underline{\hat{y}} = h_\theta(\underline{x}) = f(\underline{\underline{W}}^{(L)} f(\underline{\underline{W}}^{(L-1)} \cdots f(\underline{\underline{W}}^{(1)} \underline{x} + \underline{b}^{(1)}) \cdots + \underline{b}^{(L-1)}) + \underline{b}^{(L)}), \tag{1.5}$$

$$\theta = \{\underline{\underline{W}}^{(l)} \in \mathbb{R}^{N_l \times N_{l-1}}, \underline{b}^{(l)} \in \mathbb{R}^{N_l} \; ; \; l = 1 \cdots L\}, \tag{1.6}$$

where $N_0 = N$ is the dimension and the input and $N_L = M$ is the dimension of the output. The architecture is fixed by specifying the number of neurons, or *width*, of the hidden layers $\{N_l\}_{l=1}^{L-1}$. The latter can be denoted $\underline{t}^{(l)} \in R^{N_l}$ and follow the recursion

$$\underline{t}^{(1)} = f(\underline{\underline{W}}^{(1)} \underline{x} + \underline{b}^{(1)}), \tag{1.7}$$

$$\underline{t}^{(l)} = f(\underline{\underline{W}}^{(l)} \underline{t}^{(l-1)} + \underline{b}^{(l)}), \quad l = 2 \cdots L-1, \tag{1.8}$$

$$\underline{\hat{y}} = f(\underline{\underline{W}}^{(L)} \underline{t}^{(L-1)} + \underline{b}^{(L)}). \tag{1.9}$$

Fixing the activation functions and the architecture of a neural network defines an hypothesis class. It is crucial that activations introduce non-linearities; the most common are the hyperbolic tangent tanh and the rectified linear unit defined as $\text{relu}(x) = \max(0, x)$. Note that it is also possible, albeit

uncommon in supervised learning applications, to define stochastic neural networks by using noisy activation functions.

An originally proposed intuition for the advantage of depth is that it enables to treat the information in a hierarchical manner; either looking at different scales in different layers, or learning more and more abstract representations [13]. Nevertheless, there is no theory to this day clarifying why in practice 'the deeper the better'.

Neural network training

Given an architecture defining h_θ, the supervised learning objective is to minimize the empirical risk $\hat{\mathcal{R}}$ with respect to the parameters θ. This optimization problems lives in the dimension of the number of parameters which can range from tens to millions. The idea underlying the majority of training algorithms is to perform a gradient descent (GD) from a random initialization of the parameters:

$$\theta_0 \sim p_{\theta_0}(\theta_0) \tag{1.10}$$

$$\theta_{t+1} \leftarrow \theta_t - \eta \nabla_\theta \hat{\mathcal{R}} = \theta_t - \eta \frac{1}{P} \sum_{k=1}^{P} \nabla_\theta \ell \left(\underline{y}_{(k)}, h_{\theta_t} \left(\underline{x}_{(k)} \right) \right) . \tag{1.11}$$

The parameter η is the learning rate, controlling the size of the step in the direction of decreasing gradient per iteration. The computation of the gradients can be performed in time scaling linearly with depth by applying the derivative chain-rule leading to the *back-propagation* algorithm [49]. A popular alternative to gradient descent is stochastic gradient descent (SGD) where the sum over the gradients for the entire training set is replaced by the sum over a small number of samples, randomly selected at each step [121, 19].

During the training iterations, one typically monitors the *training error* (another name for the empirical risk given a training data set) and the *validation error*. The latter corresponds to the empirical risk computed on a set of points held-out from the training set, the validation set, to assess the generalization ability of the model either along the training or in order to select hyperparameters of training such as the value of the learning rate. A posteriori, the performance of the model is judged from the *generalization error*, which is evaluated on the never seen *test set*. While two different training algorithms (e.g. GD vs SGD) may achieve zero training error, they may differ in the level of generalization they typically reach.

Open questions and challenges

Building on the fundamental concepts presented in the previous paragraphs, practitioners managed to bring deep learning to unanticipated performances in the automatic processing of images, speech and text (see [79] for a few years old review). Still, many of the greatest successes in the field of neural network where obtained using ingenious tricks while many fundamental theoretical questions at a very basic level remain unresolved.

Regarding the optimization first, (S)GD training generally discovers parameters close to zero risk. Yet gradient descent is guaranteed to converge to the neighborhood of a global minimum only for a convex function and is otherwise expected to get stuck in a local minimum. Therefore efficiency of gradient based optimization is a priori a paradox given the empirical risk $\hat{R}$ is non-convex in the parameters θ. Second, the representation capacity of deep learning models is also poorly understood. Results in the literature that relate the size and architecture of a network to a measure of its ability to learn are too far from realistic settings to guide choices of practitioners. On the one hand, traditional bounds in statistics, considering worst cases, appear overly pessimistic. On the other hand, historical statistical physics analyses of learning, briefly reviewed in Section 1.2, only concern simple architectures and synthetic data. This lack of theory results in potentially important waste: in terms of time lost by engineers in trial and error to optimize their solution, and in terms of electrical resources used to train and re-train possibly oversized networks and store potentially un-necessarily large training data sets. Lastly, the generalization ability of deep neural networks trained by gradient descent is still unaccounted for. The size of training data sets is limited by the cost of human labelling. Thus training a deep and wide network amounts

in practice to fitting a model of millions of degrees of freedom against a somehow relatively small amount of data points. Nevertheless it does not systematically lead to *overfitting*. The resulting neural networks can have good predictions both on inputs seen during training and on new inputs.

The success of deep learning, beyond these apparent theoretical puzzles,certainly lies in the interplay of advantageous properties of training algorithms, the neural network hypothesis class and structures in typical data (e.g. real images, conversations). Disentangling the role of the different ingredients is a very active line of research (see [153] for a review) among which this thesis is a modest contribution to. In particular, Chapter 5 touches the question of the generalization capability of neural networks.

1.1.2 Unsupervised learning

Density estimation and generative modelling

The goal of unsupervised learning is to directly extract structure from data. Compared to the supervised learning setting the training data set is made of a set of example inputs $\mathcal{D} = \{\underline{x}_{(k)}\}_{k=1}^{P}$ without corresponding outputs. A simple example of unsupervised learning is clustering, consisting in the discovery of unlabelled subgroups in the training data. Most unsupervised learning algorithms either implicitly or explicitly adopt a probabilistic viewpoint and implement *density estimation*. The idea is to approximate the true density $p(\underline{x})$ from which the training data was sampled by the closest (in various senses) element among a family of parametrized distributions over the input space $\{p_\theta(.),\ \theta \in \mathbb{R}^{N_\theta}\}$. The selected p_θ is then a model of the data. Structural properties of the data can sometimes be inferred from it, such as dependencies according to its factorization (see Section 2.1.1). If the model p_θ is easy to sample, it can also be used to generate new inputs comparable to the training data points - which leads to the terminology of *generative models*. In this context, *unsupervised deep learning* exploits the representational power of deep neural networks to create sophisticated candidate p_θ.

A common formalization of the learning objective is to maximize the *likelihood*, defined as the logarithm of the probability of i.i.d. draws from the model p_θ to have generated the training data $\mathcal{D} = \{\underline{x}_{(k)}\}_{k=1}^{P}$,

$$\max_\theta \prod_{k=1}^{P} p_\theta(\underline{x}_{(k)}) \quad \Longleftrightarrow \quad \max_\theta \sum_{k=1}^{P} \log p_\theta(\underline{x}_{(k)}). \tag{1.12}$$

The second logarithmic additive formulation is generally preferred. It can be interpreted as the minimization of the Kullback-Leibler divergence between the empirical distribution $p_{\mathcal{D}}(\underline{x}) = \sum_{k=1}^{P} \delta(\underline{x} - \underline{x}_{(k)})/P$ and the model p_θ:

$$\min_\theta \mathrm{KL}(p_{\mathcal{D}}||p_\theta) = \min_\theta \int \mathrm{d}\underline{x}\, p_{\mathcal{D}}(\underline{x}) \log \frac{p_{\mathcal{D}}(\underline{x})}{p_\theta(\underline{x})} \quad \Longleftrightarrow \quad \max_\theta \sum_{k=1}^{P} \log p_\theta(\underline{x}_{(k)})\,, \tag{1.13}$$

although considering the divergence with the discrete empirical measure is slightly abusive.

The detail of the optimization algorithm here depends on the specification of p_θ. As we will see, the likelihood in itself is often intractable and learning consists in a gradient ascent on at best a lower bound, otherwise an approximation, of the likelihood.

A few years ago, an alternative strategy called adversarial training was introduced [50] leading to a remarkable quality of samples produced by generative models. However, the training of models appearing in this thesis, presented in the following paragraphs, will fall under the classical maximum-likelihood paradigm.

Deep Variational Autoencoders

A variational autoencoder (VAE) [71, 120] defines a density p_θ obtained by propagating a simple distribution through a deep neural network. It can be formalized by introducing a latent variable $\underline{z} \in \mathbb{R}^N$ and a deep neural network h_θ similar to (1.5) of input dimension N:

$$\underline{z} \sim p_z(\underline{z}) \tag{1.14}$$
$$\underline{x} \sim p_\theta(\underline{x}|\underline{z}) = p_{\mathrm{out}}(\underline{x}|h_\theta(\underline{z})), \tag{1.15}$$

where p_z is typically a factorized distribution on $\mathbb{R}^N$ easy to sample (e.g. a standard normal distribution), and $p_{\text{out}}(.|h_\theta(\underline{z}))$ is for instance a multivariate Gaussian distribution with mean and covariance that are function of $h_\theta(\underline{z})$. The computation of the likelihood of one training sample $\underline{x}_{(k)}$ requires then the marginalization over the latent variable $\underline{z}$,

$$p_\theta(\underline{x}) = \int \mathrm{d}\underline{z}\; p_{\text{out}}(\underline{x}|h_\theta(\underline{z}))p_z(\underline{z}). \tag{1.16}$$

This multidimensional integral cannot be performed analytically in the general case. It is also hard to evaluate numerically as it does not factorize over the dimensions of $\underline{z}$ which are mixed by the neural network h_θ. Yet a lower bound on the log-likelihood can be defined by introducing a tractable conditional distribution $q(\underline{z}|\underline{x})$ that will play the role of an approximation of the intractable *posterior* distribution $p_\theta(\underline{z}|\underline{x})$ implicitly defined by the model:

$$\begin{aligned}
\log p_\theta(\underline{x}) &= \int \mathrm{d}\underline{z}\; q(\underline{z}|\underline{x}) \log p_\theta(\underline{x}) \qquad (1.17)\\
&= \int \mathrm{d}\underline{z}\; q(\underline{z}|\underline{x}) \log \frac{p_\theta(\underline{x},\underline{z})}{p_\theta(\underline{z}|\underline{x})}\\
&= \int \mathrm{d}\underline{z}\; q(\underline{z}|\underline{x}) \log \frac{p_\theta(\underline{x},\underline{z})}{p_\theta(\underline{z}|\underline{x})} \times \frac{q(\underline{z}|\underline{x})}{q(\underline{z}|\underline{x})}\\
&= \int \mathrm{d}\underline{z}\; q(\underline{z}|\underline{x}) \left[-\log q(\underline{z}|\underline{x}) + \log p_\theta(\underline{x},\underline{z})\right] + \mathrm{KL}(q(\underline{z}|\underline{x})||p_\theta(\underline{z}|\underline{x}))\\
&\geq \int \mathrm{d}\underline{z}\; q(\underline{z}|\underline{x}) \left[-\log q(\underline{z}|\underline{x}) + \log p_\theta(\underline{x},\underline{z})\right] = \mathrm{LB}(q,\theta,\underline{x}) \qquad (1.18)
\end{aligned}$$

where the last inequality comes from the fact that a KL-divergence is always positive; it is replaced by an equality if and only if $q(\underline{z}|\underline{x}) = p_\theta(\underline{z}|\underline{x})$. Provided q has a tractable expression and is easy to sample, the remaining lower bound $\mathrm{LB}(q,\theta,\underline{x})$ can be approximated by a Monte Carlo method. Maximum likelihood learning is then approached by the maximization of the lower bound $\mathrm{LB}(q,\theta,\underline{x})$, which requires in practice to parametrize the tractable posterior $q = q_\phi$ as well, typically with a neural network. In order to adjust the parameters θ and ϕ through gradient descent a last reformulation is nevertheless necessary. The expectation over q_ϕ estimated via Monte Carlo indeed involves the ϕ parameters. To disentangle this dependence $\underline{z} \sim q_\phi(\underline{z}|\underline{x})$ must be rewritten as $\underline{z} = g(\epsilon,\phi,\underline{x})$ with $\epsilon \sim p_\epsilon(\epsilon)$ independent of ϕ. This so-called re-parametrization trick allows to compute the gradients of $\mathrm{LB}(q_\phi,\theta,\underline{x})$ with respect to θ and ϕ.

In this thesis we will mention variational autoencoders in the perspectives of Chapter 4 and Chapter 6. We will discuss a different proposition to approximate the likelihood and the alternative Bayesian learning objective involving prior knowledge on the parameters θ.

Restricted and Deep Boltzmann Machines

Models described in the preceding paragraphs comprised only *feed forward* neural networks. In feed forward neural networks, the state or value of successive layers is determined following the recursion (1.7) - (1.9), in one pass from inputs to outputs. Boltzmann machines instead involve *undirected* neural networks which consist of stochastic neurons with symmetric interactions. The probability associated to a neuron state is a function of neighboring neurons, themselves reciprocally function of the first neuron. Sampling a configuration therefore requires an equilibration in the place of a simple forward pass.

A Restricted Boltzmann Machine (RBM) [3, 139] with M hidden neurons defines a joint distribution over an input (or visible) layer $\underline{x} \in \{0,1\}^N$ and a hidden layer $\underline{t} \in \{0,1\}^M$

$$p_\theta(\underline{x},\underline{t}) = \frac{1}{\mathcal{Z}} e^{\underline{a}^\top \underline{x} + \underline{b}^\top \underline{t} + \underline{x}^\top \underline{\underline{W}} \underline{t}}, \qquad \theta = \{\underline{\underline{W}},\underline{a},\underline{b}\}, \tag{1.19}$$

where $\mathcal{Z}$ is the normalization factor, similar to the partition function of statistical physics. The parametric density model over inputs is then the marginal $p_\theta(\underline{x}) = \sum_{\underline{t}\in\{0,1\}^M} p_\theta(\underline{x},\underline{t})$. Identically to VAEs, RBMs can represent sophisticated distributions at the cost of an intractable likelihood.

Indeed the summation over 2^{M+N} terms in the partition function cannot be simplified by an analytical trick and is only realistically doable for small models.

This intractibility remains a priori an issue for the learning by gradient ascent of the log-likelihood. Given an input training point $\underline{x}_{(k)}$, the derivatives of $\log p(\underline{x}_{(k)})$ with respect to trainable parameters are

$$\nabla_{\underline{a}} \log p(\underline{x}_{(k)}) = \underline{x}_{(k)} - \langle \underline{x} \rangle \,, \tag{1.20a}$$

$$\nabla_{\underline{b}} \log p(\underline{x}_{(k)}) = \langle \underline{t} \rangle_{\underline{x}_{(k)}} - \langle \underline{t} \rangle \,, \tag{1.20b}$$

$$\nabla_{\underline{\underline{W}}} \log p(\underline{x}_{(k)}) = \langle \underline{t}\, \underline{x}^\top \rangle_{\underline{x}_{(k)}} - \langle \underline{t}\, \underline{x}^\top \rangle, \tag{1.20c}$$

where we introduce the notation $\langle . \rangle$ for the average over the RBM measure (1.19) and $\langle . \rangle_{\underline{x}_{(k)}}$ for the average over the RBM measure conditioned on $\underline{x} = \underline{x}_{(k)}$, also called the *clamped* RBM measure. To approximate the gradients, sampling over the clamped measure is actually easy as the hidden neurons are independent when conditioned on the inputs; in other words $p(\underline{t}|\underline{x})$ is factorized. Yet sampling over the complete RBM measure is more involved and requires for example the equilibration of a Markov Chain. It is a costly operation that would need to be repeated at each parameter update in the gradient ascent. Instead it is commonly approximated by *contrastive divergence* (CD) [55], which approximates the $\langle . \rangle$ in (1.20) by the final state of a Monte Carlo Markov chain initialized in $\underline{x}_{(k)}$ and updated for a small number of steps.

RBMs were the first effective generative models using neural networks. They found applications in various domains including dimensionality reduction [57], classification [78], collaborative filtering [125], feature learning [27], and topic modeling [58]. Used for an unsupervised pre-training of deep neural networks layer by layer [56, 14], they also played a crucial role in the take-off of supervised deep learning.

Furthermore, they can be generalized to Deep Boltzmann Machines (DBMs) [123], where several hidden layers are stacked on top of each other. The measure defined by a DBM of depth L is

$$p_\theta(\underline{x}, \underline{t}) = \frac{1}{\mathcal{Z}} e^{\underline{a}^\top \underline{x} + \underline{x}^\top \underline{\underline{W}}^{(1)} \underline{t} + \underline{b}^{(L)\top} \underline{t}^{(L)} + \sum_{l=1}^{L-1} \underline{t}^{(l)\top} \underline{\underline{W}}^{(l+1)} \underline{t}^{(l+1)} + \underline{b}^{(l)\top} \underline{t}^{(l)}} \tag{1.21}$$

$$\theta = \{\underline{a}\,,\ \underline{b}^{(l)},\ \underline{\underline{W}}^{(l)}\ ;\ l = 1 \cdots L\}. \tag{1.22}$$

Similarly to (1.20), the log-likelihood gradient ascent with respect to the parameters involves averages over the DBM measure and the corresponding clamped measure. In this case both of them involve an equilibration to be approximated by a Markov Chain. The contrastive divergence algorithm can be generalized to circumvent this difficulty, but the approximation of the gradients appear all the more less controlled.

Open questions and challenges

Generative models involving neural networks such as VAE, GANs and RBMs have great expressive powers at the cost of not being amenable to exact treatment. Their training, and sometimes even their sampling requires approximations. From a practical standpoint, whether these approximations can be either made more accurate or less costly is an open direction of research. Another important related question is the evaluation of the performance of generative models [122]. To start with the objective function of training is very often itself intractable (e.g. the likelihood of a VAE or a RBM), and beyond this objective, the unsupervised setting does not define a priori a test task for the performance of the model. Additionally, unsupervised deep learning inherits some of the theoretical puzzles already discussed in the supervised learning section. In particular, assessing the difficulty to represent a distribution and select a sufficient minimal model and/or training data set seems today far out of reach of theory.

In this thesis, unsupervised learning models will be discussed in particular in Chapter 4. A novel deterministic framework for RBMs and DBMs with a tractable learning objective is derived, allowing to exploit efficiently their representative power.

1.2 A brief history of mean-field methods for neural networks

The mathematical description of learning implies to deal with a large set of correlated random variables: the data, the network parameters, and the resulting neuron activations. The major hurdle in our theoretical understanding of algorithms based on deep neural networks is the difficulty in performing computations involving these high-dimensional multivariate distributions. This is one aspect of the so-called *curse of dimensionality*. Here, exact computations are almost never possible and numerical treatments are unrealistically costly. Finding good approximations is therefore crucial. Mean-field methods can be understood as one approximation strategy. Originally developed in the statistical physics literature, their applications to machine learning, and in particular to neural networks, already have a long history and significant contributions to their records.

In the 80s and 90s, a series of works pioneered the analysis of learning with neural networks through the statistical physics lense. By focusing on simple models with simple data distributions, and relying on the mean-field method of *replicas*, these papers managed to predict quantitatively important properties such as *capacities* - the number of training data point that could be memorized by a model - or *learning curves* - the generalization error (or population risk) of a model as a function of the size of the training set. This effort was initiated by the study of the Hopfield model [5], an undirected neural network providing associative memory [60]. The analysis of feed forward networks with simple architectures followed (among which [44, 45, 104, 156, 95, 106, 37, 96, 7]). Physicists, accustomed to studying natural phenomena, fruitfully brought the tradition of modelling to their investigation of learning. Their approach was in contrast to the focus of the machine learning theorists on 'worst case' guarantees specific to an hypothesis class and that must hold for any data distribution (e.g. Vapnik-Chervonenkis dimension and Rademacher complexity). The originality of their approach, along with the heuristic character of the derivations of mean-field approximations, may nonetheless explain the minor impact of their theoretical contributions in the machine learning community at the time.

Before the deep learning era, mean-field methods probably had a greater influence in the practice of unsupervised learning through density estimation, where we saw that approximations are almost always necessary. In particular the simplest method of naive mean-field, that will also be our first example in Chapter 3, was easily adopted and even extended by statisticians (see e.g. [154] for a recent textbook and [16] for a recent review). The belief propagation algorithm is another example of a well known mean-field methods by machine learners - it was actually discovered in both communities. Yet, these applications rarely involved neural networks and rather relied on simple probabilistic models such as mixtures of elementary distributions. They also did not take full advantage of the latest, simultaneous developments in statistical physics of the mean-field theory of disordered systems.

In this context, the inverse Ising problem has been a notable exception. The underlying question, rooted in theoretical statistical physics, is to infer the parameters of an Ising model given a set of equilibrium configurations. This corresponds to the unsupervised learning of the parameters of a Boltzmann machine (without hidden units) in the machine learning jargon. The corresponding Boltzmann distribution, with pairwise interactions, is remarkable, not only to physicists, as it is the least biased model under the assumption of fixed first and second moments; in the sense that it maximizes the entropy. For this problem, physicists lead an efficient cross-fertilization and proposed dedicated developments of advanced mean-field methods for applications in other fields, and in particular in bio-physics (see [100] for a recent review). Nevertheless, these works almost never considered the case of Boltzmann machines with hidden units, more common in the machine learning community, especially at the first hours of deep learning.

The language barrier between communities is undoubtedly a significant hurdle delaying the transfer of developments in one field to the other. In machine learning, the potential of the most recent progress of mean-field approximations was already advocated for in a pioneering workshop mixing communities in 1999 [105]. Yet their first widely-used application is possibly the Approximate Message Passing (AMP) algorithm for compressed sensing in 2009 [34] - related to the belief propagation algorithm and even more closely to the TAP equations known in the physics of disordered systems. Meanwhile, there have been much tighter connections between developments in statistical physics and algorithmic solutions in the context of Constraint Satisfaction Problems (CSPs). The very first popular application of advanced mean-field methods outside of physics,

beyond naive mean-field and belief propagation, is probably the survey propagation algorithm [91] in 2002. It borrows from the 1RSB cavity method (that will not be mentioned in this dissertation) to solve efficiently certain types of CSPs.

The great leap forward in the performance of machine learning with neural networks brought by deep learning algorithms, along with the multitude of theoretical and practical challenges it has opened, re-ignited the interest of physicists. While the power of advanced mean-field theories, was mostly unexploited in this context at the time this thesis started, this strategy has yielded over the last couple of years a number of contributions that would be difficult to list exhaustively (some of them were recently reviewed in [21]). In these works, applying mean-field ideas to deep neural networks has served different objectives and required different simplifying assumptions such as considering an infinite size limit, random (instead of learned) weights, or vanishing learning rates. Hence, there is no such a thing as one mean-field theory of deep neural networks and these contributions are rather complementary pieces of solving a great puzzle. In this thesis in particular, we investigated mean-field insights in models being explicitely trained, either on real or synthetic data sets. To this end we notably exploited some recent progress in the statistical physics treatment of disordered systems. In the following Chapters of this First Part we formally introduce the fundamental frameworks and mean-field tools underlying the research presented in the Second Part.

2 Statistical inference and statistical physics

(Fundamental theoretical frameworks)

To study neural networks we will be led to manipulate high-dimensional probability distributions. Statistical inference consists in extracting useful information from these complicated objects. By adopting a probabilistic interpretation of natural systems composed of many elementary components statistical physics is interested in similar questions. In particular the theory of disordered systems, considering different sources of randomness, appears especially relevant. In this Chapter we introduce these important theoretical frameworks underlying the approach adopted in this thesis.

2.1 Statistical inference

Joint probability distributions will be the starting point of the different problems that will interest us in the following, as they arise from the definition of machine learning models or potentially from the choice of a statistical model introduced for the purpose of theory. Inference questions will aim at understanding the properties of these distributions. For instance we may ask what is the expected value of one of the random variable of the model, or whether the realization of a random variable can be recovered from the observations of others.

Note that the practical problem of training neural networks from real data discussed in Chapter 1 does not a priori fit in this perspective as there is a priori no need to assume for a joint probability distribution neither for the training data nor for the learned parameters. However in Section 2.1.2 we will discuss how the learning problem can be cast as inference. Yet it will not be our only concern and we shall consider other inference questions arising naturally from the definition of neural network models.

2.1.1 Statistical models representations

Graphical representations have been developed to represent and efficiently exploit (in)dependencies between random variables encoded in joint probability distributions. In this thesis they will be useful tools to concisely present the model under scrutiny, as well as direct supports for some derivations of inference procedures. Let us briefly present two types of graphical representations.

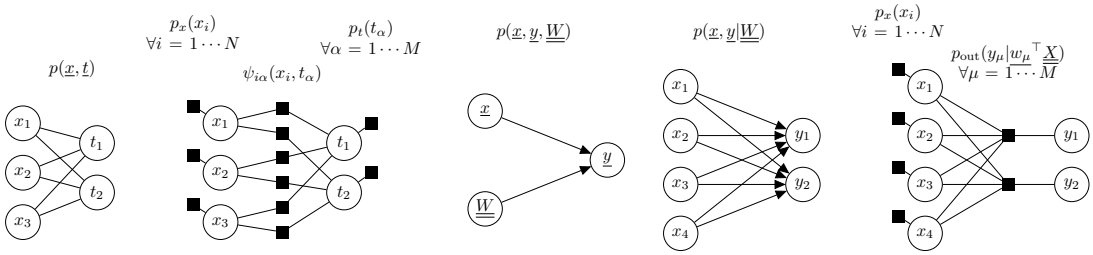
Probabilistic graphical models

Formally, a probabilistic graphical model is a graph $G = (V, E)$ with nodes V representing random variables and edges E representing direct interactions between random variables. In many statistical models of interest, it is not necessary to keep track of all the possible combinations of realizations of the variables as the joint probability distribution can be broken up into factors involving only subsets of the variables. The structure of the connections E reflects this factorization.

There are two main types of probabilistic graphical models: *directed graphical models* (or Bayesian networks) and *undirected graphical models* (or Markov Random Fields). They allow to represent different independencies and factorizations. In the next paragraphs we provide intuitions and review some useful properties of graphical models, a good reference to understand all the facets of this powerful tool is [75].

Undirected graphical models In undirected graphical models the direct interaction between a subset of variables $C \subset V$ is represented by undirected edges interconnecting each pair in C. This fully connected subgraph is called a *clique* and associated with a real positive *potential* functions ψ_C over the variable $\underline{x}_C = \{x_i\}_{i \in C}$ carried by C. The joint distribution over all the variables carried by V, $\underline{x}_V$ is the normalized product of all potentials

$$p(\underline{x}_V) = \frac{1}{\mathcal{Z}} \prod_{C \in \mathcal{C}} \psi_C(\underline{x}_C). \tag{2.1}$$



(a) Restricted Boltzmann Machine **(b)** Feed forward neural network.

Figure 2.1: (a) Undirected probabilistic graphical model (left) and factor graph representation (right). **(b)** Left: Directed graphical model for $p(\underline{x},\underline{y},\underline{\underline{W}})$ without assumptions of factorizations for the channel and priors. Middle: Directed graphical model reflecting factorization assumptions for $p(\underline{x},\underline{y}|\underline{\underline{W}})$. Right: Corresponding factor graph representation.

Example (i): the Restricted Boltzmann Machine,

$$p(\underline{x},\underline{t}) = \frac{1}{\mathcal{Z}} e^{\underline{x}^\top \underline{\underline{W}} \underline{t}} p_x(\underline{x}) p_t(\underline{t}) \tag{2.2}$$

with factorized p_x and p_t is handily represented unsing an undirected graphical model depicted in Figure 2.1a. The corresponding set of cliques is the set of all the pairs with one input unit (indexed by $i = 1\cdots N$) and one hidden unit (indexed by $\alpha = 1\cdots M$), joined with the set of all single units. The potential function are immediately recovered from (2.2),

$$\mathcal{C} = \{\{i\},\{\alpha\},\{i,\alpha\}\,;\, i = 1\cdots N,\, \alpha = 1\cdots M\}\,, \qquad \psi_{i\alpha}(x_i,t_\alpha) = e^{x_i W_{i\alpha} t_\alpha}\,, \tag{2.3}$$

$$p(\underline{x},\underline{t}) = \frac{1}{\mathcal{Z}} \prod_{\{i,\alpha\}\in\mathcal{C}} \psi_{i\alpha}(x_i,t_\alpha) \prod_{i=1}^{N} p_x(x_i) \prod_{\alpha=1}^{M} p_t(t_\alpha). \tag{2.4}$$

It belongs to the subclass of *pairwise* undirected graphical models for which the size of the cliques is at most two.

Undirected graphical models handily encode *conditional independencies*. Let $A, B, S \subset V$ be three disjoint subsets of nodes of G. A and B are said to be independent given S if $p(A,B|S) = p(A|S)p(B|S)$. In the graph representation it corresponds to cases where S separates A and B: there is no path between any node in A and any node in B that is not going through S.

Example (i): In the RBM, hidden units are independent given the inputs, and conversely:

$$p(\underline{t}|\underline{x}) = \prod_{\alpha=1}^{M} p(t_\alpha|\underline{x}), \qquad p(\underline{x}|\underline{t}) = \prod_{i=1}^{M} p(x_i|\underline{t}). \tag{2.5}$$

This property is easily spotted by noticing that the graphical model (Figure 2.1a) is *bipartite*.

Directed graphical model A directed graphical model uses a Directed Acyclic Graph (DAG), specifying directed edges E between the random variables V. It induces an ordering of the random variables and in particular the notion of parent nodes $\pi_i \subset V$ of any given vertex $i \in V$: the set of vertices j such that $j \to i \in E$. The overall joint probability distribution factorizes as

$$p(\underline{x}) = \prod_{i\in V} p(x_i|\underline{x}_{\pi_i}). \tag{2.6}$$

Example (ii): The stochastic single layer feed forward network $\underline{y} = g(\underline{\underline{W}}\underline{x}; \underline{\epsilon})$, where $g(\cdot; \underline{\epsilon})$ is a function applied component-wise including a stochastic noise $\underline{\epsilon}$ that is equivalent to a conditional distribution $p_{\text{out}}(\underline{y}|\underline{\underline{W}}\underline{x})$, and where inputs and weights are respectively drawn from distributions $p_x(\underline{x})$ and $p_W(\underline{\underline{W}})$, has a joint probability distribution

$$p(\underline{y}, \underline{x}, \underline{\underline{W}}) = p_{\text{out}}(\underline{y}|\underline{\underline{W}}\underline{x}) p_x(\underline{x}) p_W(\underline{\underline{W}}), \tag{2.7}$$

precisely following such a factorization. It can be represented with a three-node DAG as in Figure 2.1b. Here we applied the definition at the level of vector/matrix valued random variables. By further assuming that p_{out}, p_W and p_x factorize over their components, we keep a factorization compatible with a DAG representation

$$p(\underline{y}, \underline{x}, \underline{\underline{W}}) = \prod_{i=1}^{N} p_x(x_i) \prod_{\mu=1}^{M} p_{\text{out}}(y_\mu | \sum_{i=1}^{N} W_{\mu i} x_i) \prod_{\mu,i} p_W(W_{\mu i}). \tag{2.8}$$

For the purpose of reasoning it may be sometimes necessary to get to the finest level of decomposition, while sometimes the coarse grained level is sufficient.

While a statistical physicist may have never been introduced to the formal definitions of graphical models, she inevitably already has drawn a few - for instance when considering the Ising model. She also certainly found them useful to guide physical intuitions. The following second form of graphical representation is probably newer to her.

Factor graph representations

Alternatively, high-dimensional joint distributions can be represented with factor graphs, that are undirected bipartite graphs $G = (V, F, E)$ with two subsets of nodes. The variables nodes V representing the random variables as in the previous section (circles in the representation) and the factor nodes F representing the interactions (squares in the representation) associated with potentials. The edge $(i\mu)$ between a variable node i and a factor node μ exists if the variable i participates in the interaction μ. We note ∂i the set of factor nodes in which variable i is involved, they are the neighbors of i in G. Equivalently we note $\partial\mu$ the neighbors of factor μ in G, they carry the arguments $\{x_i\}_{i\in\partial\mu}$, shortened as $\underline{x}_{\partial\mu}$, of the potential ψ_μ. The overall distributions is recovered in the factorized form:

$$p(\underline{x}) = \frac{1}{\mathcal{Z}} \prod_{\mu=1}^{M} \psi_\mu(\underline{x}_{\partial\mu}). \tag{2.9}$$

Compared to an undirected graphical models, the cliques are represented by the introduction of factor nodes.

Examples: The factor-graph representation of the RBM (i) is not much more informative than the pairwise undirected graphical (see Figure 2.1a). For the feed forward neural networks (ii) we draw the factor graph of $p(\underline{y}, \underline{x}|\underline{\underline{W}})$ (see Figure 2.1b).

2.1.2 Some inference questions in neural networks for machine learning

To fix the ideas, we present what kind of inference questions arising in the context of deep learning will be of particular interest to us.

Inference in generative models

Generative models used for unsupervised learning are precisely statistical models defining high-dimensional distributions with complex dependencies. As we have seen in Section 1.1.2, the most common training objective in unsupervised learning is the maximization of the log-likelihood - the log of the probability assigned by the generative model to the training set $\{\underline{x}_{(k)}\}_{k=1}^{P}$. Computing the probability of observing a given sample $\underline{x}_{(k)}$ is an inference question. It requires to marginalize

over all the hidden representations of the problem. For instance in the RBM (i) assuming binary hidden units,

$$p(\underline{x}_{(k)}) = \frac{1}{\mathcal{Z}} \sum_{\underline{t}\in\{0,1\}^M} e^{\underline{x}_{(k)}{}^\top \underline{\underline{W}}\underline{t}} p_x(\underline{x}_{(k)}) p_t(\underline{t}). \tag{2.10}$$

While the numerator will be easy to evaluate, the computation of the partition function is troublesome. Inference in generative models will be discussed in particular in Chapter 4.

Mutual information and reconstruction in stochastic deep neural networks

Recently, it has been proposed to analyse supervised learning with deep feed forward neural networks with information theory [147, 137]. A quantity of particular interest in this setting is the mutual information between the input $\underline{x}$ and output $\underline{y}$ of a neural network for a given value of the weights. The mutual information is a functional of the joint probability distribution:

$$I(\underline{x},\underline{y}) = \mathrm{KL}(p(\underline{x},\underline{y})||p(\underline{x})p(\underline{y})) = \int \mathrm{d}\underline{x}\, \mathrm{d}\underline{y}\, p(\underline{x},\underline{y}) \log \frac{p(\underline{x},\underline{y})}{p(\underline{x})p(\underline{y})}. \tag{2.11}$$

It measures the dependence between two random variables. Expressed as a KL-divergence it is zero if and only if $p(\underline{x},\underline{y}) = p(\underline{x})p(\underline{y})$, so if the random variables are independent. Otherwise it is always positive. In fact, the mutual information between input and output even diverges to $+\infty$ for a deterministic map. We shall consider networks with stochastic activations to ensure that the mutual information $I(\underline{x},\underline{y})$ is well-defined.

A related inference question is whether from the observed output $\underline{y}$ of a feed forward neural network, the input $\underline{x}$ at its origin can be reconstructed. If the two variables are largely independent, due for instance to a high level of noise, their mutual information is low while the reconstruction is impossible. In our toy example (ii), the *posterior* distribution $p(\underline{x}|\underline{\underline{W}},\underline{y})$ is deriving from the statistical model following *Bayes rule*

$$p(\underline{x}|\underline{\underline{W}},\underline{y}) = \frac{p(\underline{y}|\underline{\underline{W}},\underline{x})p_x(\underline{x})}{p(\underline{y}|\underline{\underline{W}})} = \frac{p(\underline{y}|\underline{\underline{W}},\underline{x})p_x(\underline{x})}{\int \mathrm{d}\underline{x}\, p(\underline{y}|\underline{\underline{W}},\underline{x})p_x(\underline{x})}. \tag{2.12}$$

In this context $p_x(\underline{x})$ is usually called the *prior*, it biases the reconstruction towards values compatible with the knowledge on $\underline{x}$ provided by the statistical model before any observation was considered. The difficulty in the corresponding inference problem is the marginalization necessary to compute the denominator $p(\underline{y}|\underline{\underline{W}})$, sometimes called the *evidence*. These questions will be the main concern of Chapter 5.

Learning as statistical inference: teacher-student scenario

Assuming a statistical model for the data and the trainable parameters, the learning problem can be formulated as an inference in the *teacher-student* setting. Let us use as an illustration the case of the supervised learning of the single-layer model (ii). We draw a weight matrix $\underline{\underline{W}}_0$ - the *teacher* - from the prior distribution p_W, along with a set of P inputs $\{\underline{x}_{(k)}\}_{k=1}^P$ i.i.d from p_x, and the corresponding outputs $\underline{y}_{(k)}$ from $p_{\mathrm{out}}(.|\underline{\underline{W}}_0\underline{x}_{(k)})$. The inference question is: can we recover the value $\underline{\underline{W}}_0$ from the observations of the P pairs $\{\underline{x}_{(k)},\underline{y}_{(k)}\}$? In other words, can we learn the parameters $\underline{\underline{W}}$ of a similar single-layer model - the *student* - using the observations as a training set so as to reproduce the ground-truth rule defined by $\underline{\underline{W}}_0$?

Note that the terminology applies for a generic inference problem of reconstruction: the statistical model used to generate the data along with the realization of the unknown signal is called the *teacher*; the statistical model assumed to perform the reconstruction is called the *student*. They may be identical as in the example above, in which case they are said to be matched. But they can also be different: a bit (e.g. wrong assumptions of prior distributions) or entirely (e.g. different architecture of graphical model), in which case they are said to be mismatched.

Of course in practical applications, data do not follow a known distribution and there is no mathematical ground truth rule underlying the input-output mapping. Yet the teacher-student setting offers interesting possibilities of analysis and we shall resort to it in Chapter 5 and Chapter 6.

2.1.3 Challenges in inference

Handling dimensions Interesting inference questions in the line of the ones mentioned in the previous section have no tractable closed-form solution. Exact computations of averages and marginals requires summing over a number of terms scaling exponentially with the size of the system. Hence the computational hardness comes from the astronomically large number of configurations to be tracked to describe a high-dimensional joint probability distribution. It is therefore necessary for inference to design strategies of approximations that can be implemented as algorithms running in finite, typically polynomial, time.

Finding good estimators While some inference questions are clearly posed - as for instance the computation of the likelihood of the RBM - some can accept multiple answers - in particular reconstruction problems. Let us consider the following setting: we are interested in recovering an unknown signal $\underline{x}_0$ from the knowledge of the prior distribution p_x, downstream observations $\underline{y}$ and the *channel* $p(\underline{y}|\underline{x})$ also called the *likelihood* in this context. The distribution of $\underline{x}$ knowing $\underline{y}$, called the *posterior* distribution, is given by Bayes rule

$$p(\underline{x}|\underline{y}) = \frac{p(\underline{y}|\underline{x})p_x(\underline{x})}{p(\underline{y})}. \tag{2.13}$$

As soon as $\underline{x}$ has more than a few components this distribution is difficult to handle. Instead of keeping all the information it encodes, it is preferable to build directly an estimator $\hat{\underline{x}}$ of the unknown signal. We can either choose

- the maximum a posteriori estimator $\hat{\underline{x}}_{\mathrm{MAP}} = \arg\max_{\underline{x}} p(\underline{x}|\underline{y})$. It corresponds to the most probable value of $\underline{x}_0$ given the prior and the observations. It minimizes the probability of errors defined as $\hat{x}_i \neq x_{0,i}$.
- the minimum mean-square error estimator $\hat{\underline{x}}_{\mathrm{MMSE}} = \int \mathrm{d}\underline{x}\ \underline{x}\, p(\underline{x}|\underline{y})$. It is the mean of the posterior distribution and it minimizes the mean-square error under the posterior distribution $\mathrm{MSE}(\hat{\underline{x}}) = \int \mathrm{d}\underline{x}\, p(\underline{x}|\underline{y})\, \|\underline{x} - \hat{\underline{x}}\|^2$.
- the maximum likelihood estimator $\hat{\underline{x}}_{\mathrm{MLE}} = \arg\max_{\underline{x}} p(\underline{y}|\underline{x})$. It discards the information of the prior.

The MAP and MMSE estimators corresponds to the *Bayesian inference* setting incorporating information about a prior. We have implicitly assumed above that the prior and the channel are perfectly known, i.e. a matched teacher-student scenario. This situation is also called the *Bayes optimal* setting. It is of particular interest to assess theoretical optimal performances of reconstruction. Still, Bayesian inference can also be used when only guesses of the prior and the channel are available. For instance a sparse analytical prior can be assumed for the reconstruction of real sparse signals. In the case where no prior on the data is known, the maximum likelihood estimator can be preferred to the arbitrariness of choosing a prior. This is the strategy of most unsupervised learning algorithm to find good parameters. One or the other of these estimators can be preferable according to the situation at hand.

Performance analysis in estimation The perspective of statistical inference and the teacher-student scenario allow to formalize an analysis of the difficulty of a learning problem. Intuitively, the definition of a ground-truth rule with a fixed number of degrees of freedom sets the minimum information necessary to extract from the observations, or training data, in order to achieve perfect reconstruction. This is an *information-theoretic limit*. The assumption of an underlying statistical model further enables the measurement of performance of different learning algorithms over the class of corresponding problems from an average view point. The traditional approach of computer science in studying the difficulty of a class of problem is to examine the *worst* case. This conservative strategy yields strong guarantees of success, yet it may be overly pessimistic compared to the experience of practitioners. By defining a distribution over the possible problems (or teachers), one can compute instead averaged performances, sometimes more informative of behaviors for *typical* instances. For deep learning, this approach may prove particularly interesting as the traditional bounds appear extremely loose when compared to practical examples.

2.2 Statistical physics of disordered systems

To solve and study inference questions we will resort to concepts and methods originally developed by physicists. In a nutshell we remind some key concepts in statistical physics.

Boltzmann distribution and free energy

Statistical physics aims at explaining the behaviors of systems composed of a large number of degrees of freedom. Although the laws of classical physics governing microscopic interactions are deterministic, it is unrealistic to keep track of the evolution of each of the elementary component to account for the behavior of the whole. Remarkably, macroscopic quantities of interest to the physicist are well described by an averaging over the elementary states according to the *Boltzmann distribution*. For a system with N degrees of freedom noted $\underline{x} \in \mathcal{X}^N$ and an energy functional $E(\underline{x})$, we have

$$p(\underline{x}) = \frac{e^{-\beta E(\underline{x})}}{\mathcal{Z}_N}, \quad \mathcal{Z}_N = \sum_{\underline{x}\in\mathcal{X}^N} e^{-\beta E(\underline{x})}, \quad \beta = 1/k_B T, \tag{2.14}$$

where we defined the partition function $\mathcal{Z}$. While the adoption of the probabilistic framework is already a great simplification to the complete microscopic description, it takes us back to the difficult task of handling high-dimensional joint probability distribution discussed in Section 2.1. Let us discuss key conceptual steps further taken by physicists to understand large interacting systems.

To characterize macroscopic systems careful attention should be given to the free energy

$$F_N = -\log \mathcal{Z}_N/\beta = -\frac{1}{\beta}\log \sum_{\underline{x}\in\mathcal{X}^N} e^{-\beta E(\underline{x})} \tag{2.15}$$

$$= -\frac{1}{\beta}\left(\sum_{\underline{x}\in\mathcal{X}^N} -\beta E(\underline{x}) e^{-\beta E(\underline{x})}/\mathcal{Z}_N + \sum_{\underline{x}\in\mathcal{X}^N} \left(\beta E(\underline{x}) + \log \mathcal{Z}_N\right) e^{-\beta E(\underline{x})}/\mathcal{Z}_N \right) \tag{2.16}$$

$$= \left\langle E(\underline{x}) \right\rangle + \left\langle \log p(\underline{x}) \right\rangle /\beta = U_N - H_N/\beta \tag{2.17}$$

where $\langle\cdot\rangle$ stands for the average over the Boltzmann distribution, U_N is the average energy and H_N is the entropy. Note that the entropy of statistical physics and the entropy of information theory [133] have the same definition relatively to a distribution.

The thermodynamic limit

The number of available configurations $\mathcal{X}^N$ grows exponentially with N, yet considering the *thermodynamic* limit $N \to \infty$ can simplify computations due to concentrations. Let $e_N = E/N$ be the energy per degree of freedom, $\Omega(E) = \sum_{\underline{x}\in\mathcal{X}^N} \delta\left(E - E(\underline{x})\right) = e^{N h_N}$ be the number of configurations at energy $E = N e_N$, and $h_N(e_N)$ the entropy per degree of freedom. Then the partition function can be re-written as a sum over the configurations of a given energy e_N

$$\mathcal{Z}_N = \sum_E \sum_{\underline{x}\in\mathcal{X}^N} \delta\left(E - E(\underline{x})\right) e^{-N\beta e_N} = \sum_{e_N} e^{-N(\beta e_N - h_N(e_N))} = \sum_{e_N} e^{-N\beta f_N(e_N)}, \tag{2.18}$$

where we define $f_N(e_N)$ the free energy density of states of energy e_N. This rewriting implies that at large N the states of energy minimizing the free energy are exponentially more likely than any other states. In the thermodynamic limit, the statistics are *dominated* by these states. Using the definitions of the limits

$$e = \lim_{N\to\infty} e_N, \quad h(e) = \lim_{N\to\infty} h_N(e_N), \quad \beta f(e) = \lim_{N\to\infty} \beta e_N - h_N(e_N), \tag{2.19}$$

we have

$$f = \min_e f(e) = \lim_{N\to\infty} F_N/N, \tag{2.20}$$

$$\mathcal{Z} = \lim_{N\to\infty} Z_N = e^{-\beta f}. \tag{2.21}$$

The existence of the previous limits is the condition to the application of the thermodynamic limit.

Disordered systems

To consider the thermodynamic limit, the reader might have imagined a translationally invariant system, readily extendable. Remarkably, the statistical physics framework can be applied more generally to systems with *quenched disorder*. Here the interactions in the system are function of the realization of random variables so that the energy functional is itself function of a global random variable, noted $\underline{\underline{W}} \in \mathcal{W}$ for instance. The energy of a state of the system $\underline{x}$ is then $E(\underline{x}; \underline{\underline{W}})$.

In principle the system properties depend on a given realization of the disorder $\underline{\underline{W}}$ - quite certainly the average value $\langle x_i x_j \rangle_W$ for instance. Yet some aggregated properties are expected to be *self-averaging* in the thermodynamic limit, meaning that they concentrate on their mean with respect to the disorder W as the fluctuations are averaged out. It is the case of the free energy, which formally verifies:

$$\lim_{N\to\infty} F_{N;W}/N = \lim_{N\to\infty} \mathbb{E}_W[F_{N;W}/N] = f. \tag{2.22}$$

Thus the typical behavior of an ensemble of complicated systems can be studied in the statistical physics framework by averaging over the disorder.

Statistical physics of inference problems

Statistical inference questions are mapped to statistical physics systems by interpreting general joint probability distributions as Boltzmann distributions (2.14). Turning back to our simple inference examples:

(i) The RBM directly borrows its definition from statistical physics with

$$E(\underline{x}, \underline{t}; \underline{\underline{W}}) = -\underline{x}^\top \underline{\underline{W}} \underline{t} - \log p_x(\underline{x}) - \log p_t(\underline{t}). \tag{2.23}$$

The inverse temperature parameter can either be considered equal to 1 or as a scaling factor of the weight matrix $\underline{\underline{W}} \leftarrow \beta \underline{\underline{W}}$ and log-priors. The weights play the role of the disorder. Here the computational hardness in estimating the log-likelihood comes from the estimation of the log-partition function, which is precisely the free energy.

(ii) The posterior in the estimation of the input from the output of a network is mapped to a Boltzmann distribution by setting $E(\underline{x}; \underline{y}) = -\log p(\underline{y}|x) p_x(\underline{x})$. The disorder is here materialized by the observations.

In this Section we introduced the key conceptual preliminary steps taken by physicists to handle high-dimensional joint distributions. These are the starting point to design approximate inference methods. Before turning to an introduction to mean-field approximations, we stress the originality of the statistical physics approach. Relying on the thermodynamic limit, these methods will provide asymptotic results. Nevertheless, experience shows that the behavior of large finite-size systems are often well explained by the infinite-size limit. Furthermore, the disorder average transposed in the language of statistical inference will equate to studying the typical rather than worst case scenario, with the advantages and limitations discussed in 2.1.3. Finally, we must emphasize that previous and following derivations are not mathematically rigorous. They are based on 'correct' assumptions allowing to push further the understanding of the problems at hand, while a formal proof of the assumptions is possibly much harder to obtain.

3 Selected overview of mean-field treatments: Free energies and algorithms

(Techniques)

Mean-field methods are a set of techniques enabling to approximate marginalized quantities of joint probability distributions by exploiting knowledge on the dependencies between random variables. They are usually said to be analytical - as opposed to numerical Monte Carlo methods. In practice they usually replace a summation exponentially large in the size of the system by an analytical formula involving a set of parameters, themselves solution of a closed set of non-linear equations. Finding the values of these parameters typically requires only a polynomial number of operations.

In this Chapter, we will give a selected overview of mean-field methods as they were introduced in the statistical physics and/or signal processing literature. A key take away of the following sections is that closely related results can be obtained from different heuristics of derivation. We will start by deriving the simplest and historically first mean-field method. We will then introduce the important broad techniques that are used in the following Chapters, high-temperature expansions, message-passing algorithms and the replica method. In a closing section we will cover the most recent extensions of mean-field methods that were stepping stones for the contributions of this thesis.

3.1 Naive mean-field

3.1.1 Variational derivation

The naive mean-field method consists in approximating the joint probability distribution of interest by a fully factorized distribution. Therefore, it ignores correlations between random variables. Among multiple methods of derivation, we present here the variational method: it is the best known methods across fields and it readily shows that, for any joint probability distribution interpreted as a Boltzmann distribution, the rather crude naive mean-field approximation yields an upper bound on the free energy. For the purpose of demonstration we consider an Ising model with spins variables $\underline{x} = (x_1, \cdots, x_N) \in \mathcal{X} = \{0,1\}^N$, and energy function

$$E(\underline{x}) = -\sum_{i=1}^{N} b_i x_i - \sum_{(ij)} W_{ij} x_i x_j = -\underline{b}^\top \underline{x} - \frac{1}{2}\underline{x}^\top \underline{\underline{W}}\underline{x}\,, \quad \underline{b} \in \mathbb{R}^N\,, \quad \underline{\underline{W}} \in \mathbb{R}^{N\times N}\,, \tag{3.1}$$

where the notation (ij) stands for pairs of connected spin-variables, and the weight matrix $\underline{\underline{W}}$ is symmetric. The choices for $\{0,1\}$ rather than $\{-1,+1\}$ for the variable values, the notations $\underline{\underline{W}}$ for weights (instead of couplings), $\underline{b}$ for biases (instead of local fields), as well as the vector notation, are leaning towards the machine learning conventions and the interpretation of the model as a Boltzmann machine (without hidden units). We denote by $q_{\underline{m}}$ a fully factorized distribution on $\{0,1\}^N$, which is a multivariate Bernoulli distribution parametrized by the mean values $\underline{m} = (m_1, \cdots, m_N) \in [0,1]^N$ of the marginals (denoted by q_{m_i}):

$$q_{\underline{m}}(\underline{x}) = \prod_{i=1}^{N} q_{m_i}(x_i) = \prod_{i=1}^{N} m_i \delta(x_i - 1) + (1 - m_i)\delta(x_i). \tag{3.2}$$

We look for the optimal $q_{\underline{m}}$ distribution to approximate the Boltzmann distribution $p(\underline{x}) = e^{-\beta E(\underline{x})}/\mathcal{Z}$ by minimizing the KL-divergence

$$\min_{\underline{m}} \mathrm{KL}(q_{\underline{m}}||p) = \min_{\underline{m}} \sum_{\underline{x}\in\mathcal{X}} q_{\underline{m}}(\underline{x}) \log \frac{q_{\underline{m}}(\underline{x})}{p(\underline{x})} \tag{3.3}$$

$$= \min_{\underline{m}} \sum_{\underline{x}\in\mathcal{X}} q_{\underline{m}}(\underline{x}) \log q_{\underline{m}}(\underline{x}) + \beta \sum_{\underline{x}\in\mathcal{X}} q_{\underline{m}}(\underline{x})(E(\underline{x})) + \log \mathcal{Z} \tag{3.4}$$

$$= \min_{\underline{m}} \ \beta G(q_{\underline{m}}) - \beta F \geq 0, \tag{3.5}$$

where the last inequality comes from the positivity of the KL-divergence. For a generic distribution q, $G(q)$ is the *Gibbs free energy* for the energy $E(\underline{x})$,

$$G(q) = \sum_{\underline{x}\in\mathcal{X}} q(\underline{x})(E(\underline{x})) + \frac{1}{\beta}\sum_{\underline{x}\in\mathcal{X}} q(\underline{x})\log q(\underline{x}) = U(q) - H(q)/\beta \geq F, \tag{3.6}$$

involving the average energy $U(q)$ and the entropy $H(q)$. It is greater than the true free energy F except when $q = p$, in which case they are equal. Note that this fact also means that the Boltzmann distribution minimizes the Gibbs free energy. Restricting to factorized $q_{\underline{m}}$ distributions, we obtain the naive mean-field approximations for the mean value of the variables (or *magnetizations*) and the free energy:

$$\underline{m}^* = \arg\min_{\underline{m}} G(\underline{m}) = \langle \underline{x} \rangle_{q_{\underline{m}^*}} \tag{3.7}$$

$$F_{\mathrm{NMF}} = G(\underline{m}^*) \geq F. \tag{3.8}$$

The choice of a very simple family of distributions $q_{\underline{m}}$ limits the quality of the approximation but allows for tractable computations of other observables, for instance the two-spins correlations $\langle x_i x_j \rangle_{q^*} = m_i^* m_j^*$ or variance of one spin $\langle x_i^2 \rangle_{q^*} - \langle x_i \rangle_{q^*}^2 = m_i^* - m_i^{*2}$.

In our example of the Boltzmann machine it is easy to compute the Gibbs free energy for the factorized ansatz,

$$U_{\mathrm{NMF}}(m) = \langle E(\underline{x}) \rangle_{q_{\underline{m}}} = -\underline{b}^\top \underline{m} - \frac{1}{2}\underline{m}^\top \underline{\underline{W}}\underline{m}\,, \tag{3.9}$$

$$H_{\mathrm{NMF}}(m) = -\langle \log q(\underline{x}) \rangle_{q_{\underline{m}}} = -\sum_{i=1}^{N} m_i \log m_i + (1-m_i)\log(1-m_i)\,, \tag{3.10}$$

$$G_{\mathrm{NMF}}(m) = U_{\mathrm{NMF}}(m) - H_{\mathrm{NMF}}(m)/\beta. \tag{3.11}$$

Looking for stationary points we find a closed set of non linear equations for the m_i^*,

$$\left.\frac{\partial G}{\partial m_i}\right|_{\underline{m}^*} = 0 \quad \Rightarrow \quad m_i^* = \sigma(\beta b_i + \sum_{j\in\partial i} \beta W_{ij} m_j^*) \quad \forall i\,, \tag{3.12}$$

where $\sigma(x) = (1+e^{-x})^{-1}$. The solutions can be computed by iterating these relations until a fixed point is reached. To understand the implication of the restriction to factorized distributions, it is instructive to compare this naive mean-field equation with the exact identity

$$\langle x_i \rangle_p = \langle \sigma(\beta b_i + \sum_{j\in\partial i} \beta W_{ij} x_j) \rangle_p\,. \tag{3.13}$$

Under the Boltzmann distribution $p(\underline{x}) = e^{-\beta E(\underline{x})}/\mathcal{Z}$, these averages are difficult to compute. The naive mean-field method is neglecting the fluctuations of the effective field felt by the variable x_i: $\sum_{j\in\partial i} W_{ij} x_j$, keeping only its mean $\sum_{j\in\partial i} W_{ij} m_j$. This incidentally justifies the name of mean-field methods.

3.1.2 When does naive mean-field hold true?

The previous derivation shows that the naive mean-field approximation allows to bound the important free energy. While this bound is expected to be rough in general, the approximation is reliable when the fluctuations of the local effective fields $\sum_{j\in\partial i} W_{ij} x_j$ are small. This happens in particular at the thermodynamic limit $N \to \infty$ in *infinite range* models, that is when weights or couplings are not only local but distributed in the entire system, or if each variable interacts directly with a non-vanishing fraction of the whole set of variables. The influence of the rest of the system can then be treated as an average background. Provided the couplings are weak enough, the naive mean-field methods even becomes asymptotically exact. This is the case of the *Curie-Weiss* model, which is the fully connected version of the model (3.1) with all $W_{ij} = 1/N$. The sum of weakly dependent variables then concentrates on its mean by the central limit theorem. We

stress that it means that for finite dimensional models (more representative of a physical system, where for instance variables are assumed to be attached to the vertices of a lattice with nearest neighbors interactions), mean-field methods are expected to be quite poor. By contrast, infinite range models are thus traditionally called *mean-field models* by physicists.

In the next Section we will recover the naive mean-field equations through a different method. The following derivation will also allow to compute corrections to the rather crude approximation we just discussed by taking into account some of the correlations it neglects.

3.2 Thouless Anderson and Palmer equations

The TAP mean-field equations [144] were originally derived as an exact mean-field theory for the Sherrington-Kirkpatrick (SK) model [134]. This emblematic *spin glass* model corresponds to a fully connected Ising model with energy (3.1) and disordered couplings W_{ij} drawn independently from a Gaussian distribution with zero mean and variance W_0/N. The derivation followed from arguments specific to the SK model. Later, it was showed that the approximation could be recovered from a second order Taylor expansion at high temperature by Plefka [114] and that it could be further corrected by the systematic computation of higher orders by Georges and Yedidia [47]. We will briefly present this last derivation, having again in mind the example of the generic Boltzmann machine (3.1).

3.2.1 Outline of the derivation

Going back to the variational formulation (3.5), we now perform a minimization in two steps. We start by considering the family of distributions $q_{\underline{m}}$ enforcing $\langle \underline{x} \rangle_{q_{\underline{m}}} = \underline{m}$ for a fixed vector of magnetizations $\underline{m}$ but without any factorization constraint. We consider the corresponding Gibbs free energy

$$\tilde{G}(q_{\underline{m}}) = U(q_{\underline{m}}) - H(q_{\underline{m}})/\beta, \tag{3.14}$$

and perform a first minimization at fixed $\underline{m}$ over the $q_{\underline{m}}$ to define another auxiliary free energy

$$G(\underline{m}) = \min_{q_{\underline{m}}} \tilde{G}(q_{\underline{m}}). \tag{3.15}$$

A second minimization over $\underline{m}$ would recover the overall unconstrained minimum of the variational problem (3.5) which is the exact free energy

$$F = -\log \mathcal{Z}/\beta = \min_{\underline{m}} G(\underline{m}). \tag{3.16}$$

Yet the actual value of $G(\underline{m})$ turns out as complicated to compute as F itself. Fortunately, $\beta G(\underline{m})$ can be easily approximated by a Taylor expansion around $\beta = 0$ due to interactions vanishing at high temperature. After expanding, the minimization over $G(\underline{m})$ yields a set a self consistent equations on the magnetizations $\underline{m}$ - called the *TAP equations* - reminiscent of the naive mean-field equations (3.12), again typically solved by iterations. Plugging the solutions $\underline{m}^*$ back into the expanded expression yields the *TAP free energy* $F_{\text{TAP}} = G(\underline{m}^*)$. Note that ultimately the approximation lies in the truncation of the expansion. At first order the naive mean-field approximation is recovered. At second order, this derivation justifies the original TAP approximation for SK model [144]. In Section 4.1 of the next Chapter, we detail the different steps here outlined for a generalization of the Boltzmann machine (3.1).

3.2.2 Illustration on the Boltzmann machine and important remarks

For the Boltzmann machine (3.1), the TAP equations and TAP free energy (truncated at second order) are [144],

$$m_i^* = \sigma\left(\beta b_i + \sum_{j\in\partial i} \beta W_{ij} m_j^* - \beta^2 W_{ij}^2 (m_j^* - \frac{1}{2})(m_i^* - m_i^{*2})\right) \quad \forall i \tag{3.17}$$

$$\beta G(\underline{m}^*) = -H_{\text{NMF}}(\underline{m}^*) - \beta \sum_{i=1}^{N} b_i m_i^* - \beta \sum_{(ij)} m_i^* W_{ij} m_j - \frac{\beta^2}{2} \sum_{(ij)} W_{ij}^2 (m_i^* - m_i^{*2})(m_j - m_j^{*2}) \,, \tag{3.18}$$

where naive mean-field entropy was defined in (3.10). For this model, albeit with $\{+1, -1\}$ variables instead of $\{0, 1\}$, several references present the details of the derivation sketched in the previous paragraph, in particular [47, 105, 159].

Onsager reaction term Compared to the naive mean-field approximation the TAP equations include a correction to the effective field called the *Onsager reaction term*. The idea is that, in the effective field at variable i, we should consider corrected magnetizations of neighboring spins $j \in \partial i$, that would correspond to the absence of variable i. This intuition echoes at two other derivations of the TAP approximation: the cavity method [90] that we will not cover and the message passing which will be discussed in the next section.

As far as the SK model is concerned, this second order correction is enough in the thermodynamic limit as the statistics of the weights imply that higher orders will typically be subleading. In general, the correct TAP equations for a given model will depend on the statistics of interactions and there is no guarantee that there exists an order of truncation leading to an exact mean-field theory.

Single instance Although the selection of the correct TAP approximation relies on the statistics of the weights, the derivation of the expansion outlined above does not require to average over them, i.e. it does not require an average over the disorder. Consequently, the approximation method is well defined for a single instance of the random disordered model. The TAP free energy and magnetizations can be computed for a given (realization of the) set of weights $\{W_{ij}\}_{(ij)}$. Note that this fact further allowed the designing of the adaptive TAP approximation [107, 108, 109], that we will discuss in Section 3.5.3, requiring no prior knowledge on the statistics of the weights. The Onsager correction is here computed directly for a given $\underline{\underline{W}}$.

Finding solutions The self-consistent equations on the magnetizations (3.17) are usually solved by turning them into an iteration scheme and looking for fixed points. This generic recipe leaves nonetheless room for interpretation: which exact form should be iterated? How should the updates for the different equations be scheduled? Which time indexing should be used? While the following scheme may seem natural,

$$m_i{}^{(t+1)} \leftarrow \sigma\left(\beta b_i + \sum_{j\in\partial i} \beta W_{ij} m_j{}^{(t)} - W_{ij}^2 \left(m_j{}^{(t)} - \frac{1}{2}\right)\left(m_i{}^{(\mathbf{t})} - m_i{}^{(\mathbf{t})^2}\right)\right) \tag{3.19}$$

it typically has more convergence issues than the following alternative scheme including the time index $t-1$,

$$m_i{}^{(t+1)} \leftarrow \sigma\left(\beta b_i + \sum_{j\in\partial i} \beta W_{ij} m_j{}^{(t)} - W_{ij}^2 \left(m_j{}^{(t)} - \frac{1}{2}\right)\left(m_i{}^{(\mathbf{t-1})} - m_i{}^{(\mathbf{t-1})^2}\right)\right). \tag{3.20}$$

This issue was first discussed by [17]. Remarkably, this last scheme, or algorithm, is actually the one obtained by the approximate message passing derivation that will be discussed in the next Section.

Solutions of the TAP equations The TAP equations can admit multiple solutions with either identical or different TAP free energy. While the true free energy F corresponds to the minimum of the Gibbs free energy, reached for the Boltzmann distribution, the TAP derivation consists in performing an effectively unconstrained minimization in two steps, but with an approximation through a Taylor expansion in between. The truncation of the expansion therefore breaks the correspondence between the discovered minimizer and the unique Boltzmann distribution. For the SK model for instance, the number of solutions of the TAP equations increases rapidly as β grows [90]. The different solutions must be weighted according to their free energy density and averaged to recover the thermodynamics predicted by the replica computation [33], another mean-field approximation discussed in Section 3.4.

3.2.3 Generalizing the Georges-Yedidia expansion

In the derivation outlined above for binary variables, $x_i = 0$ or 1, the mean of each variable m_i was fixed. This is enough to parametrize the corresponding marginal distribution $q_{m_i}(x_i)$. Yet the expansion can actually be generalized to Potts variables (taking multiple discrete values) or real variables by introducing appropriate parameters for the marginals. For real variables, another level of approximation is introduced by restricting the set of marginal distributions tested to a parametrized family of distributions. By choosing a Gaussian parametrization, one recovers TAP equations equivalent to the approximate message passing algorithm that will be discussed in the next Section. In Section 4.1, we will present a derivation for real-valued Boltzmann machines with a Gaussian parametrization.

3.3 Belief propagation and approximate message passing

Another route to rediscover the TAP equations is through the approximation of message passing algorithms. Variations of the latter were discovered multiple times in different fields. In physics they were written in a restricted version as soon as 1935 by Bethe [15]. In statistics, they were developed by Pearl as methods for probabilistic inference [112]. In this section we will start by introducing a case-study of interest, the Generalized Linear Model. We will then proceed by steps to outline the derivation of the Approximate Message Passing (AMP) algorithm from the Belief Propagation (BP) equations.

3.3.1 Generalized linear model

Definition We introduce the *Generalized Linear Model* (GLM) which is a fairly simple model to illustrate message passing algorithms and which is also an elementary brick of the inference questions that will interest us in the following. It falls under the teacher-student set up, a student model is used to reconstruct a signal from a teacher model producing indirect observations. In the GLM, the product of an unknown signal $\underline{x}_0 \in \mathbb{R}^N$ and a known weight matrix $\underline{\underline{W}} \in \mathbb{R}^{N\times M}$ is observed through a noisy channel $p_{\text{out},0}$,

$$\begin{cases} \underline{\underline{W}} \sim p_W(\underline{\underline{W}}) \\ \underline{x}_0 \sim p_{x_0}(\underline{x}_0) = \prod_{i=1}^{N} p_{x_0}(x_{0,i}) \end{cases} \Rightarrow \underline{y} \sim p_{\text{out},0}(\underline{y}|\underline{\underline{W}}\underline{x}_0) = \prod_{\mu=1}^{M} p_{\text{out},0}(y_\mu|\underline{w_\mu}^\top \underline{x}_0). \tag{3.21}$$

The probabilistic graphical model corresponding to this teacher is represented on Figure 3.1. The prior over the signal p_{x_0} is supposed to be factorized, and the channel $p_{\text{out},0}$ likewise. The inference problem is to produce an estimator $\hat{\underline{x}}$ for the unknown signal $\underline{x}_0$ from the observations $\underline{y}$. Given the prior p_x and the channel p_{out} of the student not necessarily matching the teacher, the posterior distribution is

$$p(\underline{x}|\underline{y},\underline{\underline{W}}) = \frac{1}{\mathcal{Z}(\underline{y},\underline{\underline{W}})} p_{\text{out}}(\underline{y}|\underline{x},\underline{\underline{W}}) p_x(\underline{x}), \quad \mathcal{Z}(\underline{y},\underline{\underline{W}}) = \int \mathrm{d}\underline{x}\; p(\underline{y}|\underline{x},\underline{\underline{W}}) p_x(\underline{x}), \tag{3.22}$$

represented as a factor graph also on Figure 3.1. The difficulty of the task is controlled by the measurement ratio $\alpha = M/N$ and the amplitude of the noise possibly present in the channel.

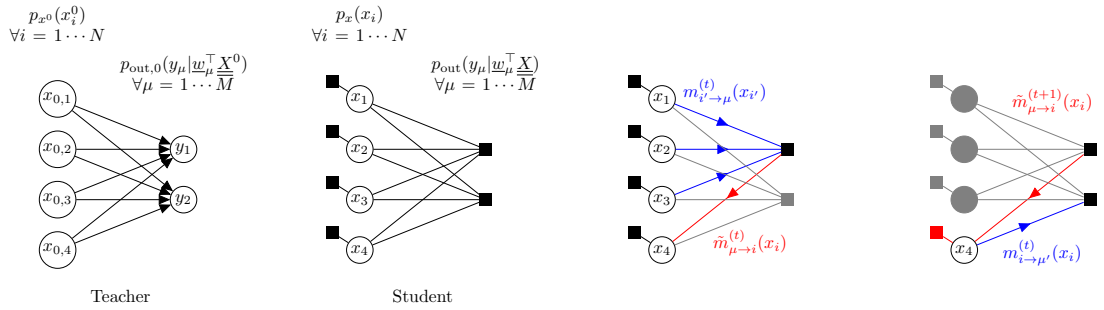


Figure 3.1: Graphical representations of the Generalized Linear Model. **Left:** Probabilistic graphical model of the teacher. **Middle left:** Factor graph representation of the posterior distribution on the signal $\underline{x}$ under the student statistical model. **Middle right and right:** Belief propagation updates (3.31) - (3.32) for approximate inference.

Applications The generic GLM underlies a number of applications. In the context of neural networks of particular interest in this thesis, the channel p_{out} generating observations $\underline{y} \in \mathbb{R}^M$ can equivalently be seen as a stochastic activation function $g(\cdot, \underline{\epsilon})$ incorporating a noise $\underline{\epsilon} \in \mathbb{R}^M$ component-wise to the output. The inference of the teacher signal in a GLM has then two possible interpretations. On the one hand, it can be interpreted as the reconstruction of the input of a stochastic single-layer neural network from its output. On the other hand, the same question can also corresponds to the Bayesian learning of a single-layer neural network with a single output - the perceptron - where this time $\{\underline{\underline{W}}, \underline{y}\}$ are interpreted as the collection of training input-output pairs and $\underline{x}_0$ plays the role of the unknown weight vector of the teacher.

However, one of the most important application of the GLM, Compressed Sensing (CS) [35], does not involve neural networks. It consist in recovering a ρ-sparse signal $\underline{x}_0$, i.e. with ρN ($< N$) non zero entries, from M noisy linear measurements gathered in $\underline{y}$. At zero noise, the information theoretic limit for the reconstruction is $M > \rho N$, or equivalently $\alpha > \rho$. Yet at this threshold the reconstruction requires the knowledge of the position of the non-zeros entries, or an exponential number of operations to test all possibilities. Hence the problem is non-trivial as soon as $M < N$. Studying the performance of algorithms in between the impossible and trivial regime is of primal practical and theoretical interest.

Statistical physics treatment and scalings From the statistical physics perspective, the effective energy functional is read from the posterior (3.22) seen as a Boltzmann distribution with energy

$$E(\underline{x}) = -\log p_{\text{out}}(\underline{y}|\underline{x}, \underline{\underline{W}}) p_x(\underline{x}) = -\sum_{\mu=1}^{M} \log p_{\text{out}}(y_\mu | \sum_{i=1}^{N} W_{\mu i} x_i) - \sum_{i=1}^{N} p_x(x_i). \tag{3.23}$$

The inverse temperature β has here no formal equivalent and can be thought as being equal to 1. The energy is therefore a function of the random realizations of $\underline{\underline{W}}$ and $\underline{y}$, playing here the role of the disorder. The weight matrix is assumed to have i.i.d. Gaussian entries with zero mean and variance $1/N$. The prior of the signal is chosen so as to ensure that the x_i (and consequently the y_μ) remain of order 1. Finally, the thermodynamic limit $N \to \infty$ is taken for a fixed measurement ratio $\alpha = M/N$.

3.3.2 Belief Propagation

The Belief Propagation algorithm is an exact inference algorithm for joint probability distributions with acyclic (or tree) underlying factor graph. It allows to compute marginals of the random variables using a set of auxiliary functions called *messages*, attached to the edges of the factor graph. Starting at a leaf, the messages communicate beliefs of a given node variable taking a given value based on the nodes and factors already visited along the tree.

General formulation For a generic joint probability distribution, BP can be written considering the underlying factor graph. Using notations from Section 2.1.1, the sum-product version of BP (as opposed to the max-sum, see e.g. [88]) consists in the update equations

$$\tilde{m}^{(t)}_{\mu\to i}(x_i) = \frac{1}{\mathcal{Z}_{\mu\to i}} \int \prod_{i'\in\partial\mu\setminus i} \mathrm{d}x_{i'}\ \psi_\mu(\underline{x}_{\partial\mu}) \prod_{i'\in\partial\mu\setminus i} m^{(t)}_{i'\to\mu}(x_{i'}), \tag{3.24}$$

$$m^{(t+1)}_{i\to\mu}(x_i) = \frac{1}{\mathcal{Z}_{i\to\mu}} p_x(x_i) \prod_{\mu'\in\partial i\setminus\mu} \tilde{m}^{(t)}_{\mu'\to i}(x_i) \tag{3.25}$$

where the i-s index the variable nodes, the μ-s index the factor nodes and the ψ_μ are the potential functions. The notation $\partial\mu\setminus i$ designate the set of neighbor variables of the factor μ except the variable i (and reciprocally for $\partial i\setminus\mu$). The partition functions $\mathcal{Z}_{i\to\mu}$ and $\mathcal{Z}_{\mu\to i}$ are normalization factors ensuring that the messages can be interpreted as probabilities. At convergence of the iterations the posterior marginals can be computed as

$$m_i(x_i) = \frac{1}{\mathcal{Z}_i} p_x(x_i) \prod_{\mu\in\partial i} \tilde{m}_{\mu\to i}(x_i) \tag{3.26}$$

and the Bethe approximation of the free energy is given by

$$F_{\text{Bethe}} = -\sum_{i\in V} \log \mathcal{Z}_i - \sum_{\mu\in F} \log \mathcal{Z}_\mu + \sum_{(i\mu)\in E} \log \mathcal{Z}_{\mu i}\,, \tag{3.27}$$

with

$$\mathcal{Z}_i = \int \mathrm{d}x_i\ p_x(x_i) \prod_{\mu\partial i} \tilde{m}_{\mu\to i}(x_i)\,, \tag{3.28}$$

$$\mathcal{Z}_\mu = \int \prod_{i\in\partial\mu} \mathrm{d}x_i\ \psi(\underline{x}_{\partial\mu}) \prod_{i\in\partial\mu} m_{i\to\mu}(x_i)\,, \tag{3.29}$$

$$\mathcal{Z}_{\mu i} = \int \mathrm{d}x_i\ \tilde{m}_{\mu\to i}(x_i) m_{i\to\mu}(x_i)\,. \tag{3.30}$$

These marginals, as well as the Bethe free energy, will only be exact if the underlying factor graph is a tree. Nonetheless, the algorithm (3.24)-(3.25), occasionally then called *loopy-BP*, can sometimes be converged on graphs with cycles and in some cases will provide high quality approximations. For instance, graphs with no short loops are locally tree like and well treated by BP provided correlations decay with distance. BP will also appear principled for some infinite range mean-field models previously discussed; an example of which being our case-study the GLM. Note that for graphs with cycles the Bethe free energy is a priori not a bound of the exact free energy.

Belief propagation for the GLM The writing of the BP-equations for our case-study is schematized on the right of Figure 3.1. There are $N \times M$ updates:

$$\tilde{m}^{(t)}_{\mu\to i}(x_i) = \frac{1}{\mathcal{Z}_{\mu\to i}} \int \prod_{i'\neq i} \mathrm{d}x_{i'}\ p_{\text{out}}(y_\mu|w_\mu{}^\top \underline{x}) \prod_{i'\neq i} m^{(t)}_{i'\to\mu}(x_{i'}), \tag{3.31}$$

$$m^{(t+1)}_{i\to\mu}(x_i) = \frac{1}{\mathcal{Z}_{i\to\mu}} p_x(x_i) \prod_{\mu'\neq\mu} \tilde{m}^{(t)}_{\mu'\to i}(x_i). \tag{3.32}$$

Despite a relatively concise formulation, running BP in practice turns out intractable since for a signal $\underline{x}$ taking continuous values it would entail keeping track of distributions on continuous variables. In this case, BP is approximated by the (G)AMP algorithm presented in the next section.

3.3.3 (Generalized) approximate message passing

The name of approximate message passing (AMP) was fixed by Donoho, Maleki and Montanari [34] who derived the algorithm in the context of Compressed Sensing. Several works from statistical

physics had nevertheless already proposed related algorithmic procedures and made connections with the TAP equations for different systems. The algorithm was derived systematically for any channel of the GLM by Rangan [115] and became Generalized-AMP (GAMP).

The systematic procedure to write AMP for a given joint probability distribution consists in first writing BP on the factor graph, second project the messages on a parametrized family of function to obtain the corresponding *relaxed-BP* and third close the equations on a reduced set of parameters by keeping only leading terms in the thermodynamic limit. We will quickly review these steps for the GLM.

Relaxed Belief Propagation

In the thermodynamic limit $M, N \to +\infty$, on can show that the scaling $1/\sqrt{N}$ of the W_{ij} and the extensive connectivity of the underlying factor graph imply that messages are approximately Gaussian. The algorithm then only needs to keep track of the two first moments. In the derivation of r-BP different means and variances are in fact introduced to parametrize messages as follows

$$\tilde{m}^{(t)}_{\mu\to i}(x_i) \propto e^{B^{(t)}_{\mu\to i}x_i + \frac{1}{2}A^{(t)}_{\mu\to i}x_i^2} \propto \int \mathrm{d}z_\mu \; p_{\text{out}}(y_\mu|z_\mu) e^{-\frac{(z_\mu - W_{\mu i}x_i - \omega^{(t)}_{\mu\to i})^2}{2V^{(t)}_{\mu\to i}}}, \tag{3.33}$$

$$m^{(t+1)}_{i\to\mu}(x_i) \propto e^{-\frac{(\hat{x}^{(t+1)}_{i\to\mu} - x_i)^2}{2C^{x\,(t+1)}_{i\to\mu}}} \propto p_x(x_i) e^{-\frac{(\lambda^{(t)}_{i\to\mu} - x_i)^2}{2\sigma^{(t)}_{i\to\mu}}}, \tag{3.34}$$

where we introduced the dummy variable $\underline{z} = \underline{\underline{W}}\,\underline{x}$, an important intermediate step in the reconstruction of $\underline{x}$ from $\underline{y}$, and the notation $\propto$ that omits the normalization factor for distributions. The BP algorithm projected on this parametrization makes up the r-BP inference algorithm with $O(M \times N)$ quantities to track. An approximation of the marginals is recovered from the projection on the parametrization of (3.26). A detailed derivation and developed algorithm of r-BP for the GLM can be found for example in [160]. Also, in Chapter 6 Section 6.1.1, we review the derivation in a slightly more general setting where the variables x_i and y_μ are possibly vectors instead of scalars.

Nonetheless, r-BP is scarcely used as such as the computational cost can be readily reduced with little more approximation. In the thermodynamic limit, the messages are closely related to the marginals as the contribution of the missing message between (3.25) and (3.26) is to a certain extent negligible. Careful book keeping of the order of contributions leads to a set of closed equations on parameters of the marginals, i.e. $O(N)$ variables, corresponding to the (G)AMP algorithm.

Approximate message passing

The GAMP algorithm with respect to marginals parameters, analogous to the messages parameters introduced in (3.33)-(3.34), is given in Algorithm (1). While steps 1) and 3) are general for any GLM with a random Gaussian weight matrix, steps 2) and 4) involve *update functions*. These special functions are defined respectively to choices of a prior on the signal $\underline{x}$ and an output channel producing the observations $\underline{y}$.

Input update functions relative to the prior

$$\mathcal{Z}^x = \int \mathrm{d}x \; p_x(x) e^{-\frac{(x-\lambda)^2}{2\sigma}} \tag{3.35}$$

$$f_1^x(\lambda,\sigma) = \frac{1}{\mathcal{Z}^x}\int \mathrm{d}x \; x\, p_x(x) e^{-\frac{(x-\lambda)^2}{2\sigma}} \tag{3.36}$$

$$f_2^x(\lambda,\sigma) = \frac{1}{\mathcal{Z}^x}\int \mathrm{d}x \; x^2\, p_x(x) e^{-\frac{(x-\lambda)^2}{2\sigma}} - f_1^x(\lambda,\sigma)^2 \tag{3.37}$$

Output update functions relative to the channel

$$\mathcal{Z}_{\text{out}}(y,\omega,V) = \int \mathrm{d}z\; p_{\text{out}}(y|z)\mathcal{N}(z;\omega,V) \tag{3.38}$$

$$g_{\text{out}}(y,\omega,V) = \frac{1}{\mathcal{Z}_{\text{out}}}\int \mathrm{d}z\; \frac{(z-\omega)}{V} p_{\text{out}}(y|z)\mathcal{N}(z;\omega,V) \tag{3.39}$$

$$\partial_\omega g_{\text{out}}(y,\omega,V) = \frac{1}{\mathcal{Z}_{\text{out}}}\int \mathrm{d}z\; \frac{(z-\omega)^2}{V^2} p_{\text{out}}(y|z)\mathcal{N}(z;\omega,V) - \frac{1}{V} - g_{\text{out}}(y,\omega,V)^2 \tag{3.40}$$

Algorithm 1 Approximate Message Passing

Input: matrix $\underline{y} \in \mathbb{R}^M$ and matrix $\underline{\underline{W}} \in \mathbb{R}^{M\times N}$:
Initialize: $\hat{x}_i$, $C_i^x \quad \forall i$ and $g_{\text{out}\,\mu}$, $\partial_\omega g_{\text{out}\,\mu} \quad \forall \mu$
repeat
1) Estimate mean and variance of $\underline{z}$ given current $\underline{\hat{x}}$

$$V_\mu^{(t)} = \sum_{i=1}^N W_{\mu i}^2 C_i^{x\,(t)} \tag{3.41}$$

$$\omega_\mu^{(t)} = \sum_{i=1}^N W_{\mu i}\hat{x}_i^{(t)} - \sum_{i=1}^N W_{\mu i}^2 C^{x\,(t)}_i\, g_{\text{out}\,\mu}^{(t-1)} \tag{3.42}$$

2) Estimate mean and variance of the gap between optimal $\underline{z}$ and $\underline{\omega}$ given $\underline{y}$

$$\partial_\omega g_{\text{out}\,\mu}^{(t)} = \partial_\omega g_{\text{out}}(y_\mu, \omega_\mu^{(t)}, V_\mu^{(t)}) \tag{3.43}$$

$$g_{\text{out}\,\mu}^{(t)} = g_{\text{out}}(y_\mu, \omega_\mu^{(t)}, V_\mu^{(t)}) \tag{3.44}$$

3) Estimate mean and variance of $\underline{x}$ given current optimal $\underline{z}$

$$\sigma_i^{(t)} = \left(-\sum_{\mu=1}^M W_{\mu i}^2 \partial_\omega g_{\text{out}\,\mu}^{(t)}\right)^{-1} \tag{3.45}$$

$$\lambda_i^{(t)} = \hat{x}_i^{(t)} + \sigma_i^{(t)}\left(\sum_{\mu=1}^M W_{\mu i} g_{\text{out}\,\mu}^{(t)}\right) \tag{3.46}$$

4) Estimate of mean and variance of $\underline{x}$ augmented of the information about the prior

$$C_i^{x\,(t+1)} = f_2^x(\lambda_i^{(t)}, \sigma_i^{(t)}) \tag{3.47}$$

$$\hat{x}_i^{(t+1)} = f_1^x(\lambda_i^{(t)}, \sigma_i^{(t)}) \tag{3.48}$$

until convergence

Relation to TAP equations Historically the main difference between the AMP algorithm and the TAP equations is that the latter was first derived for binary variables with 2-body interactions while the former was proposed for continuous random variables with N-body interactions. The details of the derivation, not described here, rely on the knowledge of the statistics of the disordered variable $\underline{\underline{W}}$ but do not require a disorder average, as in the Georges-Yedidia expansion yielding the TAP equation. By focusing on the GLM with a random Gaussian weight matrix scaling as $O(1/\sqrt{N})$ (similar to the couplings of the SK model) we naturally obtained TAP equations at second order, with an Onsager term in the update (3.42) of ω_μ. The TAP free energy can also be recovered by plugging-in the approximate parametrized messages in the definition of the Bethe free energy (3.27). Yet an advantage of the AMP derivation from BP is that it explicitly provides 'correct' time indices in the iteration scheme to solve the self consistent equations.

Reconstruction with AMP The AMP algorithm is therefore a practical reconstruction algorithm which can be run on a single instance and estimate an unknown signal $\underline{x}_0$. Note that the

prior p_x and channel p_{out} used in the algorithm correspond to the student statistical model and they may be different from the true underlying teacher model that generates $\underline{x}_0$ and $\underline{y}$. In other words, the AMP algorithm may be used either in the Bayes Optimal or in the mismatched setting defined in Section 2.1.3. Remarkably, it is also possible to now consider a disorder average in the thermodynamic limit to study the average case computational hardness of the GLM inference problem in either of these configurations.

State Evolution

The statistical analysis of the AMP equations in the average case leads to another closed set equations in the thermodynamic limit $N \to \infty$ that was called State Evolution (SE) in [34]. Its derivation starts from the r-BP equations and relies on the assumption of independent incoming messages to invoke the Central Limit Theorem. It is therefore only necessary to follow the evolution of a set of means and variances. When the different variables and factors are statistically equivalent, as it is the case of the GLM, the State Evolution reduces to a few scalar equations. The interested reader should refer to Section 6.1.2 for a detailed derivation in a more general setting.

Mismatched setting In the general mismatched setting we need to carefully differentiate the teacher and the student. We note p_{x_0} the prior used by the teacher. We also rewrite its channel $p_{\text{out},0}$ as the explicit function $g_0(\cdot, \epsilon)$ assuming the noise ϵ to be distributed according to the standard normal distribution. The tracked quantities are the *overlaps*,

$$q = \lim_{N\to\infty} \frac{1}{N}\sum_{i=1}^{N} \hat{x}_i^2\,, \quad m = \lim_{N\to\infty} \frac{1}{N}\sum_{i=1}^{N} \hat{x}_i x_{0,i}\,, \quad q_0 = \lim_{N\to\infty} \frac{1}{N}\sum_{i=1}^{N} x_{0,i}^2 = \mathbb{E}_{p_{x_0}}[x_0^2], \tag{3.49}$$

along with auxiliary V, $\hat{q}$, $\hat{m}$ and $\hat{\chi}$:

$$\hat{q}^{(t)} = \int \mathrm{d}\epsilon\; p_{\epsilon_0}(\epsilon) \int \mathrm{d}\omega\; \mathrm{d}z\; \mathcal{N}(z,\omega;0,\underline{\underline{Q}}^{(t)}) g_{\text{out}}(\omega, g_0\left(z;\epsilon\right), V^{(t)})^2 \tag{3.50}$$

$$\hat{m}^{(t)} = \int \mathrm{d}\epsilon\; p_{\epsilon_0}(\epsilon) \int \mathrm{d}\omega\; \mathrm{d}z\; \mathcal{N}(z,\omega;0,\underline{\underline{Q}}^{(t)}) \partial_z g_{\text{out}}(\omega, g_0\left(z;\epsilon\right), V^{(t)}) \tag{3.51}$$

$$\hat{\chi}^{(t)} = -\int \mathrm{d}\epsilon\; p_{\epsilon_0}(\epsilon) \int\; dd\omega \mathrm{d}z\; \mathcal{N}(z,\omega;0,\underline{\underline{Q}}^{(t)}) \partial_\omega g_{\text{out}}(\omega, g_0\left(z;\epsilon_\mu\right), V^{(t)}) \tag{3.52}$$

$$q^{(t+1)} = \int \mathrm{d}x_0\; p_{x_0}(x_0) \int \mathcal{D}\xi\; f_1^x\left((\alpha\hat{\chi}^{(t)})^{-1}\left(\sqrt{\alpha\hat{q}^{(t)}}\xi + \alpha\hat{m}^{(t)}\underline{x}_0\right); (\alpha\hat{\chi}^{(t)})^{-1}\right)^2 \tag{3.53}$$

$$m^{(t+1)} = \int \mathrm{d}x_0\; p_{x_0}(x_0) \int \mathcal{D}\xi\; x_0 f_1^x\left((\alpha\hat{\chi}^{(t)})^{-1}\left(\sqrt{\alpha\hat{q}^{(t)}}\xi + \alpha\hat{m}^{(t)}\underline{x}_0\right); (\alpha\hat{\chi}^{(t)})^{-1}\right) \tag{3.54}$$

$$V^{(t+1)} = \int \mathrm{d}x_0\; p_{x_0}(x_0) \int \mathcal{D}\xi\; f_2^x\left((\alpha\hat{\chi}^{(t)})^{-1}\left(\sqrt{\alpha\hat{q}^{(t)}}\xi + \alpha\hat{m}^{(t)}x_0\right); (\alpha\hat{\chi}^{(t)})^{-1}\right) \tag{3.55}$$

where we use the notation $\mathcal{N}(\cdot;\cdot,\cdot)$ for the normal distribution, $\mathcal{D}\xi$ for the standard normal measure and the covariance matrix $\underline{\underline{Q}}^{(t)}$ is given at each time step by

$$\underline{\underline{Q}}^{(t)} = \begin{bmatrix} q_0 & m^{(t)} \\ m^{(t)} & q^{(t)} \end{bmatrix}.$$

Due to the self-averaging property, the performance of the reconstruction by the AMP algorithm on an instance of size N can be tracked along the iterations given

$$\text{MSE}(\hat{x}) = \frac{1}{N}\sum_{i=1}^{N}(\hat{x}_i - x_{0,i})^2 = q - 2m + q_0, \tag{3.56}$$

with only minor differences coming from finite-size effects. State Evolution also provides an efficient procedure to study from the theoretical perspective the AMP fixed points for a generic model, such as the GLM, as a function of some control parameters. It reports the average results for running the complete AMP algorithm on $O(N)$ variables using a few scalar equations. Furthermore, the State Evolution equations simplify further in the Bayes optimal setting.

Bayes optimal setting When the prior and channel are identical for the student and the teacher, the true unknown signal $\underline{x}_0$ is in some sense statistically equivalent to the estimate $\hat{\underline{x}}$ coming from the posterior. More precisely one can prove the Nishimori identities [104, 63, 101] (or [67] for a concise demonstration and discussion) implying that $q = m$, $V = q_0 - m$ and $\hat{q} = \hat{m} = \hat{\chi}$. Only two equations are then necessary to track the performance of the reconstruction:

$$\hat{q}^{(t)} = \int \mathrm{d}\epsilon \; p_{\epsilon^0}(\epsilon) \int \mathrm{d}\omega \; \mathrm{d}z \; \mathcal{N}(z, \omega; 0, \underline{\underline{Q}}^{(t)}) g_{\mathrm{out}}(\omega, g^0(z; \epsilon), V^{(t)})^2 \tag{3.57}$$

$$q^{(t+1)} = \int \mathrm{d}x_0 \; p_{x_0}(x_0) \int \mathcal{D}\xi \; f_1^x \left((\alpha \hat{\chi}^{(t)})^{-1} \left(\sqrt{\alpha \hat{q}^{(t)}} \xi + \alpha \hat{m}^{(t)} \underline{x}_0 \right); (\alpha \hat{\chi}^{(t)})^{-1} \right)^2 . \tag{3.58}$$

3.4 Replica method

Another powerful technique from the statistical physics of disordered systems to examine models with infinite range interactions is the replica method. It enables an analytical computation of the quenched free energy via non-rigorous mathematical manipulations. More developed introductions to the method can be found in [101, 22].

3.4.1 Steps of a replica computation

The basic idea of the replica computation is to apply the average over the disorder to the identity $\log \mathcal{Z} = \lim_{n \to 0} (\mathcal{Z}^n - 1)/n$, while at first considering $n \in \mathbb{N}$, before taking the $n \to 0$ limit - thus taking advantage of the fact that the average of a power of $\mathcal{Z}$ is easier to compute than the average of a logarithm. We illustrate the key steps for the calculation of the partition function of the GLM (3.22).

Disorder average for the replicated system: coupling of the replicas The average of $\mathcal{Z}^n$ for $n \in \mathbb{N}$ can be seen as the partition function of a system with $n+1$ non interacting replicas of $\underline{x}$ indexed by $a \in \{0, \cdots, n\}$, where the first replica $a = 0$ is representative of the teacher and the n other replicas are identically distributed as the student:

$$\mathbb{E}_{\underline{\underline{W}}, \underline{y}, \underline{x}_0} [\mathcal{Z}^n] = \mathbb{E}_{\underline{\underline{W}}} \left[\int \mathrm{d}\underline{y} \; \mathrm{d}\underline{x}_0 \; p_{\mathrm{out},0}(\underline{y} | \underline{\underline{W}} \underline{x}_0) p_{\underline{x}_0}(\underline{x}_0) \left(\int \mathrm{d}\underline{x} \; p_{\mathrm{out}}(\underline{y} | \underline{\underline{W}} \underline{x}) p_x(\underline{x}) \right)^n \right] \tag{3.59}$$

$$= \mathbb{E}_{\underline{\underline{W}}} \left[\int \mathrm{d}\underline{y} \prod_{a=0}^{n} \left(\mathrm{d}\underline{x}_a \; p_{\mathrm{out},a}(\underline{y} | \underline{\underline{W}} \underline{x}_a) p_{x_a}(\underline{x}_a) \right) \right] \tag{3.60}$$

$$= \mathbb{E}_{\underline{\underline{W}}} \left[\int \mathrm{d}\underline{y} \prod_{a=0}^{n} \left(\mathrm{d}\underline{x}_a \; \mathrm{d}\underline{z}_a \; \delta(\underline{z}_a - \underline{\underline{W}} \underline{x}_a) p_{\mathrm{out},a}(\underline{y} | \underline{\underline{W}} \underline{x}_a) p_{x_a}(\underline{x}_a) \right) \right] . \tag{3.61}$$

To perform the average over the disordered interactions $\underline{\underline{W}}$ we consider the statistics of $\underline{z}_a = \underline{\underline{W}} \underline{x}_a$. Recall that $W_{\mu i} \sim \mathcal{N}(W_{\mu i}; 0, 1/N)$, independently for all μ and i. Consequently, the $\underline{z}_a$ are jointly Gaussian in the thermodynamic limit with means and covariances

$$\mathbb{E}_{\underline{\underline{W}}}[z_{a,\mu}] = \mathbb{E}_{\underline{\underline{W}}} \left[\sum_{i=1}^{N} W_{\mu i} x_{a,i} \right] = 0 , \quad E_{\underline{\underline{W}}} \left[z_{a,\mu} z_{b,\nu} \right] = \sum_{i=1}^{N} x_{a,i} x_{b,i} / N = q_{ab}. \tag{3.62}$$

The overlaps, that we already introduced in the SE formalism, naturally re-appear. We introduce the notation $\underline{\underline{q}}$ for the $(n+1) \times (n+1)$ overlap matrix. Note that integrating out the disorder $\underline{\underline{W}}$ shared by the $n+1$ replicas will therefore leave us with an effective system of now coupled replicas.

Change of variable for the overlaps: decoupling of the variables Considering the overlaps as variables, and considering the Fourier representation of the Dirac distribution fixing the consistency between overlaps and replicas, we multiply the replicated average by

$$1 = \int \prod_{a,b} \mathrm{d}N q_{ab}\, \delta(N q_{ab} - \sum_{i=1}^{N} x_{a,i} x_{b,i}) = \int \prod_{a,b} \mathrm{d}N q_{ab} \int \prod_{a,b} \mathrm{d}N \hat{q}_{ab}\, e^{\hat{q}_{ab}(N q_{ab} - \sum_{i=1}^{N} x_{a,i} x_{b,i})}. \tag{3.63}$$

As a result

$$\begin{aligned} \mathbb{E}_{\underline{\underline{W}},\underline{y},\underline{x}_0}\left[\mathcal{Z}^n\right] = \int \prod_{a,b} \mathrm{d}N q_{ab} \int \prod_{a,b} \mathrm{d}N \hat{q}_{ab}\, \exp\left(N \hat{q}_{ab} q_{ab}\right) \\ \int \mathrm{d}\underline{y} \prod_{a=0}^{n} \mathrm{d}\underline{z}_a\, p_{\mathrm{out},a}(\underline{y}|\underline{z}_a) \exp\left(-\frac{1}{2}\sum_{\mu=1}^{M}\sum_{a,b} z_{a,\mu} z_{b,\mu} (\underline{\underline{q}}^{-1})_{ab} - M C(\underline{\underline{q}}, n)\right) \\ \int \prod_{a=1}^{n} \mathrm{d}\underline{x}_a\, p_{x_a}(\underline{x}_a) \exp\left(-\hat{q}_{ab} \sum_{i=1}^{N} x_{a,i} x_{b,i}\right) \end{aligned} \tag{3.64}$$

where $C(\underline{\underline{q}}, n)$ is related to the normalization of the Gaussian distributions over the $\underline{z}_a$ variables, and the integrals can be factorized over the i-s and μ-s. Thus we obtain

$$\mathbb{E}_{\underline{\underline{W}},\underline{y},\underline{x}_0}\left[\mathcal{Z}^n\right] = \int \prod_{a,b} \mathrm{d}N q_{ab} \int \prod_{a,b} \mathrm{d}N \hat{q}_{ab}\, e^{N \hat{q}_{ab} q_{ab}} e^{M \log \hat{\mathcal{I}}_z(\underline{\underline{q}})} e^{N \log \hat{\mathcal{I}}_x(\underline{\underline{\hat{q}}})}, \tag{3.65}$$

$$\hat{\mathcal{I}}_z(\underline{\underline{q}}) = \int \mathrm{d}y \prod_{a=0}^{n} \mathrm{d}z_a\, p_{\mathrm{out},a}(y|z_a) \exp\left(-\frac{1}{2}\sum_{a,b} z_a z_b (q_{ab})^{-1} - C(\underline{\underline{q}}, n)\right), \tag{3.66}$$

$$\hat{\mathcal{I}}_x(\underline{\underline{\hat{q}}}) = \int \prod_{a=1}^{n} \mathrm{d}x_a\, p_{x_a}(x_a) \exp\left(-\hat{q}_{ab} x_a x_b\right), \tag{3.67}$$

where we introduce the notation $\underline{\underline{\hat{q}}}$ for the auxiliary overlap matrix with entries $(\underline{\underline{\hat{q}}})_{ab} = \hat{q}_{ab}$. The decoupling of the x_i and the z_μ of the infinite range system yields pre-factors N and M in the exponential arguments and consequently an integral for the replicated average that is easily computed in the thermodynamic limit by the saddle point method:

$$\mathbb{E}_{\underline{\underline{W}},\underline{y},\underline{x}_0}\left[\mathcal{Z}^n\right] \simeq e^{N \underset{\underline{\underline{q}}\,\underline{\underline{\hat{q}}}}{\mathrm{extr}} \left[\phi(\underline{\underline{q}},\underline{\underline{\hat{q}}})\right]}, \quad \phi(\underline{\underline{q}},\underline{\underline{\hat{q}}}) = \sum_{a,b} \hat{q}_{ab} q_{ab} + \alpha \hat{\mathcal{I}}_z(\underline{\underline{q}}) + \hat{\mathcal{I}}_x(\underline{\underline{\hat{q}}}), \tag{3.68}$$

where we defined the replica potential ϕ.

Exchange of limits: back to the quenched average The thermodynamic average of the log-partition is recovered through an a priori risky mathematical manipulation: (i) perform an analytical continuation from $n \in \mathbb{N}$ to $n \to 0$

$$\frac{1}{N}\mathbb{E}_{\underline{\underline{W}},\underline{y},\underline{x}_0}\left[\log \mathcal{Z}\right] = \lim_{n\to 0} \frac{1}{nN} \mathbb{E}_{\underline{\underline{W}},\underline{y},\underline{x}_0}\left[\mathcal{Z}^n - 1\right] = \lim_{n\to 0} \frac{1}{nN} \log \mathbb{E}_{\underline{\underline{W}},\underline{y},\underline{x}_0}\left[\mathcal{Z}^n\right] \tag{3.69}$$

and (ii) exchange limits

$$-f = \lim_{n\to 0} \frac{1}{n} \lim_{N\to\infty} \frac{1}{N} \log \mathbb{E}_{\underline{\underline{W}},\underline{y},\underline{x}_0}\left[\mathcal{Z}^n\right] = \lim_{n\to 0} \frac{1}{n} \underset{\underline{\underline{q}}\,\underline{\underline{\hat{q}}}}{\mathrm{extr}} \left[\phi(\underline{\underline{q}},\underline{\underline{\hat{q}}})\right]. \tag{3.70}$$

Despite the apparent lack of rigour in taking these last steps, the replica method has been proven to yield exact predictions in the thermodynamic limit for different problems and in particular for the GLM [119, 9].

Saddle point solution: choice of a replica ansatz At this point, we are still left with the problem of computing the extrema of $\phi(\underline{\underline{q}}, \underline{\underline{\hat{q}}})$. To solve this optimization problem over $\underline{\underline{q}}$ and $\underline{\underline{\hat{q}}}$, a natural assumption is that replicas, that are a pure artefact of the calculation, are equivalent. This is reflected in a special structure for overlap matrices between replicas that only depend on three parameters each,

$$\underline{\underline{q}} = \begin{bmatrix} q_0 & m & m & m \\ m & q & q_{12} & q_{12} \\ m & q_{12} & q & q_{12} \\ m & q_{12} & q_{12} & q \end{bmatrix}, \quad \underline{\underline{\hat{q}}} = \begin{bmatrix} \hat{q}_0 & \hat{m} & \hat{m} & \hat{m} \\ \hat{m} & \hat{q} & \hat{q}_{12} & \hat{q}_{12} \\ \hat{m} & \hat{q}_{12} & \hat{q} & \hat{q}_{12} \\ \hat{m} & \hat{q}_{12} & \hat{q}_{12} & \hat{q} \end{bmatrix}, \tag{3.71}$$

here given as an example for $n = 3$ replicas. Plugging this *replica symmetric* (RS) ansatz in the expression of $\phi(\underline{\underline{q}}, \underline{\underline{\hat{q}}})$, taking the limit $n \to 0$ and looking for the stationary points as a function of the parameters q, m, q_{12} and $\hat{m}$, $\hat{q}$, $\hat{q}_{12}$ recovers a set of equations equivalent to SE (1.7), albeit without time indices. Hence the two a priori different heuristics of BP and the replica method are remarkably consistent under the RS assumption.

Nevertheless, the replica symmetry can be spontaneously broken in the large N limit and the dominating saddle point do not necessarily correspond to the RS overlap matrix. This replica symmetry breaking (RSB) corresponds to substantial changes in the structure of the examined Boltzmann distribution. It is among the great strength of the replica formalism to naturally capture it. We will not investigate here this direction further. By considering the Bayes optimal setting, we will avoid the necessity of considering the breaking of the replica symmetry [101, 22, 160] in the problems discussed in Chapters 5 and 6.

Bayes optimal setting As in SE the equations simplify in the matched setting, where the first replica corresponding to the teacher becomes equivalent to all the others. The replica free energy of the GLM is then given as the extremum of a potential over two scalar variables:

$$-f = \mathrm{extr}_{q\hat{q}} \left[-\frac{1}{2} q\hat{q} + \mathcal{I}_x(\hat{q}) + \alpha \mathcal{I}_z(q_0, q) \right] \tag{3.72}$$

$$\mathcal{I}_x(\hat{q}) = \int \mathcal{D}\xi \, \mathrm{d}x \, p_x(x) e^{-\hat{q}\frac{x^2}{2} + \sqrt{\hat{q}}\xi x} \log \left(\int \mathrm{d}x' \, p_x(x') e^{-\hat{q}\frac{x'^2}{2} + \sqrt{\hat{q}}\xi x'} \right) \tag{3.73}$$

$$\mathcal{I}_z(q, q_0) = \int \mathcal{D}\xi \, \mathrm{d}y \, \mathrm{d}z \, p_{\mathrm{out}}(y|z) \mathcal{N}(z; \sqrt{q}\xi, q_0 - q) \log \left(\int \mathrm{d}z' \, p_{\mathrm{out}}(y|z') \mathcal{N}(z'; \sqrt{q}\xi, q_0 - q) \right). \tag{3.74}$$

The saddle point equations corresponding to the extremization (3.72), fixing the values of q and $\hat{q}$, would again be found equivalent to the Bayes optimal SE (3.57) - (3.58).

3.4.2 Assumptions and relation to other mean-field methods

A crucial point in the above derivation of the replica formula is the extensivity of the interactions of the infinite range model that allowed the factorization of the N scaling of the argument of the exponential integrand. The statistics of the disorder $\underline{\underline{W}}$ and in particular the independence of all the $W_{\mu i}$ was also necessary. This is an important assumption for the technique to go through, although it can be possible to relax it for some types of correlation statistics, as we will see in Section 3.5.3.

Note that the replica method directly enforces the disorder averaging and does not provide a prediction at the level of the single instance. Therefore it cannot be turned into a practical algorithm of reconstruction. Nonetheless, we have seen that the saddle point equations of the replica derivation, under the RS assumption, matches the SE equations derived from BP. This is sufficient to theoretically study inference questions under a teacher-student scenario in the Bayes optimal setting.

In the mismatched setting however, the predictions of the replica method under the RS assumption and the equivalent BP conclusions can be wrong. By introducing the symmetry breaking between replicas, the method can sometimes be corrected. It is an important endeavor of the

replica formalism to grasp the importance of the overlaps and connect the form of the replica ansatz to the properties of the joint probability distribution examined. When BP fails on loopy-graph, correlations between variables are not decaying with distance, which manifests into an RSB phase. Subsequently, a message passing formalism taking into account (one step of) replica symmetry breaking was proposed [89, 88]; it maintains the consistency of the replica method with the corresponding ansatz.

3.5 Extensions of interest for this thesis

In the previous Section we saw how frameworks and procedures of mean-field approximations rely on structural and statistical properties of the model under scrutiny. While we focused on the simple, and original, examples of the SK-Boltzmann machine and the GLM with Gaussian i.i.d weight matrices as examples, the span of applicability of mean-field methods now goes far beyond. We present here some of the recent applications of these concepts in more complex cases that are of particular interest for the rest of this thesis.

3.5.1 Streaming AMP for online learning

In learning applications, it is sometimes advantageous for speed or generalization concerns to only treat a subset of examples at the time - making for instance the SGD algorithm the most popular training algorithm in deep learning. Sometimes also, the size of the current data sets may exceed the available memory. Methods implementing a step-by-step learning as the data arrives are referred as *online*, *streaming* or *mini-batch* learning, as opposed to *offline* or *batch* learning.

In [85], a mini-batch version of the AMP algorithm is proposed. On the example of the GLM, one imagines receiving at each iteration a subset of the components of $\underline{y}$ to reconstruct $\underline{x}$. We denote by $\underline{y}_{(k)}$ these successive mini-batches. Bayes formula gives the posterior distribution over $\underline{x}$ after seeing k mini-batches

$$p(\underline{x}|\underline{y}_{(k)}, \{\underline{y}_{(k-1)}, \cdots \underline{y}_{(1)}\}) = \frac{p(\underline{y}_{(k)}|\underline{x})p(\underline{x}|\{\underline{y}_{(k-1)}, \cdots \underline{y}_{(1)}\})}{\int \mathrm{d}\underline{x}\; p(\underline{y}_{(k)}|\underline{x})p(\underline{x}|\{\underline{y}_{(k-1)}, \cdots \underline{y}_{(1)}\})}. \tag{3.75}$$

This formula suggests the iterative scheme of using as a prior on $\underline{x}$ at iteration k the posterior obtained at iteration $k-1$. This idea can be implemented in different approximate inference algorithms, as proposed by [20] using a variational method. In the regular version of AMP an effective factorized posterior is given at convergence by the input update functions (3.35)-(3.37):

$$p(x|\underline{y}, \underline{\underline{W}}) \simeq \prod_{i=1}^{N} \frac{1}{\mathcal{Z}_x(\lambda_i, \sigma_i)} p_x(x_i) e^{-\frac{(\lambda_i - x_i)^2}{2\sigma_i}}. \tag{3.76}$$

Plugging this posterior approximation in the iterative scheme yields the mini-AMP algorithm using the converged values of $\lambda_{(\ell)_i}$ and $\sigma_{(\ell)_i}$ at each anterior mini-batch $\ell < k$ to compute the prior

$$p_x^{(k)}(\underline{x}) = p(\underline{x}|\{\underline{y}_{(k-1)}, \cdots \underline{y}_{(1)}\}, \underline{\underline{W}}) \simeq \prod_{i=1}^{N} \frac{1}{\mathcal{Z}_{x,i}}\; p_x(x_i)\; e^{-\sum_{\ell=1}^{k-1} \frac{(\lambda_{(\ell),i} - x_i)^2}{2\sigma_{(\ell),i}}}, \tag{3.77}$$

where the $\mathcal{Z}_{x,i}$ normalize each marginal factor. Compared to a naive mean-field variational approximation of the posterior, AMP takes into account more correlations and is indeed found to perform better in experiments reported by [85]. An other advantage of the AMP based online inference is that it is amenable to theoretical analysis by a corresponding State Evolution. In Chapter 6, we follow the lines of [85] to derive a streaming version of the Cal-AMP algorithm (see next section), for the training of minimal models of neural networks.

3.5.2 A special special case of GAMP: Cal-AMP

The calibration AMP algorithm was introduced in [130, 131], relatively to the problem of *blind calibration*. In the generic setting of the inference of an unknown signal from a set of observations,

one can imagine a situation where the channel is not perfectly known and needs to be calibrated. For the GLM problem, a possible formalization of this uncertainty is the addition of unobserved calibration variables, noted $\underline{s} \in \mathbb{R}^M$ here, in the channel likelihood:

$$p^s_{\text{out}}(\underline{y}|\underline{\underline{W}}\underline{x}, \underline{s}) = \prod_{\mu=1}^{M} p^s_{\text{out}}(y_\mu|\underline{w}_\mu^T\underline{x}, s_\mu) \iff y_\mu = g(\underline{w}_\mu^T\underline{x}; \epsilon_\mu, s_\mu), \quad \epsilon_\mu \sim p_\epsilon(\epsilon_\mu)\ \forall \mu. \tag{3.78}$$

The recovery of the value of $\underline{s}$ will only be feasible if multiple observations $\{\underline{y}_{(k)}\}$ are available. If a set of complementary pairs $\{\underline{x}_{(k)}, \underline{y}_{(k)}\}$ is furthermore available a priori, then one can imagine a *supervised* procedure to calibrate $\underline{s}$ before dealing with the original problem of reconstruction of $\underline{x}$. Otherwise, a blind calibration has to be attempted.

A Bayesian inference approach was proposed in [130, 131], using P observations $\{\underline{y}_{(k)}\}_{k=1}^P$ to compute simultaneously, in the Bayes-optimal setting, the MMSE estimators of the input signals $\{\underline{x}_{(k)}\}_{k=1}^P$ and the calibration $\underline{s}$. The posterior distribution examined including prior distributions is

$$p(\{\underline{x}_{(k)}\}_{k=1}^P, \underline{s}|\underline{y}, \underline{\underline{W}}) = \frac{1}{\mathcal{Z}(\underline{\underline{W}}, \{\underline{y}_{(k)}\}_{k=1}^P)} \prod_{k,\mu=1}^{P,M} p^s_{\text{out}}(y_{(k)_\mu}|\underline{w}_\mu^\top\underline{x}_{(k)}, s_\mu) \prod_{k,i=1}^{P,N} p_x(x_{(k)_i}) \prod_{\mu=1}^{M} p_s(s_\mu). \tag{3.79}$$

The Cal-AMP algorithm was derived following the steps of Section 3.3 (see [131] for details of this derivation). It is closely related to the GAMP for the reconstruction of the $\{\underline{x}_{(k)}\}_{k=1}^P$, with two main differences. First it considers P observations and P signals to reconstruct simultaneously where AMP only has one, thereby multiplying the number of update equations. Second, the update output functions (3.38)-(3.40) are modified so as to take into account the effective output channel with uncertain calibration variables

$$p_{\text{out}}(y|z) = \int \mathrm{d}s\ p^s_{\text{out}}(y|z, s)p_s(s). \tag{3.80}$$

The proposed strategy is numerically tested in [130, 131] for a series of sub-problems, and succeeds at the reconstruction task when M and/or P are large enough.

In Chapter 6, we re-derive the algorithm using a slightly less factorized version of the posterior (recovering the algorithm of [130, 131] as a particular case). This is the starting point towards the analysis of Bayesian learning in multi-layer neural networks, an ongoing work presented in this last Chapter.

3.5.3 Algorithms and free energies beyond i.i.d. matrices

The derivations outlined in the previous Sections of the equivalent replica, TAP and AMP equations required the weight matrices to have Gaussian i.i.d. entries. Different weight statistics are a priori feasible if one finds a way to perform the corresponding disorder average in the replica computation or to evaluate the corresponding Onsager term in the TAP equations. For AMP however one considers a high connectivity limit of the BP equations relying on the independence of incoming messages at each variable or factor node. This high connectivity limit, albeit introducing short loops, turns out correct in the examined models with i.i.d. weights as suggests the consistency between AMP, TAP and replica equations. Rigorous proof were also given for the SK model [143] and the GLM [9]. Nevertheless, this derivation may appear incorrect for correlated weight matrices that are promoting dependencies between messages.

Efforts to broaden in practice the class of matrices amenable to such mean-field treatments lead to a series of works in statistical physics and signal processing with related propositions. Parisi and Potters pioneered this direction by deriving mean-field equations for orthogonal weight matrices using a high-temperature expansion [111]. The adaptive TAP approach proposed by Opper and Winther [107, 108] further allowed for inference in densely connected graphical models without prior knowledge on the weight statistics. The Onsager term of these TAP equations was evaluated using the cavity method for a given weight sample. The resulting equations were then understood

to be a particular case of the Expectation Propagation (EP) [92] - belonging to the class of message passing algorithms for approximate inference - yet applied in densely connected models [109]. An associated approximation of the free energy called Expectation Consistently (EC) was additionally derived from the EP messages. Subsequently, Kabashima and collaborators [135, 136, 66] focused on the perceptron and the GLM to propose TAP equations and a replica derivation of the free energy for the ensemble of orthogonally invariant random weight matrices. In the singular value decomposition of such weight matrices $\underline{\underline{W}} = \underline{\underline{U}}\,\underline{\underline{S}}\,\underline{\underline{V}}^\top \in \mathbb{R}^{M\times N}$ the orthogonal basis matrices $\underline{\underline{U}}$ and $\underline{\underline{V}}$ are drawn uniformly at random from respectively $\mathrm{O}(M)$ and $\mathrm{O}(N)$, while the diagonal matrix of singular values $\underline{\underline{S}}$ has an arbitrary spectrum. The consistency between the EC free energy and the replica derivation for orthogonally invariant matrices was verified by [68] for signal recovery from linear measurements (the GLM without G). From the algorithmic perspective, Fletcher, Rangan and Schniter [117, 129] applied the EP to the GLM to obtain the (Generalized) Vector-Approximate Message Passing (G-VAMP) algorithm. Remarkably, these authors proved that the behavior of the algorithm could be characterized in the thermodynamic limit, provided the weight matrix is drawn from the orthogonally invariant ensemble, by a set of scalar State Evolution equations similarly to the AMP algorithm. These equations are again related to the saddle point equations of the replica free energy. Concurrently, Opper, Cakmak and Winther proposed an alternative procedure for solving TAP equations with orthogonally invariant weight matrices in Ising spin systems relying on an analysis of iterative algorithms [103].

Below we present the aforementioned free energy as proposed by [135, 136, 66], and the G-VAMP algorithm of [129].

Replica free energy for the GLM in the Bayes Optimal setting

Consider the ensemble of orthogonally invariant weight matrices with spectral density $\sum_{i=1}^N \delta(\lambda - \lambda_i)/N$ of their 'square' $\underline{\underline{W}}\underline{\underline{W}}^\top$ converging in the thermodynamic limit $N \to +\infty$ to $\rho_\lambda(\lambda)$. The quenched free energy of the GLM in the Bayes optimal setting derived in [66, 136] writes

$$-f = \mathrm{extr}_{q\hat{q}} \left[-\frac{1}{2} q\hat{q} + \mathcal{I}_x(\hat{q}) + \mathcal{J}_z(q_0, q, \alpha, \rho_\lambda) \right] \tag{3.81}$$

$$\mathcal{J}_z(q_0, q, \alpha, \rho_\lambda) = \mathrm{extr}_{u\hat{u}} \left[F_{\rho_\lambda,\alpha}(q_0 - q, \hat{u}/\lambda_0) + \frac{\hat{u}q_0}{2} - \frac{\alpha\hat{u}u}{2\lambda_0} + \alpha\mathcal{I}_z(q_0\lambda_0/\alpha, u) \right], \tag{3.82}$$

where $\mathcal{I}_x$ and $\mathcal{I}_z$ were defined as (3.73)-(3.74) and the spectral density $\rho_\lambda(\lambda)$ appears via its mean $\lambda_0 = \mathbb{E}_\lambda[\lambda]$ and in the definition of

$$\begin{aligned} F_{\rho_\lambda,\alpha}(q, u) = \frac{1}{2} \mathrm{extr}_{\Lambda_q,\Lambda_u} \left[-(\alpha - 1)\log\Lambda_u - \mathbb{E}_\lambda \log(\Lambda_u\Lambda_q + \lambda) + \Lambda_q q + \alpha\Lambda_u u \right] \\ -\frac{1}{2}(\log q + 1) + \frac{\alpha}{2}(\log u + 1). \end{aligned} \tag{3.83}$$

Gaussian random matrices are a particular case of the considered ensemble. Their singular values are characterized asymptotically by the Marcenko-Pastur distribution [86]. In this case, one can check that the above expression reduces to (3.72). More generally, note that $\mathcal{J}_z$ generalizes $\mathcal{I}_z$.

Vector Approximate Message Passing for the GLM

The VAMP algorithm consists in writing EP [92] with Gaussian messages on the factor graph representation of the GLM posterior distribution given on Figure 3.2. The estimation of the signal $\underline{x}$ is decomposed onto four variables, two duplicates of $\underline{x}$ itself and two duplicates of the linear transformation $\underline{z} = \underline{\underline{W}}\,\underline{x}$. The potential functions ψ_x and ψ_z of factors connecting copies of the same variable are Dirac distributions enforcing their equality. The factor node linking $\underline{z}^{(2)}$ and $\underline{x}^{(2)}$ is assumed Gaussian with variance going to zero. The procedure of derivation, equivalent to the projection of the BP algorithm on Gaussian messages, is recalled in Appendix A and leads to Algorithm 2. Like for AMP, the algorithm features some auxiliary variables introduced along the derivation. At convergence the duplicated $\hat{\underline{x}}_1$, $\hat{\underline{x}}_2$ (and $\hat{\underline{z}}_1$, $\hat{\underline{z}}_2$) are equal and either can be returned by the algorithm as an estimator. For readability, we omitted the time indices in the iterations that here simply follow the indicated update.

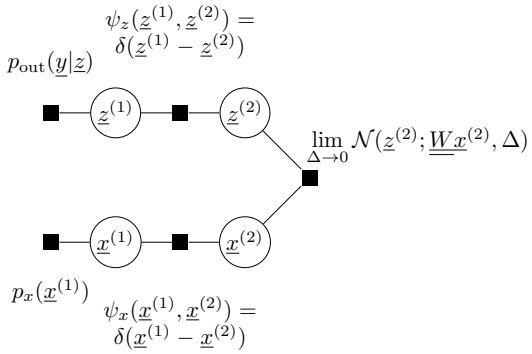


Figure 3.2: Factor graph representation of the GLM for the derivation of VAMP

Algorithm 2 Vector Approximate Message Passing

Input: vector of observations $\underline{y} \in \mathbb{R}^M$ and weight matrix $\underline{\underline{W}} \in \mathbb{R}^{M \times N}$:
Initialize: $\underline{\underline{A}}_1^x$, $\underline{B}_1^x$, $\underline{\underline{A}}_1^z$, $\underline{B}_1^z$
repeat

$$\hat{\underline{x}}_1 = f_1^x(\underline{B}_1^x, \underline{\underline{A}}_1^x), \qquad \underline{\underline{C}}_1^x = f_2^x(\underline{B}_1^x, \underline{\underline{A}}_1^x) \tag{3.84}$$

$$\underline{\underline{A}}_2^x = \underline{\underline{C}}_1^{x\ -1} - \underline{\underline{A}}_1^x, \qquad \underline{B}_2^x = \underline{\underline{C}}_1^{x\ -1}\hat{\underline{x}}_2 - \underline{B}_1^x \tag{3.85}$$

$$\hat{\underline{z}}_1 = f_1^z(\underline{B}_1^z, \underline{\underline{A}}_1^z), \qquad \underline{\underline{C}}^z{}_1 = f_2^z(\underline{B}_1^z, \underline{\underline{A}}_1^z) \tag{3.86}$$

$$\underline{\underline{A}}_2^z = \underline{\underline{C}}_1^{x\ -1} - \underline{\underline{A}}_1^z, \qquad \underline{B}_2^z = \underline{\underline{C}}_1^{x\ -1}\hat{\underline{x}}_2 - \underline{B}_1^z \tag{3.87}$$

$$\hat{\underline{x}}_2 = g_1^x(\underline{B}_2^x, A_2^x, \underline{B}_2^z, A_2^z), \qquad \underline{\underline{C}}_2^x = g_2^x(\underline{B}_2^x, A_2^x, \underline{B}_2^z, A_2^z) \tag{3.88}$$

$$\underline{\underline{A}}_1^x = \underline{\underline{C}}_2^{x\ -1} - \underline{\underline{A}}_2^x, \qquad \underline{B}_1^x = \underline{\underline{C}}_2^{x\ -1}\hat{\underline{x}}_1 - \underline{B}_2^x \tag{3.89}$$

$$\hat{\underline{z}}_2 = g_1^z(\underline{B}_2^x, A_2^x, \underline{B}_2^z, A_2^z), \qquad \underline{\underline{C}}^z{}_2 = g_2^z(\underline{B}_2^x, A_2^x, \underline{B}_2^z, A_2^z) \tag{3.90}$$

$$\underline{\underline{A}}_1^z = \underline{\underline{C}}_2^{x\ -1} - \underline{\underline{A}}_2^z, \qquad \underline{B}_1^z = \underline{\underline{C}}_2^{x\ -1}\hat{\underline{x}}_1 - \underline{B}_2^z \tag{3.91}$$

until convergence
Output: signal estimate $\hat{\underline{x}}_1 \in \mathbb{R}^N$, and estimated covariance $\underline{\underline{C}}_1^x \in \mathbb{R}^{N \times N}$

For a given instance of the GLM inference problem, i.e. a given weight matrix $\underline{\underline{W}}$, one can always launch either the AMP algorithm or the VAMP algorithm to attempt the reconstruction. If the weight matrix has i.i.d. zero mean Gaussian entries, the two strategies are conjectured to be equivalent and GAMP can be provably convergent for certain settings [116]. If the weight matrix is not Gaussian but orthogonally invariant, then only VAMP is expected to always converge. More generally, even in cases where none of these assumptions are verified, VAMP has been observed to have less convergence issues than AMP.

Like for AMP, a State Evolution can also be derived for VAMP (which was actually directly proposed for the multi-layer GLM [38]). It rigorously characterizes the behavior of the algorithm when $\underline{\underline{W}}$ is orthogonally invariant. One can also verify that the SE fixed points can be mapped to the solutions of the saddle point equations of the replica free energy (3.81) (see Section 1 of Supplementary Material of [40]); so that the robust algorithmic procedure can advantageously be used to compute the fixed points to be plugged in the replica potential to approximate the free energy.

3.5.4 Model composition

Another recent and ongoing direction of extension of mean-field methods is the combination of solutions to elementary models to tackle more sophisticated inference questions. The graphical representations introduced in Section 2.1.1 are here of great help. In a complicated joint probability distribution, it is sometimes possible to identify well-known sub-models, such as the GLM or the RBM already discussed in this thesis. Understanding how and when it is justified to plug-in different solutions is of course non-trivial and a very promising direction of research. For instance, in Chapter 4, we perform an ad hoc combination of the RBM and GLM model to design a practical algorithm operating on arbitrary (real) data sets, albeit not yet analyzable. An instead theoretically grounded extension in this direction is the treatment of multi-layer GLMs, or in other words multi-layer neural networks. In [84] a multi-layer version of AMP is derived, assuming Gaussian i.i.d weight matrices, along with a State Evolution and a free energy. In [38], the multi-layer version of the VAMP algorithm is derived with the corresponding State Evolution for orthogonally invariant weight matrices. In Chapter 5, we derive the corresponding replica free energy. Yet another example, in Chapter 6, we derive AMP and SE while combining a calibration problem with a GLM. The derivation presented there will follow the lines of ML-AMP although in a slightly more general case.

Part II

Contributions

4 Mean-field inference for (deep) unsupervised learning

Unsupervised learning often consists in fitting a parametric probabilistic model to a collection of empirical samples. To this end, neural networks are a precious tool. They can define joint probability distributions with complex correlations able to reproduce data from the real world such as images or sounds. As exact inference on large neural networks is almost never possible, it is interesting to consider mean-field methods in this context. They are precisely designed to exploit structures and properties of interactions in large graphs. Yet, they also rely on certain assumptions of randomness and homogeneity which can be broken when parameters are chosen by a training algorithm. In this Chapter, we show that mean-field methods can indeed be helpful in the unsupervised training and downstream application of a class of neural network models called Boltzmann machines. In a first Section, we present the derivation of mean-field equations for approximate inference in Boltzmann machines. In particular, we generalize the usual case of binary variables to arbitrary real-valued variables. In a second Section, we leverage the formalism derived in the first Section to propose a new training algorithm for Restricted Boltzmann Machines. We verify numerically that the mean-field strategy achieves state-of-art performance. These two first sections are based on the publications [42] and [150]. In a third Section, we discuss an application where the representative power of RBMs can be exploited through mean-field inference for a reconstruction task. The designed method successfully integrates unsupervised learning to improve Bayesian reconstruction. This project was led by E. W. Tramel and published in [151]. Finally, as perspectives, we hint at further directions leveraging mean-field methods for the nowadays more common feed forward models used for unsupervised learning.

4.1 Mean-field inference in Boltzmann machines

We start by presenting mean-field approximations for Boltzmann machines with an arbitrary architecture before turning to the specific case of the Restricted and Deep Boltzmann machines - RBMs and DBMs - that are of special interest in machine learning. To derive TAP free energies, we choose here the formalism of Georges-Yedidia [47] which allows to compute systematically corrections to the naive mean-field approximation by the addition of orders to the expansion.

4.1.1 Georges-Yedidia expansion for binary Boltzmann machines

We consider the binary vector $\underline{x} \in \{0,1\}^N$ following the distribution defined by a Boltzmann machine with symmetric weight matrix $\underline{\underline{W}} \in \mathbb{R}^{N\times N}$ and bias vector $\underline{b} \in \mathbb{R}^N$,

$$E(\underline{x}) = -\underline{b}^\top \underline{x} - \frac{1}{2}\underline{x}^\top \underline{\underline{W}}\underline{x}, \quad p(\underline{x}) = \frac{1}{\mathcal{Z}} e^{-\beta E(\underline{x})}. \tag{4.1}$$

This model is closely related to the pairwise Ising model for which the TAP equations [144], the Plefka expansion [114] and the Georges-Yedidia expansion [47] were originally derived. The only difference consists in assuming $x_i = 0$ or 1 instead of $x_i = \pm 1$. The derivation already outlined in Section 3.2.1 can be found in detail for this original case in different references including [105] and [159]. Following these steps we obtain the Georges-Yedidia expansion for the binary Boltzmann

machine for $\underline{x}_i \in \{0,1\}$ up to the third order as a function of the marginals (or magnetizations),

$$\begin{aligned}-\beta G(\underline{m}) = &-\sum_{i=1}^{N} m_i \log m_i + (1-m_i)\log(1-m_i) + \beta \sum_{i=1}^{N} b_i m_i + \beta \sum_{(ij)} W_{ij} m_i m_j \\ &+ \frac{\beta^2}{2} \sum_{(ij)} W_{ij}^2 (m_i - {m_i}^2)(m_j - {m_j}^2) \\ &+ \frac{2\beta^3}{3} \sum_{(ij)} W_{ij}^3 (m_i - {m_i}^2)(1/2 - m_i)(m_j - {m_j}^2)(1/2 - m_j) \\ &+ \beta^3 \sum_{(ijk)} W_{ij} W_{jk} W_{ki} (m_i - {m_i}^2)(m_j - {m_j}^2)(m_k - {m_k}^2) + \cdots\end{aligned} \tag{4.2}$$

where the summation over (ij) runs over distinct pairs of connected variables and similarly (ijk) runs over distinct triplets of interconnected variables. Note that in the above energy definition (4.1) we assumed a fully connected graph of connections, yet any other architecture can be recovered by setting some entries of $\underline{\underline{W}}$ to zero. To recover an approximation of the free energy, we must fix the values of the marginals by finding the stationary points m_i^* of G,

$$\left.\frac{\partial G(\underline{m})}{\partial m_i}\right|_{m_i^*} = 0 \quad \forall i = 1 \cdots N \quad \Rightarrow \quad G(\underline{m}^*). \tag{4.3}$$

These conditions can be formulated as a set of self-consistent equations verified by the m_i^* (noted simply m_i in the following),

$$\begin{aligned}m_i = \sigma\Big(&\beta b_i + \sum_{j \in \partial i} \beta W_{ij} m_j - \beta^2 W_{ij}^2 (m_j - 1/2)(m_i - {m_i}^2) \\ &+ \frac{2\beta^3}{3} \sum_{j \in \partial i} W_{ij}^3 \left[2(1/2 - m_i)^2 - (m_i - {m_i}^2)\right](m_j - {m_j}^2)(1/2 - m_j) \\ &+ \frac{\beta^3}{2} \sum_{(jk) \in \partial i} W_{ij} W_{jk} W_{ki} (1/2 - m_i)(m_j - {m_j}^2)(m_k - {m_k}^2) + \cdots\Big)\end{aligned} \tag{4.4}$$

where $\sigma(x) = (1+e^{-x})^{-1}$ is the sigmoid function, and $(jk) \in \partial i$ corresponds to the set of connected pairs of variables that are also connected to variable i. A truncation of the expansion (4.2) at a given order is reflected by a truncation of the sigmoid argument in (4.4). We refer to the resulting set of equations as the TAP equations of a given order, allowing to compute the corresponding TAP free energy of a given order. In particular, the naive mean-field approximation discussed in Section 3.1 is recovered by keeping only the first order.

4.1.2 Georges-Yedidia expansion for generalized Boltzmann machines

The binary variable case presented above can be recovered from a more general setting considering the x_i-s to be a priori real-valued. Formally we consider now $\underline{x} \in \mathbb{R}^N$ governed by the energy function and parametrized distribution

$$E(\underline{x}) = -\sum_{(ij)} W_{ij} x_i x_j - \frac{1}{\beta} \log p_x(x_i; \theta_i)\,, \quad p(\underline{x}) = \frac{1}{\mathcal{Z}} e^{\frac{\beta}{2} \underline{x}^\top \underline{\underline{W}} \underline{x}} \prod_{i=1}^{N} p_x(x_i; \theta_i), \tag{4.5}$$

where $p_x(x_i; \theta_i)$ is an arbitrary prior distribution with parameter θ_i. For Bernoulli prior with parameter $\sigma(\beta b_i)$ we recover the measure of binary Boltzmann machine. However we choose here a prior that does not depend on the temperature a priori. We now derive the Georges-Yedidia expansion for this general case following the outline discussed in 3.2.1, and highlighting the differences with the binary case.

We note that inference in the generalized fully connected Boltzmann machine is somehow related to the symmetric rank-1 matrix factorization problem, which also features pairwise interactions.

Similarly, inference for the bi-partite RBM maps to the asymmetric rank-1 matrix factorization. However, conversely to the Boltzmann inference, these factorizations are reconstruction problems. The mean-field techniques, derived in [80, 81], allow there to compute the MMSE estimator of unknown signals from approximate marginals. Here our main goal is to evaluate the free energy.

Minimization for fixed marginals

While fixing the value of the first moment is sufficient for binary variables, we now need more than one constraint in order to minimize the Gibbs free energy at a given value of the marginals. In the same spirit of the AMP algorithm we assume a Gaussian parametrization of the marginals. We note $\underline{a}$ the first moment of $\underline{x}$ and $\underline{c}$ its variance. We wish to compute the constrained minimum over the distributions q on $\mathbb{R}^N$

$$G(\underline{a},\underline{c}) = \min_q \left[\langle E(\underline{x})\rangle_q - H(q)/\beta \mid \langle \underline{x}\rangle_q = \underline{a}\,, \langle \underline{x}^2\rangle_q = \underline{a}^2 + \underline{c} \right], \tag{4.6}$$

where the notation of squared vectors corresponds here and below to the vectors of squared entries. It is equivalent to an unconstrained problem with Lagrange multipliers $\underline{\lambda}(\underline{a},\underline{c},\beta)$ and $\underline{\xi}(\underline{a},\underline{c},\beta)$

$$G(\underline{a},\underline{c}) = \min_q \left[\langle E(\underline{x})\rangle_q - H(q)/\beta - \underline{\lambda}^\top(\langle \underline{x}\rangle_q - \underline{a})/\beta - \underline{\xi}(\langle \underline{x}^2\rangle_q - \underline{a}^2 - \underline{c})/\beta \right]. \tag{4.7}$$

The terms depending on the distribution q in the the functional to minimize above can be interpreted as a Gibbs free energy for the effective energy functional

$$\tilde{E}(\underline{x}) = E(\underline{x}) - \underline{\lambda}^\top \underline{x}/\beta - \underline{\xi}^\top \underline{x}^2/\beta. \tag{4.8}$$

The solution of the minimization problem (4.10) is therefore the corresponding Boltzmann distribution

$$q_{\underline{a},\underline{c}}(\underline{x}) = \frac{e^{-\beta\tilde{E}(\underline{x})}}{\tilde{\mathcal{Z}}} = \frac{1}{\tilde{\mathcal{Z}}} e^{-\beta E(\underline{x}) + \underline{\lambda}(\underline{a},\underline{c},\beta)^\top \underline{x} + \underline{\xi}(\underline{a},\underline{c},\beta)^\top \underline{x}^2} \tag{4.9}$$

and the minimum $G(\underline{a},\underline{c})$ is

$$\begin{aligned} -\beta G(\underline{a},\underline{c}) &= -\underline{\lambda}^\top \underline{a} - \underline{\xi}^\top(\underline{a}^2 + \underline{c}) + \log \int \mathrm{d}\underline{x}\; e^{-\beta E(\underline{x}) + \underline{\lambda}^\top \underline{x} + \underline{\xi}^\top \underline{x}^2} \\ &= \log \int \mathrm{d}\underline{x}\; e^{-\beta E(\underline{x}) + \underline{\lambda}^\top (\underline{x} - \underline{a}) + \underline{\xi}^\top (\underline{x}^2 - \underline{a}^2 - \underline{c})}, \end{aligned} \tag{4.10}$$

where the Lagrange multipliers $\underline{\lambda}(\underline{a},\underline{c},\beta)$ and $\underline{\xi}(\underline{a},\underline{c},\beta)$ enforcing the constraints are still implicit. Defining a functional $\tilde{G}$ for arbitrary vectors $\tilde{\underline{\lambda}} \in \mathbb{R}^N$ and $\tilde{\underline{\xi}} \in \mathbb{R}^N$,

$$-\beta\tilde{G}(\underline{a},\underline{c},\tilde{\underline{\lambda}},\tilde{\underline{\xi}}) = \log \int \mathrm{d}\underline{x}\; e^{-\beta E(\underline{x}) + \tilde{\underline{\lambda}}^\top (\underline{x} - \underline{a}) + \tilde{\underline{\xi}}^\top (\underline{x}^2 - \underline{a}^2 - \underline{c})}, \tag{4.11}$$

we have

$$a_i = \langle x_i\rangle_{q_{\underline{a},\underline{c}}} \Rightarrow -\beta \left.\frac{\partial \tilde{G}}{\partial \tilde{\lambda}_i}\right|_{\underline{\lambda},\underline{\xi}} = 0, \qquad -\beta \left.\frac{\partial^2 \tilde{G}}{\partial \tilde{\lambda}_i^2}\right|_{\underline{\lambda},\underline{\xi}} = \langle x_i^2\rangle_{q_{\underline{a},\underline{c}}} - a_i^2 > 0, \tag{4.12}$$

$$c_i + a_i^2 = \langle x_i^2\rangle_{q_{\underline{a},\underline{c}}} \Rightarrow -\beta \left.\frac{\partial \tilde{G}}{\partial \tilde{\xi}_i}\right|_{\underline{\lambda},\underline{\xi}} = 0, \qquad -\beta \left.\frac{\partial^2 \tilde{G}}{\partial \tilde{\xi}_i^2}\right|_{\underline{\lambda},\underline{\xi}} = \langle (x_i^2)^2\rangle_{q_{\underline{a},\underline{c}}} - (c_i + a_i^2)^2 > 0. \tag{4.13}$$

Hence the Lagrange multipliers are identified as minimizers of $-\beta\tilde{G}$ and

$$-\beta G(\underline{a},\underline{c}) = -\beta\tilde{G}(\underline{a},\underline{c},\underline{\lambda}(\underline{a},\underline{c},\beta),\underline{\xi}(\underline{a},\underline{c},\beta)) = \min_{\tilde{\underline{\lambda}},\tilde{\underline{\xi}}} -\beta\tilde{G}(\underline{a},\underline{c},\tilde{\underline{\lambda}},\tilde{\underline{\xi}}). \tag{4.14}$$

The true free energy $F = -\log \mathcal{Z}/\beta$ would eventually be recovered by minimizing the constrained minimum $G(\underline{a},\underline{c})$ with respect to its arguments. Nevertheless, the computation of G and $\tilde{G}$ involves an integration over $\underline{x} \in \mathbb{R}^N$ and remains intractable. The following step of the Georges-Yedidia derivation consists in approximating these functionals by a Taylor expansion at infinite temperature where interactions are neutralized.

Expansion around $\beta = 0$

To perform the expansion we introduce the notation $A(\beta, \underline{a}, \underline{c}) = -\beta G(\underline{a}, \underline{c})$. We also define the auxiliary operator

$$U(\underline{x}; \beta) = -\frac{1}{2}\underline{x}^\top \underline{\underline{W}}\underline{x} + \frac{1}{2}\langle \underline{x}^\top \underline{\underline{W}}\underline{x}\rangle_{q_{\underline{a},\underline{c}}} - \sum_{i=1}^{N} \frac{\partial \lambda_i}{\partial \beta}(x_i - a_i) - \sum_{i=1}^{N} \frac{\partial \xi_i}{\partial \beta}(x_i^2 - a_i^2 - c_i), \tag{4.15}$$

that allows to write concisely for any observable O the derivative of its average with respect to β,

$$\frac{\partial \langle O(\underline{x}; \beta)\rangle_{q_{\underline{a},\underline{c}}}}{\partial \beta} = \left\langle \frac{\partial O(\underline{x}; \beta)}{\partial \beta} \right\rangle_{q_{\underline{a},\underline{c}}} - \langle U(\underline{x}; \beta) O(\underline{x}; \beta)\rangle_{q_{\underline{a},\underline{c}}}. \tag{4.16}$$

To compute the derivatives of $\underline{\lambda}$ and $\underline{\xi}$ with respect to β we note that

$$\frac{\partial A}{\partial a_i} = -\beta \frac{\partial \tilde{G}}{\partial a_i} = -\lambda_i(\beta, \underline{a}, \underline{c}) - 2a_i \xi_i(\beta, \underline{a}, \underline{c})\,, \tag{4.17}$$

$$\frac{\partial A}{\partial c_i} = -\beta \frac{\partial \tilde{G}}{\partial c_i} = -\xi_i(\beta, \underline{a}, \underline{c}), \tag{4.18}$$

where we used that $\partial \tilde{G}/\partial \tilde{\lambda}_i = 0$ and $\partial \tilde{G}/\partial \tilde{\xi}_i = 0$ when evaluated for $\underline{\lambda}(\underline{a}, \underline{c}, \beta)$ and $\underline{\xi}(\underline{a}, \underline{c}, \beta)$. Consequently,

$$\frac{\partial \xi_i}{\partial \beta} = -\frac{\partial}{\partial c_i}\frac{\partial A}{\partial \beta}\,, \qquad \frac{\partial \lambda_i}{\partial \beta} = -\frac{\partial}{\partial a_i}\frac{\partial A}{\partial \beta} + 2a_i \frac{\partial \xi_i}{\partial \beta}. \tag{4.19}$$

We can now proceed to compute the first terms of the expansion that will be performed for the functional A.

Zeroth order Substituting $\beta = 0$ in the definition of A we have

$$A(0, \underline{a}, \underline{c}) = -\underline{\lambda}(0, \underline{a}, \underline{c})^\top \underline{a} - \underline{\xi}(0, \underline{a}, \underline{c})^\top (\underline{a}^2 + \underline{c}) + \log \tilde{\mathcal{Z}}^0(\underline{\lambda}(0, \underline{a}, \underline{c}), \underline{\xi}(0, \underline{a}, \underline{c})), \tag{4.20}$$

with

$$\tilde{\mathcal{Z}}^0(\underline{\lambda}(0, \underline{a}, \underline{c}), \underline{\xi}(0, \underline{a}, \underline{c})) = \int \mathrm{d}\underline{x}\; e^{\underline{\lambda}(0,\underline{a},\underline{c})^\top \underline{x} + \underline{\xi}(0,\underline{a},\underline{c})^\top \underline{x}^2} \prod_{i=1}^{N} p_x(x_i; \theta_i) \tag{4.21}$$

$$= \prod_{i=1}^{N} \int \mathrm{d}x_i\; e^{\lambda_i(0,\underline{a},\underline{c}) x_i + \xi_i(0,\underline{a},\underline{c}) x_i^2} p_x(x_i; \theta_i). \tag{4.22}$$

At infinite temperature the interaction terms of the energy do not contribute so that the integral in $\tilde{\mathcal{Z}}^0$ factorizes and can be evaluated numerically in the event that it does not have a closed-form.

First order We compute the derivative of A with respect to β. We use again that $\underline{\lambda}(\underline{a}, \underline{c}, \beta)$ and $\underline{\xi}(\underline{a}, \underline{c}, \beta)$ are stationary points of $\tilde{G}$ to write

$$\frac{\partial A}{\partial \beta} = -\beta \frac{\partial \tilde{G}}{\partial \beta} = \frac{\partial}{\partial \beta}\left[\log \int \mathrm{d}\underline{x}\; e^{-\beta E(\underline{x}) + \underline{\lambda}(\underline{a},\underline{c},\beta)^\top (\underline{x} - \underline{a}) + \underline{\xi}(\underline{a},\underline{c},\beta)^\top (\underline{x}^2 - \underline{a}^2 - \underline{c})}\right] \tag{4.23}$$

$$= \left\langle \frac{\partial}{\partial \beta}(-\beta E(\underline{x})) + \frac{\partial \underline{\lambda}}{\partial \beta}^\top (\underline{x} - \underline{a}) + \frac{\partial \underline{\xi}}{\partial \beta}^\top (\underline{x}^2 - \underline{a}^2 - \underline{c}) \right\rangle_{q_{\underline{a},\underline{c}}} \tag{4.24}$$

$$= \frac{1}{2}\langle \underline{x}^\top \underline{\underline{W}}\underline{x}\rangle_{q_{\underline{a},\underline{c}}}. \tag{4.25}$$

At infinite temperature the average over the product of variables becomes a product of averages so that we have

$$\left.\frac{\partial A}{\partial \beta}\right|_{\beta=0} = \frac{1}{2}\underline{a}^\top \underline{\underline{W}}\underline{a} = \sum_{(ij)} W_{ij} a_i a_j. \tag{4.26}$$

Second order Using the first order derivative of A we can compute the derivatives of the Lagrange parameters (4.19) and the auxillary operator at infinite temperature,

$$\left.\frac{\partial \xi_i}{\partial \beta}\right|_{\beta=0} = 0\,, \qquad \left.\frac{\partial \lambda_i}{\partial \beta}\right|_{\beta=0} = -\sum_{j\in\partial i} W_{ij} a_j\,, \qquad U(\underline{x};0) = -\sum_{(ij)} W_{ij}(x_i - a_i)(x_j - a_j).$$

The second order derivative is then easily computed at infinite temperature

$$\left.\frac{\partial^2 A}{\partial \beta^2}\right|_{\beta=0} = \frac{1}{2}\frac{\partial}{\partial \beta}\left(\langle \underline{x}^\top \underline{\underline{W}}\, \underline{x}\rangle_{q_{\underline{a},\underline{c}}}\right)\Big|_{\beta=0} = -\frac{1}{2}\langle U(\underline{x};0)(\underline{x}^\top \underline{\underline{W}}\, \underline{x})\rangle^{\beta=0}_{q_{\underline{a},\underline{c}}} \tag{4.27}$$

$$= \sum_{(ij)} W_{ij}^2 \langle (x_i - a_i) x_i (x_j - a_j)\rangle^{\beta=0}_{q_{\underline{a},\underline{c}}} = \sum_{(ij)} W_{ij}^2 c_i c_j. \tag{4.28}$$

TAP free energy for the generalized Boltzmann machine

Stopping at the second order of the systematic expansion, and gathering the different terms derived above we have

$$-\beta G(\underline{a},\underline{c}) = -\underline{\lambda}(0,\underline{a},\underline{c})^\top \underline{a} - \underline{\xi}(0,\underline{a},\underline{c})^\top (\underline{a}^2 + \underline{c}) + \log \tilde{\mathcal{Z}}^0(\underline{\lambda}(0,\underline{a},\underline{c}), \underline{\xi}(0,\underline{a},\underline{c})) \tag{4.29}$$
$$+ \beta \sum_{(ij)} W_{ij} a_i a_j + \frac{\beta^2}{2}\sum_{(ij)} W_{ij}^2 c_i c_j,$$

where the values of the parameters $\underline{\lambda}(0,\underline{a},\underline{c})$ and $\underline{\xi}(0,\underline{a},\underline{c})$ are implicitly defined through the stationary conditions (4.12)-(4.13). The TAP approximation of the free energy also requires to consider the stationary points of the expanded expression as a function of $\underline{a}$ and $\underline{c}$.

This second condition yields the relations

$$-2\xi_i(0,\underline{a},\underline{c}) = -\beta^2 \sum_{j\in\partial i} W_{ij}^2 c_j = A_i \tag{4.30}$$

$$\lambda_i(0,\underline{a},\underline{c}) = A_i a_i + \beta \sum_{j\in\partial i} W_{ij} a_j = B_i \tag{4.31}$$

where we define new variables A_i and B_i. While the extremization with respect to the Lagrange multipliers gives

$$a_i = \frac{1}{\mathcal{Z}_i^x}\int \mathrm{d}x_i\; x_i p_x(x_i;\theta_i) e^{-\frac{A_i}{2}x_i^2 + B_i x_i} = f_1^x(B_i, A_i;\theta_i), \tag{4.32}$$

$$c_i = \frac{1}{\mathcal{Z}_i^x}\int \mathrm{d}x_i\; x_i^2 p_x(x_i;\theta_i) e^{-\frac{A_i}{2}x_i^2 + B_i x_i} - a_i^2 = f_2^x(B_i, A_i;\theta_i), \tag{4.33}$$

where we introduce update functions f_1^x and f_2^x with respect to the partition function

$$\mathcal{Z}_i^x(B_i, A_i;\theta_i) = \int \mathrm{d}x_i\; p_x(x_i;\theta_i) e^{-\frac{A_i}{2}x_i^2 + B_i x_i}. \tag{4.34}$$

Finally we can rewrite the TAP free energy as

$$-\beta G(\underline{a},\underline{c}) = -\underline{B}^\top \underline{a} + \underline{A}^\top(\underline{a}^2 + \underline{c})/2 + \sum_{i=1}^N \log \tilde{\mathcal{Z}}_i^{\,x}(B_i, A_i;\theta_i) \tag{4.35}$$
$$+ \beta \sum_{(ij)} W_{ij} a_i a_j + \frac{\beta^2}{2}\sum_{(ij)} W_{ij}^2 c_i c_j,$$

with the values of the parameters set by the self-consistency conditions (4.30), (4.31), (4.32) and (4.33), which are the TAP equations of the generalized Boltzmann machine at second order. Note that the naive mean-field equations are recovered by ignoring the second order terms in β^2.

Relation to message passing The TAP equations obtained above must correspond to the fixed points of the Approximate Message Passing (AMP) following the derivation from Belief Propagation (BP) that was presented in Section 3.3.3. In the Appendix B of [150] we derive the relaxed-BP equations for the generalized Boltzmann machine:

$$B_{i\to j}^{(t)} = \sum_{k\in\partial i\setminus j} \beta W_{ik} a_{k\to i}^{(t)}, \quad A_{i\to j}^{(t)} = -\sum_{k\in\partial i\setminus j} \beta^2 W_{ik}^2 c_{k\to i}^{(t)}, \tag{4.36}$$

$$a_{i\to j}^{(t)} = f_1^x(B_{i\to j}^{(t-1)}, A_{i\to j}^{(t-1)}; \theta_i), \quad c_{i\to j}^{(t)} = f_2^x(B_{i\to j}^{(t-1)}, A_{i\to j}^{(t-1)}; \theta_i). \tag{4.37}$$

To recover the TAP equations for them we define

$$B_i^{(t)} = \sum_{k\in\partial i} \beta W_{ik} a_{k\to i}^{(t)}, \quad A_i^{(t)} = -\sum_{k\in\partial i} \beta^2 W_{ik}^2 c_{k\to i}^{(t)}, \tag{4.38}$$

$$a_i^{(t)} = f_1^x(B_i^{(t-1)}, A_i^{(t-1)}; \theta_i), \quad c_i^{(t)} = f_2^x(B_i^{(t-1)}, A_i^{(t-1)}; \theta_i). \tag{4.39}$$

As $B_i^{(t)} = B_{i\to j}^{(t)} + \beta W_{ij} a_{j\to i}^{(t)}$ and $A_i^{(t)} = A_{i\to j}^{(t)} - \beta^2 W_{ij}^2 c_{j\to i}^{(t)}$ we have by developing f_2^x that $c_i^{(t)} = c_{i\to j}^{(t)} + O(\beta)$ so that

$$A_i^{(t)} = -\beta^2 \sum_{j\in\partial i} W_{ij}^2 c_j^{(t)} + o(\beta^2). \tag{4.40}$$

By developing f_1^x we also have

$$a_k^{(t)} = f_1^x(B_{k\to j}^{(t-1)} + \beta W_{kj} a_{j\to i}^{(t-1)}\ ,\ A_{k\to j}^{(t-1)} - \beta^2 W_{kj}^2 c_{j\to k}^{(t-1)}; \theta_i) \tag{4.41}$$

$$= a_{k\to j}^{(t)} + \frac{\partial f_1^x}{\partial B_k} \beta W_{kj} a_{j\to k}^{(t-1)} + O(\beta^2), \tag{4.42}$$

with $\frac{\partial f_1^x}{\partial B_k}(B_k^{(t-1)}, A_k^{(t-1)}; \theta_k) = c_k^{(t)}$. Finally, by replacing in the definition of B_i the messages we obtain

$$B_i^{(t)} = \sum_{k\in\partial i} \beta W_{ik} a_{k\to i}^{(t)} = \sum_{k\in\partial i} \beta W_{ik} a_k^{(t)} - \beta W_{ki} c_k^{(t)} a_{i\to k}^{(t-1)}. \tag{4.43}$$

As $a_{i\to k}^{(t-1)} = a_i^{(t-1)} + O(\beta)$ and using the definition of $A_i^{(t)}$, we finally recover

$$B_i^{(t)} = \sum_{k\in\partial i} \beta W_{ik} a_k^{(t)} + A_i^{(t)} a_i^{(t-1)}. \tag{4.44}$$

Hence we indeed recover the TAP equations as the AMP fixed points in (4.39), (4.40) and (4.44). Beyond the possibility to cross-check our results, the message passing derivation also specifies a scheme of updates to solve the self-consistency equations obtained by the Georges-Yedidia expansion. In the applications we consider below we should resort to this time indexing with good convergence properties [17].

Solutions of the TAP equations As already discussed in Section 3.2, the TAP equations do not necessarily admit a single solution. In practice, different fixed points are reached when considering different initializations of the iteration of the self-consistent equations. In the applications of the TAP formalism discussed in the remaining of this Chapter we will discuss the strategy we adopt to take into account this multiplicity.

4.1.3 Application to RBMs and DBMs

Generalized Restricted Botlzmann Machines

In machine learning applications the bipartite architecture of RBMs, that we recall on Figure 4.1, is of particular interest. Traditionally they have binary units, yet they can be generalized to

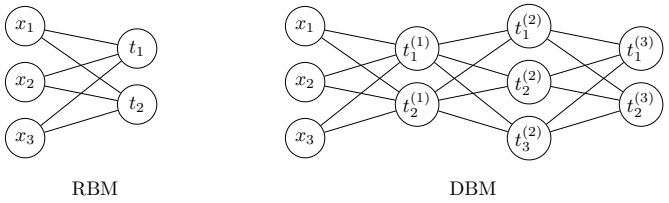


Figure 4.1: Graphical models for the Restricted Boltzmann Machine and the Deep Boltzmann Machine

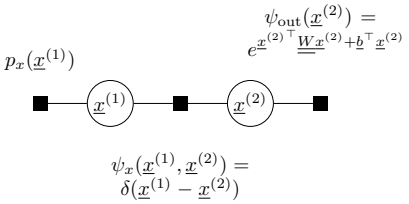


Figure 4.2: Fully connected BM factor graph for adaTAP

arbitrary domains (sometimes then called GRBM) following the extension we discussed above. The corresponding measure for a model with N input (or visible) units and M hidden units is

$$p(\underline{x}, \underline{t}; \underline{\theta}) = \prod_{i=1}^{N} p_x(x_i; \theta_i^x) \prod_{\alpha=1}^{M} p_h(t_\alpha; \theta_\alpha^h) e^{\underline{x}\,\underline{\underline{W}}\,\underline{t}}, \tag{4.45}$$

where the weight matrix $\underline{\underline{W}}$ has dimension $N \times M$ and the temperature is set as $\beta = 1$. Note that the expansion at small β can equivalently be interpreted as a small weight expansion here. We use the notation $\underline{\theta}$ for the collection of all the parameters of the model including $\underline{\theta}^x = (\theta_1^x, \cdots \theta_N^x)$, $\underline{\theta}^h = (\theta_1^h, \cdots \theta_M^h)$ and $\underline{\underline{W}}$. The TAP free energy (4.35) is rewritten as

$$\begin{aligned} -G_{\text{RBM}}(\underline{a}^x, \underline{c}^x, \underline{a}^h, \underline{c}^h) = &\sum_{i=1}^{N} -B_i^x a_i^x + A_i^x ({a_i^x}^2 + c_i^x)/2 + \log \mathcal{Z}_i^x(B_i^x, A_i^x; \theta_i^x)) \\ &+ \sum_{\alpha=1}^{M} -B_\alpha^h a_\alpha^h + A_\alpha^h ({a_\alpha^h}^2 + c_\alpha^h)/2 + \log \mathcal{Z}_\alpha^h(B_\alpha^h, A_\alpha^h; \theta_\alpha^h)) \\ &+ \sum_{(i\alpha)} W_{i\alpha} a_i^x a_\alpha^h + \sum_{(i\alpha)} W_{i\alpha}^2 c_i^x c_\alpha^h, \end{aligned} \tag{4.46}$$

where the sums over $(i\alpha)$ runs over all i-s and α-s and

$$\mathcal{Z}_i^x(B_i^x, A_i^x; \theta_i^x) = \int \mathrm{d}x\; p_x(x; \theta_i^x)\, e^{-A_i^x x^2/2 + B_i^x x}, \tag{4.47}$$

$$\mathcal{Z}_\alpha^h(B_\alpha^h, A_\alpha^h; \theta_\alpha^h) = \int \mathrm{d}t\; p_h(t; \theta_\alpha^h)\, e^{-A_\alpha^h t^2/2 + B_\alpha^h t}. \tag{4.48}$$

The values of the parameters are computed by running Algorithm 3 corresponding to AMP, where the differentiated visible and hidden update functions, f_1^x, f_2^x and f_1^h, f_2^h, are following the usual definition with respect to the partitions $\mathcal{Z}_i^x$ and $\mathcal{Z}_\alpha^h$.

Deep Botlzmann Machines

It is also possible to define *deep* models of Boltzmann machines by considering several stacked hidden layers. A Deep Boltzmann Machines (DBM) [123] with L hidden layers defines the distribution

$$p(\underline{x}, \underline{t}^{(1)}, .., \underline{t}^{(L)}; \underline{\theta}) = \frac{1}{\mathcal{Z}} e^{\underline{x}^\top \underline{\underline{W}} \underline{t}^{(1)} + \sum_{\ell=1}^{L-1} \underline{t}^{(\ell)\top} \underline{\underline{W}}^{(\ell)} \underline{t}^{(l+1)}} \prod_{i=1}^{N} p_x(x_i; \theta_i^x) \prod_{l=1}^{L} \prod_{\alpha=1}^{M_l} p_\alpha^{(\ell)}(t_\alpha^{(\ell)}; \theta_\alpha^{h\,(\ell)}). \tag{4.49}$$

Here again the specific architecture considered can be interpreted as a particular case of the general fully connected Boltzmann machine. The corresponding TAP free energy writes

$$
\begin{aligned}
-G_{\text{DBM}}(\underline{a}^x, \underline{c}^x, \{\underline{a}^{h,\ell}, \underline{c}^{h,\ell}\}_{\ell=1}^L) = &\sum_{i=1}^N -B_i^x a_i^x + A_i^x({a_i^x}^2 + c_i^x)/2 + \log \tilde{\mathcal{Z}}_i^{0,x}(B_i^x, A_i^x; \theta_i^x)) \\
&+ \sum_{\ell=1}^L \sum_{\alpha=1}^M -B_\alpha^{h,\ell} a_\alpha^{h,\ell} + A_\alpha^{h,\ell}({a_\alpha^{h,\ell}}^2 + c_\alpha^{h,\ell})/2 + \log \tilde{\mathcal{Z}}_\alpha^{0,h,\ell}(B_\alpha^{h,\ell}, A_\alpha^{h,\ell}; \theta_\alpha^{h,\ell})) \\
&+ \sum_{(i\alpha)} W_{i\alpha}^{(1)} a_i^x a_\alpha^{h,1} + \sum_{(i\alpha)} {W_{i\alpha}^{(1)}}^2 c_i^x c_\alpha^{h,1}, \\
&+ \sum_{\ell=1}^{L-1} \left(\sum_{(\alpha\beta)} W_{\alpha\beta}^{(\ell)} a_\alpha^{h,\ell} a_\beta^{h,\ell+1} + \sum_{(\alpha\beta)} {W_{\alpha\beta}^{(\ell)}}^2 c_\alpha^{h,\ell} c_\beta^{h,\ell+1} \right),
\end{aligned}
\tag{4.50}
$$

and the corresponding AMP algorithm straightforwardly follows from Algorithm 3. The DBM is also bipartite, with connections only between layers of even depth and odd depth. Therefore the TAP parameters of even and odd layers can be updated sequentially, as for the hidden and input parameters of the RBM.

Algorithm 3 TAP solutions for GRBMs

Input: $\underline{\underline{W}}$, $\underline{\theta}^x$, $\underline{\theta}^h$, $\underline{a}^{x,(0)}$, $\underline{c}^{x,(0)}$, τ
Initialize: $t = 0$

repeat
Hidden Side Updates
$A_\alpha^{h,(t+1)} = -\sum_i W_{i\alpha}^2 c_i^{x,(t)}$
$B_\alpha^{h,(t+1)} = A_\alpha^{h,(t+1)} a_\alpha^{h,(t)} + \sum_i W_{i\alpha} a_i^{x,(t)}$
$a_\alpha^{h,(t+1)} = f_1^h(B_\alpha^{h,(t+1)}, A_\alpha^{h,(t+1)}; \theta_\alpha^h)$
$c_\alpha^{h,(t+1)} = f_2^h(B_\alpha^{h,(t+1)}, A_\alpha^{h,(t+1)}; \theta_\alpha^h)$

Visible Side Updates
$A_i^{x,(t+1)} = -\sum_\alpha W_{i\alpha}^2 c_\alpha^{h,(t+1)}$
$B_i^{x,(t+1)} = A_i^{x,(t+1)} a_i^{x,(t)} + \sum_\alpha W_{i\alpha} a_\alpha^{h,(t+1)}$
$a_i^{x,(t+1)} = f_1^x(B_i^{x,(t+1)}, A_i^{x,(t+1)}; \theta_i^x)$
$c_i^{x,(t+1)} = f_2^x(B_i^{x,(t+1)}, A_i^{x,(t+1)}; \theta_i^x)$

$t = t + 1$
until Convergence or $t = \tau$

Algorithm 4 AdaTAP for binary BMs

Input: $\underline{\underline{J}}$, $\underline{b}$, τ
Initialize: $t = 0$, $\underline{\underline{A}}_1^{(0)}$, $\underline{B}_1^{(0)}$
repeat
Prior Updates
$a_{1,i}^{(t+1)} = \sigma(B_{1,i}^{(t)} - A_{1,i}^{(t)}/2)$
$c_{1,i}^{(t+1)} = a_{1,i}^{(t+1)}(1 - a_{1,i}^{(t+1)})$
$A_{2,i}^{(t+1)} = 1/c_{1,i}^{(t+1)} - A_{1,i}^{(t+1)}$
$B_{2,i}^{(t+1)} = 1/(1 - a_{1,i}^{(t+1)}) - B_{1,i}^{(t+1)}$

Interaction Updates
$\underline{\underline{C}}_2^{(t+1)} = (\underline{\underline{A}}_2^{(t+1)} - J)^{-1}$
$a_{2,i}^{(t+1)} = \sum_j \left(\underline{\underline{C}}_2^{(t+1)}\right)_{ij} \left(B_{2,j}^{(t+1)} + b_j\right)$
$A_{1,i}^{(t+1)} = 1/\left(\underline{\underline{C}}_2^{(t+1)}\right)_{ii} - A_{2,i}^{(t+1)}$
$B_{1,i}^{(t+1)} = a_{2,i}^{(t+1)} / \left(\underline{\underline{C}}_2^{(t+1)}\right)_{ii} - B_{2,i}^{(t+1)}$
$t = t + 1$
until Convergence or $t = \tau$

4.1.4 Adaptive-TAP fixed points for binary BM

As already discussed in Sections 3.1 and 3.2, for certain weight statistics, there exists an order of truncation of the Georges-Yedidia expansion which results in exact inference in the thermodynamic limit. In the general case nevertheless, the truncation is ad-hoc and adding orders allows to improve the accuracy of the approximation. Alternatively, one can prefer the adaptive TAP (adaTAP) strategy [108, 107, 109] which computes an Onsager correction to the naive mean-field adaptively to a given weight matrix and without any assumption on the underlying statistics of its entries.

The adaTAP equations originally derived by the cavity method [90] can also be recovered via different equivalent variants of message passing. We choose here the conventions of the VAMP [117] procedure, originally applied to the GLM. We focus on the case of the fully connected Boltzmann machine with binary units $\underline{x} \in \{0,1\}^N$ and weight matrix $\underline{\underline{J}} \in \mathbb{R}^{N \times N}$ and biases $\underline{b} \in \mathbb{R}^N$. Using the factor graph of Figure 4.2, we follow the steps described in Appendix A to obtain Algorithm 4,

where the input and output update functions were directly replaced, following the usual definitions:

Input functions

$$a_{1,i} = f_1^x(B_{1,i}, A_{1,i}) = \frac{1}{\mathcal{Z}_i^x} \int \mathrm{d}x\, x \left(\frac{1}{2}\delta(x) + \frac{1}{2}\delta(x-1)\right) e^{-A_{1,i}x^2/2 + B_{1,i}x} \tag{4.51}$$

$$c_{1,i} = f_2^x(B_{1,i}, A_{1,i}) = \frac{1}{\mathcal{Z}_i^x} \int \mathrm{d}x\, x^2 \left(\frac{1}{2}\delta(x) + \frac{1}{2}\delta(x-1)\right) e^{-A_{1,i}x^2/2 + B_{1,i}x} \tag{4.52}$$

Output/Interaction functions

$$a_{2,i} = g_1^x(\underline{B}_2, \underline{\underline{A}}_2) = \frac{1}{\mathcal{Z}_{\text{out}}} \int \mathrm{d}\underline{x}\, x_i \psi_{\text{out}}(\underline{x}) e^{-\underline{x}^\top \underline{\underline{A}}_2 \underline{x}/2 + \underline{B}_2^\top \underline{x}} \tag{4.53}$$

$$\underline{\underline{C}}_2 = g_2^x(\underline{B}_2, \underline{\underline{A}}_2) = \frac{1}{\mathcal{Z}_{\text{out}}} \int \mathrm{d}\underline{x}\, \underline{x}\underline{x}^\top \psi_{\text{out}}(\underline{x}) e^{-\underline{x}^\top \underline{\underline{A}}_2 \underline{x}/2 + \underline{B}_2^\top \underline{x}} \tag{4.54}$$

with the interaction potential $\psi_{\text{out}}(\underline{x}) = e^{\underline{x}^\top \underline{\underline{W}} \underline{x} + \underline{b}^\top \underline{x}}$ and $\underline{\underline{A}}_2 = \text{diag}(\underline{A}_2)$ the diagonal matrix with entries equal to the $A_{2,i}$. Compared to the VAMP Algorithm 2, we restrict the estimated covariances to be diagonal except for $\underline{\underline{C}}_2$, a common simplification.

The bottleneck in the execution time of Algorithm 4 is the evaluation of the matrix inverse in the update of $\underline{\underline{C}}_2$ which has a typical complexity of $O(N^3)$, and must be repeated at each iteration. Hence the price to pay for the adaTAP/VAMP algorithm is a significant increase of the computation time.

4.2 Applications to Boltzmann machine learning with hidden units

Boltzmann machines are used as parametric probabilistic models for unsupervised learning. The idea is to fit the weights and biases (or more generally local parameters for non-binary BMs) to a set of training data points so that the BM measure mimics the empirical data distribution. To achieve this goal the common objective of learning is to maximize the log-likelihood, yet an exact implementation of this approach is computationally intractable for all but the smallest models.

In the statistical physics community, Boltzmann machines typically do not feature any hidden units. This fully-visible Boltzmann architecture is justified as it corresponds to the model of maximum entropy under the constraint of fixed first and second moments. In fact the process of learning weights from a set of configurations at equilibrium is referred to as the inverse Ising problem in the physics literature. Learning algorithms based on sophisticated extensions of mean-field methods have been developed for these fully-visible Boltzmann machines(see [100] for a recent review). However, since they only include couplings between pairs of variables such models cannot successfully capture higher-order correlations. Conversely, in the RBM, the hidden layer mediates interactions between all the visible units and drastically increases the representative power of the model.

In the machine learning community fast approximate Monte Carlo methods, specifically contrastive divergence (CD)[55] and persistent CD (PCD) [98, 145], have made large-scale training of RBMs possible. These rather crude methods have popularized RBMs even though the reason of their efficiency remains unclear. We propose to examine an alternative strategy based on mean-field approximations. In the 80s and 90s, this direction was already investigated in a number of previous works [113, 54, 43, 70], albeit in small models with binary units and for artificial data sets very different from modern machine learning benchmarks. More recently, a deterministic training based on naive mean-field [157, 145] was tested in the large-scale and found to provide poor performance when compared to both CD and PCD. Here we show that going beyond naive mean-field allows to bridge the gap in efficiency with the approximate Monte Carlo methods. Additionally the deterministic mean-field framework offers a tractable way of evaluating the learning success by exploiting the mean-field observables.

4.2.1 Mean-field likelihood and deterministic training

Mean-field likelihood

In RBMs, the log-likelihood of a training set $\sum_{k=1}^{P} \log p(\underline{x}_{(k)};\underline{\theta})/P$, follows from the marginalization over the hidden units of the definition (4.45),

$$\log p(\underline{x};\underline{\theta}) = \sum_{i=1}^{N} \log p_x(x_i;\theta_i^x) + \sum_{\alpha=1}^{M} \int \mathrm{d}t_\alpha \ \log p_h(t_\alpha;\theta_\alpha^h) e^{\sum_i W_{i\alpha} t_\alpha x_i} - \log \mathcal{Z}(\underline{\theta}). \tag{4.55}$$

The log-partition it involves is nothing else but the intractable free energy of the RBM discussed in the beginning of this Chapter. It can be estimated using advanced, yet unfortunately costly, Monte Carlo methods such as Annealed Importance Sampling (AIS) [126]. Practitioners often prefer to monitor the surrogate pseudo-likelihood.[1] Instead, we propose to leverage mean-field inference to estimate this objective of learning and among other applications, help comparing different training algorithm quantitatively.

To compute the TAP free energy introduced in the previous Section, one needs to solve the TAP equations using Algorithm 3 and plug the solutions into the definition (4.46). We consider the TAP solutions resulting from initializations of the inference algorithm in data points of the training (or testing) set, $\underline{a}^{x,(0)} = \underline{x}_{(k)}$ and $\underline{c}^{x,(0)} = 0$ and average them uniformly. This choice follows from that the trained RBM will be used in a space of configurations neighboring the real data. Below, we investigate numerically the properties of the TAP solutions in learning experiments, and in particular find that their number grows as the weight magnitude (playing the role of an inverse temperature here) increases along the training.

The distribution of the weight entries after learning usually results from successive parameter updates and is a priori unknown. Therefore, the order of truncation of the small weight expansion is inevitably ad-hoc and the accuracy of the approximation hard to estimate. Alternatively, the AdaTAP free energy could be considered, yet at a higher computational cost as discussed in Section 4.1.4. In the numerical experiments presented in the following, we compare different orders and find that, albeit heuristic, the second order truncation yields a practical effective scoring of the learning process.

Mean-field training algorithm

The usual strategy for training RBMs consists in an approximate gradient ascent of the log-likelihood. The popular CD and PCD algorithms [55, 145] use fast Monte Carlo approximations of the gradients of $\log \mathcal{Z}(\underline{\theta})$. Alternatively we consider the gradients of the TAP free energy truncated at second order (4.46), which yields

$$\Delta\theta_i^x = \left\langle \frac{\partial}{\partial\theta_i^x} \log p_x(x_i;\theta_i^x) \right\rangle_{\underline{x}} - \frac{\partial}{\partial\theta_i^x} \log \mathcal{Z}_i^x(B_i^x, A_i^x;\theta_i^x)\,, \tag{4.56}$$

$$\Delta\theta_\alpha^h = \left\langle \frac{\partial}{\partial\theta_\alpha^h} \log p(t_\alpha|\underline{x};\underline{\theta}) \right\rangle_{\underline{x}} - \frac{\partial}{\partial\theta_\alpha^h} \log \mathcal{Z}_\alpha^h(B_\alpha^h, A_\alpha^h;\theta_\alpha^h)\,, \tag{4.57}$$

$$\Delta W_{i\alpha} = x_i f_1^h(\sum_{i=1}^{N} W_{i\alpha} x_i, 0;\theta_\alpha^h) - \sum_{(i\alpha)} \left(a_i^x a_\alpha^h + W_{i\alpha} c_i^x c_\alpha^h \right). \tag{4.58}$$

These directions of increment can be seen as approximations of the exact likelihood gradients or exact gradients of the mean-field likelihood. The latter can be regarded in itself as an objective of learning. The first term of the three gradients depend on the data point $\underline{x}$ and can be evaluated exactly. In practice we consider training with mini-batches over which these data-dependent terms are averaged. The second term in the gradient requires to compute TAP solutions, and corresponds to the part evaluated by Monte Carlo in CD. We compare multiple procedures:

[1] The pseudo likelihood consists in assuming $p(\underline{x}) \approx \prod_{i=1}^{N} p(x_i|\underline{x}_{\setminus i})$, where $\underline{x}_{\setminus i}$ stands for all the components of $\underline{x}$ expect the i-th, and the conditional probabilities have a closed form expression. Note however that its evaluation for one given $\underline{x}$ still requires as many operations as the size of the data space which is typically of a few hundreds.

- TAP-τ. We initialize the TAP inference in the data points of the mini-batch considered at the given parameter update and average the gradients over the TAP solutions retrieved after a given number τ of iterations of Algorithm 3. We speed up the gradient evaluation by an early stopping after τ steps, instead of converging the inference at each parameter update.

- Persistent-TAP-τ (P-TAP-τ). We initialize the TAP inference at the very beginning of training in a collection of randomly selected training data point (typically as many as the size of a mini-batch), and iterate the TAP inference for a few steps at each parameter update, always conserving the TAP solutions of the previous update as the initialization of the next one. Since the model parameters are only slightly changing between updates, TAP fixed points of neighboring steps of the gradient ascent are expected to be closely related, and the convergence from one to the next within a few iterations.

These strategies are comparable to CD-τ [55] and P-CD [145] where iterations of the inference algorithm are replaced by Gibbs sampling on the RBM.

In Algorithm 5, we summarize the mean-field training. The learning rate is noted γ. We include an ℓ_2 penalty with a coefficient ϵ, also called weight decay, promoting small weights; it is a common regularization of training for generalization that is all the more interesting here since the TAP likelihood relies on a small weight expansion. We also consider a momentum term involving the previous weight updates with a parameter η, known to speed up the training.

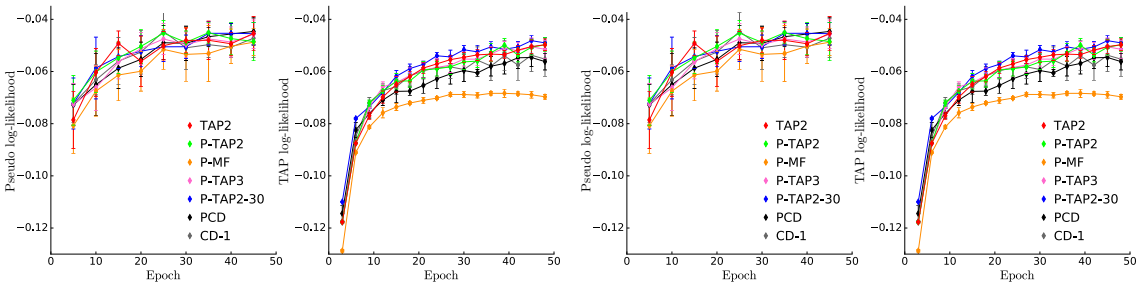
Algorithm 5 GRBM Mean-field training

Input: $\{\underline{x}_{(k)}\}_{k=1}^{P}$
Initialize: $W_{i\alpha}^{(0)} \sim \mathcal{N}(0,\sigma)$, $\underline{\theta}^{x,(0)}$, $\underline{\theta}^{h,(0)}$, $t=0$
repeat
  for All mini-batches $\{\underline{x}_{(k)}\}_{k=1}^{K}$ **do**
    if Persistent **and** $t>0$ **then**
      $\underline{a}_{(k)}^{(t+1)}, \underline{c}_{(k)}^{(t+1)}, \underline{B}_{(k)}^{(t+1)}, \underline{A}_{(k)}^{(t+1)} \leftarrow$ Alg. 3$(W^{(t)}, \underline{\theta}^{(t)}, \underline{a}_{(k)}^{x,(t-1)}, \underline{c}_{(k)}^{x,(t-1)}, \tau) \quad \forall k$
    else
      $\underline{a}_{(k)}^{(t+1)}, \underline{c}_{(k)}^{(t+1)}, \underline{B}_{(k)}^{(t+1)}, \underline{A}_{(k)}^{(t+1)} \leftarrow$ Alg. 3$(W^{(t)}, \underline{\theta}^{(t)}, \underline{x}_{(k)}, \underline{0}, \tau) \quad \forall k$
    end if
    $W_{i\alpha}^{(t+1)} \leftarrow W_{i\alpha}^{(t)} + \gamma \Delta W_{i\alpha}^{(t)} + \gamma\epsilon W_{i\alpha}^{2\,(t)} + \eta \Delta W_{i\alpha}^{(t-1)}$
    $\theta_{\alpha}^{h,(t+1)} \leftarrow \theta_{\alpha}^{h,(t)} + \gamma \Delta\theta_{\alpha}^{h,(t)}$
    $\theta_{i}^{x,(t+1)} \leftarrow \theta_{i}^{x,(t)} + \gamma \Delta\theta_{i}^{x,(t)}$

    $t = t+1$
  end for
until Convergence

	bin-MNIST	CalTech-silh.	bin-MNIST	real-MNIST	CBCL
Visible N	784	784	784	784	361
Hidden M	500	500	500	100,500	256
Batch K	100	256	100	100	20
Prior Vis.	B.	B.	B.	Tr. Gauss.-B.	Tr. Gauss.
Prior Hid.	B.	B.	B.	B.	B.
Learning γ	0.005	0.01	0.005	$10^{-2}, 10^{-5}$	0.005
Decay ϵ	0.001	0.001	0.001	0.001	0.01
Momentum η	0	0	0.5	0.5	0.5
Fig.	4.3a 4.4	4.3b 4.4b	4.5a	4.6a	4.6b

Table 4.1: Parameter settings for GRBM trainings.



(a) MNIST **(b)** Caltech 101 - Sihouette

Figure 4.3: Evolution of the normalized pseudo-log-likelihood and mean-field-log-likelihood over test sets during training for different training procedures (averages over 10 independent trainings with standard deviations reported as error bars). Hyperparameters of training are given in Table 4.1. **(a)** The naive mean-field appears as the least efficient, while others are approximately equivalent. **(b)** Persistent mean-field algorithms have difficulties in the likelihood in the first epochs of training. The complexity of the training set, 101 classes unevenly represented over only 4 100 training points, might explain this unexpected behavior. The persistent chains all converge to similar non-informative blurs in the earliest training epochs. Nevertheless,these algorithms eventually reach identical likelihood to the rest of the algorithms tested.

4.2.2 Numerical experiments

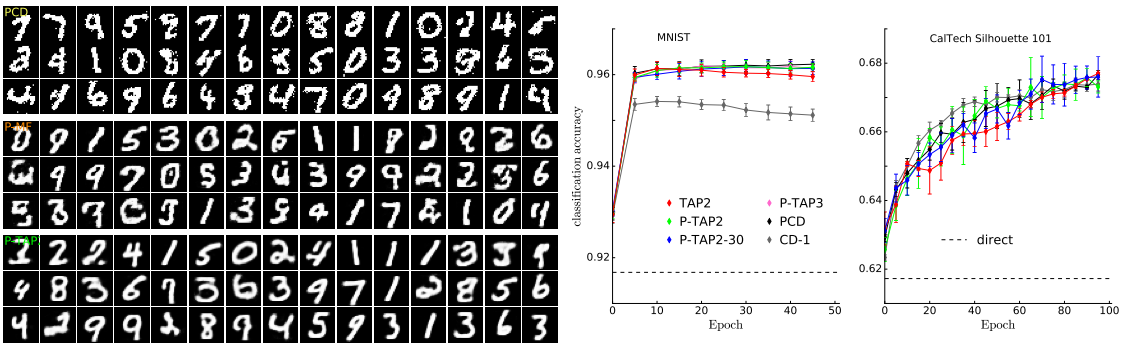
We reproduce here some of the numerical experiments published in [42] and [150].

Comparing orders of truncations and different gradient strategies on binary RBMs

In a first set of experiments we focus on binary RBMs to validate the proposed mean-field deterministic training, for which we provide a public Julia package [149]. We use two different data sets. The MNIST data set[2] contains handwritten digit images labeled from 0 to 9. Each image is comprised of 28×28 pixels taking values in the range $[0, 255]$. The MNIST data set was binarized by setting all non-zero pixels to 1. The 28×28 pixel version of the Caltech 101 Silhouette data set[3] consists of black regions of a primary foreground scene objects on a white background, labeled among 101 object classes according to the object in the original picture. Following previous studies evaluating RBMs on these data sets, we fix the number of RBM hidden units to 500. To compare RBM training approaches we do not include momentum nor adaptive learning rates here. When comparing learning procedures on the same plot, all free parameters of the training were set identically as reported in Table 4.1.

We test performances for mean-field trainings comparing the first-order (MF), second-order (TAP2), and third-order (TAP3) small weight expansions, with either persistent (prefix P) or reinitialized inference at each update. The inference algorithm is run for $\tau = 3$ iterations for all mean-field trainings expect for one of them (P-TAP2-30) where we wait for $\tau = 30$ iterations each time. For benchmark, we also train RBM models using CD-1, following the prescriptions of [53], and PCD, as implemented in [145]. On Figure 4.3a and 4.3b we report the evolution of the pseudo log-likelihood (left panels) and the second order mean-field likelihood (right panels). The ascent of the log-likelihood surrogates over training epochs is very similar across all methods. The mean-field trainings do maximize the pseudo-likelihood and, reciprocally, the approximate Monte Carlo trainings do maximize the TAP-likelihood. On MNIST, we compare the different orders of truncation of the mean-field expansion. As expected by design, the persistent TAP2 algorithm with 30 iterations of the inference (P-TAP2-30) achieves the best maximization of the TAP log-likelihood. However, P-TAP2, with only 3 iterations, achieves a very close performance, making it preferable for faster training. Moreover, we note that P-TAP2 demonstrates improvements in terms of pseudo-likelihood with respect to P-MF, in agreement with the previously noted disappointment

[2] http://yann.lecun.com/exdb/mnist/
[3] http://www.vision.caltech.edu/Image_Datasets/Caltech101/



(a) Peristent chains at 50 epochs of training **(b)** Downstream classification

Figure 4.4: Additional elements of comparison between training algorithms. Hyperparameters of training are given in Table 4.1. **(a)** 24 randomly selected persistent chains used to compute parameter updates in Algorithm 5 . Top: PCD, Middle: PMF, Bottom: P-TAP2. **(b)** Classification accuracy achieved by a logistic regression on the hidden representation at different stage of learning.

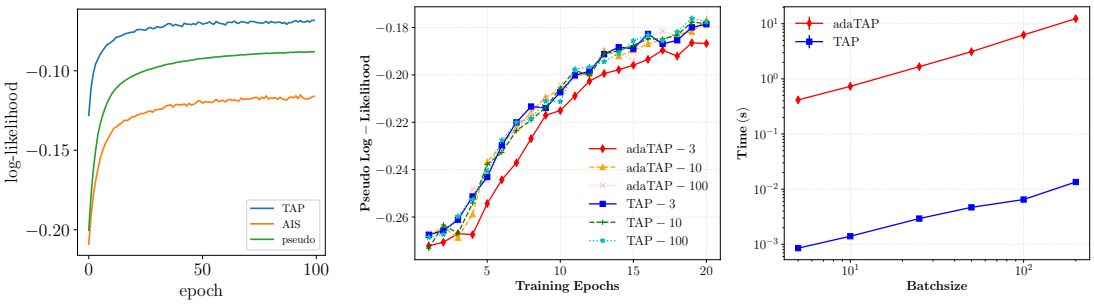
of naive mean-field [157, 145] and the improvements brought by the consideration of the second order observed in toy examples [43, 70]. However, the P-TAP3 does not yield significantly better pseudo-likelihood results than P-TAP2. The third order term of the small weight expansion, given by (4.2) for fully connected binary BMs, does not feature the triangular diagrams for the bipartite RBM. As a result it sums over as many terms as the second order, but at a higher order in the small $W_{i\alpha}$-s, and is finally expected to be sub-leading. In the following we mainly focus on the second order expansion.

On Figure 4.4 we examine further the quality of the different trainings. The persistent chains in Figure 4.4a gives an idea of the performance of the RBM as a generative model. The PCD samples and the P-TAP2 marginal means are recognizable digits. While the chains extracted from a P-MF training are of poorer quality, with half of them featuring non-identifiable digits. Here we further confirm the necessity to include the second order of the expansion to achieve competitive training. On Figure 4.4b we also evaluate these RBM training algorithms through the prism of a downstream task. We consider the supervised classification, via logistic regression, of the two data sets from the learned hidden marginals $\underline{a}^h(\underline{x}) = p(\underline{t}|\underline{x};\underline{\theta})$, interpreted as learned features. The MNIST classification accuracy of the RBMs trained with the P-TAP2 algorithms is roughly equivalent with that obtained when using PCD training, while CD-1 training yields markedly poorer classification accuracy. The slight decrease in performance of CD-1 and TAP2 along the training might be emblematic of over-fitting by the non-persistent algorithms, although no decrease in the TAP-likelihood of the test set was observed. For the Caltech 101 Silhouettes data set, the classification task is a priori much more difficult. The persistent algorithms do not yield better results on this task. However, we observe that the performance of deterministic mean-field RBM trainings is at least comparable with both CD-1 and PCD.

Lastly, we note the computation times for each of these approaches. For a Julia implementation of the tested RBM training techniques running on a 3.2 GHz Intel i5 processor, the average wall times for fitting a single 100-sample batch normalized against the model complexity are 14.10 ± 0.97 μs/batch/unit for PCD, which uses only a single sampling step, 21.25 ± 0.22 μs/batch/unit for PMF, 37.22 ± 0.34 μs/batch/unit for P-TAP2, and 64.88 ± 0.45 μs/batch/unit for P-TAP3, each of which use 3 inference iterations. Run times of the P-MF and P-TAP2 approaches are therefore comparable with PCD.

Annealed importance sampling and AdaTAP, pushing precision further?

On Figure 4.5a we compare the evolution for the previously considered likelihood surrogates: the TAP log-likelihood and the pseudo-likelihood, to the Annealed Importance Sampling (AIS) [99] estimation of the likelihood as implemented in [126]. By considering Markov Chains at multiple temperatures, AIS produces an accurate, albeit costly, Monte Carlo approximation of partition functions. We find that during the considered mean-field training all three measures have the



(a) Estimation **(b)** Adaptive TAP training

Figure 4.5: (a) Comparison for the mean-field training of a binary RBM on MNIST of likelihood estimates. The AIS results are averaged over 100 runs. For each run, 14 500 intermediate distributions are generated between the initial and target distribution using the same schedule as in [126]. The pseudo-likelihood is computed using a stochastic approximation over 100 of the 784 pixels. Hyperparameters of training are given in Table 4.1. **(b) Left:** Evolution of the pseudo-likelihood along the training of a binary RBM with 784 visible units and 500 hidden units, on the 5000 first images of binary-MNIST, with a learning rate of 0.001 and batches of size 100. Different number of iterations τ of the TAP inference (Algorithm 3) and adaTAP inference (Algorithm 4) are compared. In all cases, a damping of 0.5 is used. **Right:** Computation time for one iteration of the inference algorithm, as a function of the batch size. Time is reported in seconds for identical experimental settings.

same qualitative behavior as a function of epochs, indicating that they are consistent in evaluating or guiding the training.

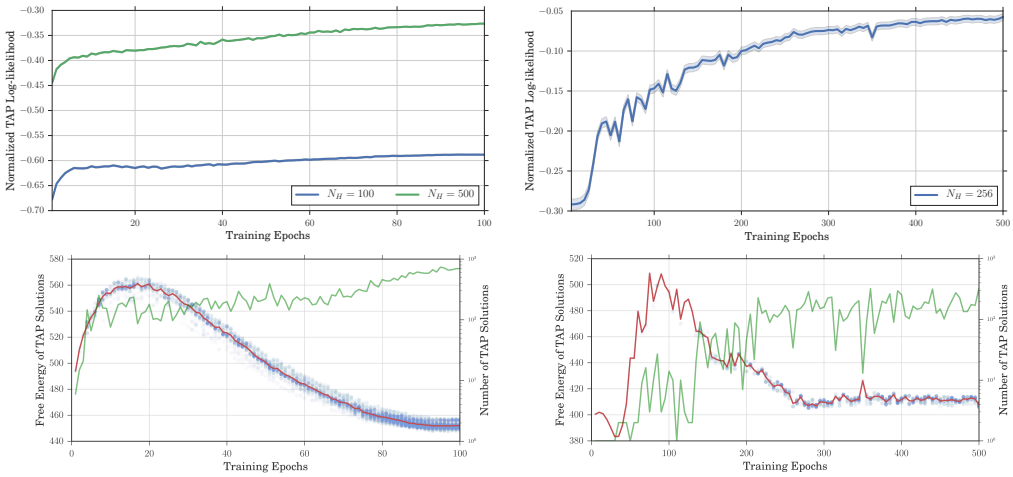
In the experiments presented above we found no improvement in mean-field training with the inclusion of the third order of the expansion. We investigate here whether replacing the inference based on the truncated expansion by the theoretically more principled adaTAP makes a difference. On Figure 4.5b the different curves correspond to different strategies of estimation of the likelihood gradients, with either TAP (Algorithm 3) or adaTAP (Algorithm 4) inference, again for a binary RBM training. All methods yield comparable results in terms of training performance, except for adaTAP with only three iterations of inference, which shows a poorer performance. Meanwhile, we also compare computational costs of the inference procedures. The need for a matrix inversion for each batch element makes adaTAP 3 orders of magnitude slower than TAP. Unfortunately the greater cost of the more accurate adaTAP is not compensated here by gains in training quality. Yet we simply replaced the TAP solutions by the AdaTAP fixed points in the mean-field gradients derived from the TAP free energy. It would be interesting to test whether using the gradients of the adaTAP free energy would yield any different outcome in terms of performance. Still, the computational time of the method would remain an issue.

Efforts to increase the accuracy beyond the heuristic application of the TAP approximation are therefore not paid by great improvements. We thus conclude that, as far as proposing a tractable and efficient training algorithm for RBMs, the TAP inference is well indicated.

Beyond the binary RBMs

We now consider the mean-field training (now fixed at the second order) of real-valued RBMs. This extension allows to go beyond toy examples and consider the unsupervised learning of real-valued data sets. We use the MNIST data set without binarization. We keep binary hidden units and choose truncated Gauss-Bernoulli real-valued units to account for the image sparsity. We also consider the CBCL face database[4]. It consists of both face and non-face 8-bit gray-scale 19×19 pixel images. For our experiments, we utilize only the face images, on which we train an RBM with binary hidden units and truncated Gaussian visible units. The detail of the corresponding update functions for mean-field inference can be found in Appendix C of [150]. The real-valued case necessitates to pay careful attention to a few numerical issues also discussed in Appendix B and C of [150]. In particular, they are the reason we consider distributions with bounded support.

[4] http://cbcl.mit.edu/software-datasets/heisele/facerecognition-database.html



(a) real-MNIST **(b)** CBCL

Figure 4.6: Top row: Normalized (per-unit) TAP log-likelihood estimate computed for $10,000$ training data samples for MNIST and $2,400$ training samples for CBCL . The TAP free energy is estimated using the converged TAP solutions from initial conditions drawn from the data samples. Solid lines indicate the average normalized TAP log-likelihood over the tested training samples and shaded regions indicate standard error. **Bottom row:** Free energy estimates of TAP solutions as a function of training epochs. In the case of the MNIST experiments, the number of hidden units considered is $M = N_H = 100$. Among the same TAP solutions as above, we retrieve the unique ones and follow their number (green line) along the training. We report as transparent blue dots their TAP free energy. The estimate of the total free energy via uniform averaging is given as a red line. Hyperparameters of training are given in Table 4.1. This experiment was led by E. W. Tramel and A. Manoel.

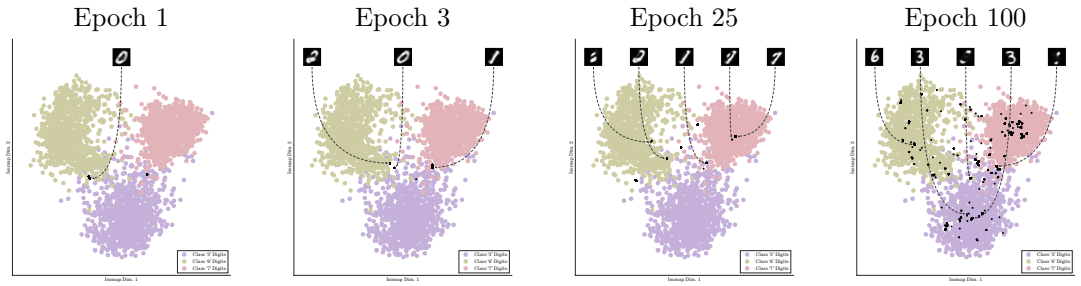
On the top row of Figure 4.6, we report the normalized TAP likelihood over successful trainings on both real-valued data sets.

In Appendix E of [150], we also demonstrate how the mean-field inference can be leveraged to train Deep Boltzmann Machines and report experiments for binary units trained on binary MNIST. The difference between DBMs and RBMs lies in the fact that beyond the log-partition, the data dependent terms in the likelihood and its gradients become intractable while they had closed-forms in the RBMs (4.55) and (4.56)-(4.58). This data-dependent terms are commonly approximated using the naive mean-field (i.e. first order) inference [123]. Instead we propose to consistently approximate the log-partition and the data-dependent term with the more powerful second-order TAP expansion. We report successful TAP likelihood maximization by the proposed algorithm. Yet DBMs are famously hard to train [126, 124, 97, 24], they require a layer-wise pre-training phase or the implementation of sophisticated tricks of weight tying. The proposed algorithm does not escape these limitations. Consequently, feed forward neural networks, VAEs [71, 120] and GANs [50], are currently preferred to exploit the advantage of deep architectures in unsupervised learning.

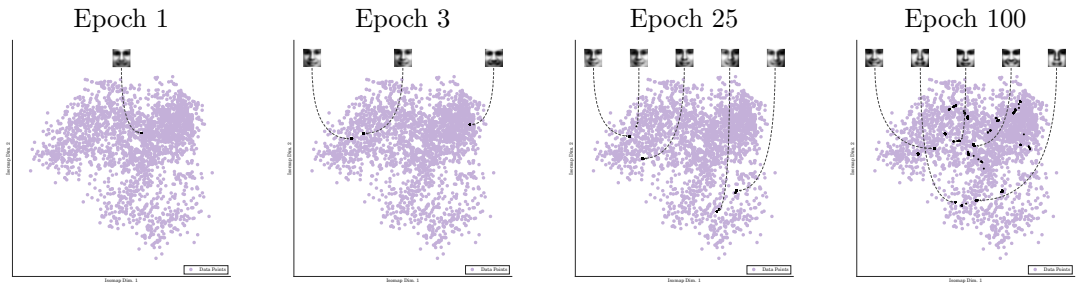
Probing the TAP solutions

The evaluation of unsupervised learning models is an open direction of research [122]. The deterministic mean-field framework provides a tractable estimate of the likelihood, but also allows to further characterize the learning state of BMs by looking at the TAP solutions. These attractors of the TAP inference gives an intuition on the representational power of a GRBM with a given set of parameters.

Previously, we examined the visual quality of the learned marginal means at the end of training (Figure 4.4a). On Figure 4.7, we use a two-dimensional embedding to compare the locations in configuration space of training data points, used as initializations of mean-field inference, and the corresponding TAP solutions, along the training on the real-valued data sets. At the beginning



(a) real-MNIST



(b) CBCL

Figure 4.7: Comparison of initial conditions (colored dots) and converged TAP solutions (black dots) for the tested data sets at different stages of training. For each data set, a two-dimensional Isomap embedding is calculated over the initialization data. Subsequently, the TAP solutions $\underline{a}_{(k)}$ are embedded in the same space. In each case, all initial variances are set to 0. Also, a random selection of the TAP solution marginals are plotted as images. **(a)** real-MNIST: Here, the ∼3,100 digits corresponding to the classes '3', '6', and '7' are drawn from the first 10,000 training samples of the MNIST data set as initializations. A reduced set of labels is used for readability. **(b)** CBCL: All available training face images are used as initializations. This experiment was led by E. W. Tramel and A. Manoel.

of training there are very few unique fixed points for the different initializations. The RBMs are not able to grasp the variety in the training data and has learned some kind of average. As the training progresses there are more and more TAP solutions that populate more evenly the space spanned by the training data points. The GRBM learns different representations rendering the complexity of the data. This observation is consistent with the intuition that the weight magnitude growing during the training is playing the role of an effectively decreasing temperature, inducing a multiplication of the TAP solutions as observed for the SK model [90]. Although we do not enter such a phase in our experiment, we could imagine the case where each initialization will lead to a different fixed point. This regime can be interpreted as an overtraining where the GRBM is merely memorizing the training data.

On the bottom row of Figure 4.6 we provide a quantitative analysis. We report the number of TAP solutions and their corresponding TAP free energies for the training of the GRBMs on real-MNIST and CBCL. We observe a fast growth in the number of fixed points at the beginning of training, followed by a relative stabilization. Moreover we note that the free energies of the different TAP solutions are concentrated compared to the overall evolution of the uniform average along the training. This implies that a weighted average would not result in a very different estimation of the overall free energy.

We have confirmed in numerical experiments that the mean-field free energy at second order allows to design an efficient training algorithm for BMs with hidden units. Moreover, the deterministic framework enables to track the progress of learning by choosing as an objective the tractable mean-field likelihood and by offering the TAP solutions as probes of the learned representations. There are no straightforward equivalent of these analyses using sampling. In the next Section, we show that mean-field BMs can also be advantageous in following applications, here as priors for Bayesian inference.

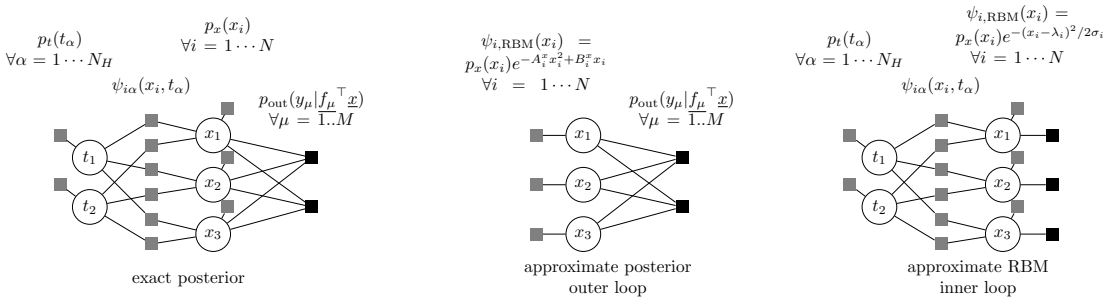


Figure 4.8: Left: Factor graph of the exact posterior including observations and RBM prior. **Middle:** Approximate posterior after mean-field inference in the RBM leading to an effective prior as seen by the outer loop of the algorithm. **Right:** Effective RBM factor graph incorporating information about the observations using the approximate posterior coming from the message passing on the CS problem.

4.3 Application to Bayesian reconstruction

We return here to the general problem of estimating an unknown signal $\underline{x}_0$ from observations $\underline{y}$ through a channel $p(\underline{y}|\underline{x})$ discussed in Section 2.1.3. Under the Bayesian paradigm, a prior distribution on the signal $p_x(\underline{x})$ that incorporates available information about the typical signal is used to help the reconstruction. Given the recent progress of unsupervised learning, an idea is to take advantage of a set of samples of the signal and learn a generative model to serve as a prior in Bayesian reconstruction. A potential hurdle to the approach remains that the joint distribution defined by the prior may be very informative only at the cost of being intractable and difficult to articulate in the Bayesian inference. Here we demonstrate on the example of an RBM prior for Compressed Sensing (CS) how mean-field methods, here envisaged through Approximate Message Passing, allow the implementation of such a program. This project was led by E. W. Tramel, so that we only review the main concepts and results here, and refer to the publication [151] for further details. In [150], we also apply this strategy in a denoising problem.

4.3.1 Combining CS and RBM inference

Compressed sensing consists in the reconstruction of a sparse signal $\underline{x}_0 \in \mathbb{R}^N$ from noisy linear measurements $\underline{y} \in R^M$. It is a special case of the GLM introduced in Section 3.3.1, with a Gaussian channel

$$p_{\rm out}(\underline{y}|\underline{\underline{F}}\underline{x}) = \prod_{\mu=1}^{M} \mathcal{N}(y_\mu; \underline{f}_\mu^\top \underline{x}, \Delta)\,, \tag{4.59}$$

where we note F the measurement matrix (with rows f_μ) with i.d.d. Gaussian entries of zero mean and variance $1/N$. Compared to the usual setting, we add here an RBM to model the correlations between the components of $\underline{x}$ resulting in the posterior

$$p(\underline{x}|\underline{y}) = \frac{1}{\mathcal{Z}(\underline{\theta})} p_x(x_i) p_{\rm out}(\underline{y}|\underline{\underline{F}}\underline{x}) \int \mathrm{d}\underline{t}\; p_t(\underline{t})\, e^{\underline{x}^\top \underline{\underline{W}}\underline{t}} \tag{4.60}$$

corresponding to the factor graph on the left of Figure 4.8. Considering both problems separately the mean-field inference of the CS problem is possible following the AMP Algorithm 1, while the mean-field inference of the RBM is realized by Algorithm 3. Both of these iterative procedures output factorized approximate marginals of $\underline{x}$. As a result they can heuristically be nested to design Algorithm 6, taking into account alternatively the observations $\underline{y}$ and the RBM prior information as schematized on the middle and right of Figure 4.8. In the Algorithm we have replaced the channel output functions following definitions 3.39-3.40,

$$\underline{g}_{\rm out}(y_\mu, \omega_\mu, V_\mu) = (y_\mu - \omega_\mu)/(V_\mu + \Delta)\,, \qquad \partial_\omega \underline{\underline{g}}_{\rm out}(y_\mu, \omega_\mu, V_\mu) = 1/(V_\mu + \Delta). \tag{4.61}$$

Algorithm 6 AMP for CS with RBM prior

Input: $\underline{y}$, F, $\underline{\underline{W}}$, $\underline{\theta}^x$, $\underline{\theta}^h$,
Initialize: $\hat{x}_i$, C_i^x $\forall i$ and ω_μ, V_μ $\forall \mu$, $t=0$

repeat *(outer loop)*

Update $V_\mu^{(t)} = \sum_{i=1}^N F_{\mu i}^2 C_i^{x\,(t)}$, $\omega_\mu^{(t)} = \sum_{i=1}^N F_{\mu i}\hat{x}_i^{(t)} - \sum_{i=1}^N F_{\mu i}^2 C^{x\,(t)}_i (y_\mu - \omega_\mu^{(t-1)})/(V_\mu^{(t-1)}+\Delta)$

Update $\sigma_i^{(t)\,-1} = -\sum_{\mu=1}^M F_{\mu i}^2/(V_\mu^{(t)}+\Delta)$, $\lambda_i^{(t)} = \hat{x}_i^{(t)} + \sigma_i^{(t)} \sum_{\mu=1}^M F_{\mu i}(y_\mu - \omega_\mu^{(t)})/(V_\mu^{(t)}+\Delta)$

run *(inner loop)*

Initialize $c_i^{x,(0)} = f_2^x(\lambda_i^{(t)}, \sigma_i^{(t)}, 0, 0)$, $a_i^{x,(0)} = f_1^x(\lambda_i^{(t)}, \sigma_i^{(t)}, 0, 0)$

Converge a_i^x, $c_i^x \leftarrow$ Alg. 3$(\underline{\underline{W}}, \underline{\theta}^x, \underline{\theta}^h, \underline{a}^{x,(0)}, \underline{c}^{x,(0)})$.

Update $\hat{x}_i^{(t+1)} = \gamma \hat{x}_i^{(t)} + a_i^x$, $C_i^{x\,(t+1)} = \gamma C_i^{x\,(t)} + c_i^x$

$t = t+1$

until Convergence

Output: $\underline{\hat{x}}$

The input update functions used to initialize and run Algorithm 3 take as argument the mean-field parameters coming from the RBM and the CS inference. They are given by,

$$\mathcal{Z}_i^x(\lambda_i, \sigma_i, A_i^x, B_i^x) = \int \mathrm{d}x\; p_x(x)\, e^{-(x-\lambda_i)^2/2\sigma_i}\, e^{-A_i^x x_i^2/2 + B_i^x x_i}\,, \tag{4.62}$$

$$f_1^x(\lambda_i, \sigma_i, A_i^x, B_i^x) = \frac{1}{\mathcal{Z}_i^x} \int \mathrm{d}x\; x\, p_x(x)\, e^{-(x-\lambda_i)^2/2\sigma_i}\, e^{-A_i^x x_i^2/2 + B_i^x x_i}\,, \tag{4.63}$$

$$f_2^x(\lambda_i, \sigma_i, A_i^x, B_i^x) = \frac{1}{\mathcal{Z}_i^x} \int \mathrm{d}x\; x^2\, p_x(x)\, e^{-(x-\lambda_i)^2/2\sigma_i}\, e^{-A_i^x x_i^2/2 + B_i^x x_i}. \tag{4.64}$$

In the last line of the outer loop, the update of the means and variance $\hat{x}_i$ and C_i^x involve a damping parameter γ between 0 and 1, which is found to stabilize the reconstruction in experiments. In the next section we review the performance of the proposed Algorithm for the reconstruction of MNIST digits.

Finally, we note that the construction of Algorithm 6 is heuristic and one could have considered for instance parallel updates rather than nested loops. The theoretical analysis of such an algorithm remains an open direction of research. In particular, future works should clarify whether, under the teacher-student scenario, a State Evolution can be derived and whether it is consistent with a replica computation.

4.3.2 Numerical experiments

To test the performances of the proposed strategy we focus on the MNIST data set featuring sparse images of handwritten digits. A vanilla AMP implementation of CS consists in assuming a sparse prior on the images and run the GLM Algorithm 1. We consider here such an implementation, where we use a Gaussian-Bernoulli prior with sparsity level ρ_i computed for each of the 784 pixels by an empirical average over the training data. In this sense, the prior is not identical across components of the signal but it does not take into account correlations. This method will serve as a reference case for the following two other implementations with RBM priors.

The idea to exploit RBMs in CS applications was pioneered by [36] and [148], who trained binary RBMs (BRBM) using Contrastive Divergence to locate the support of the non-zeros entries of sparse signals, and combined it with an AMP reconstruction. The knowledge of the location of the non-zeros entries indeed allows to solve the CS problem as soon as their number K is matched by the number of measurements M, which is an information theoretic threshold. We consider this

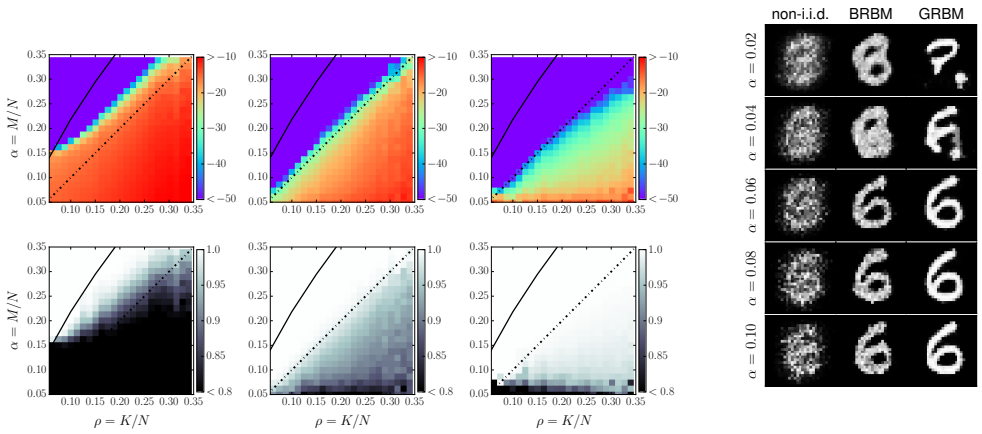


Figure 4.9: **Left:** CS reconstruction performance over the first $1,000$ digit images of the MNIST test set in the plane sparsity ρ-measurement ratio α. Results for non-i.i.d. factorized prior, the binary RBM and the GRBM are on the left, center, and right, respectively. The $M = K$ information theoretic transition is indicated by the black dotted line, and the AMP transition for i.i.d factorized prior [77] by the solid one. *Top:* Average reconstruction accuracy in MSE measured in dB. *Bottom:* Average reconstruction correlation $(\underline{x}_0 - \bar{\underline{x}_0})^T(\hat{\underline{x}} - \bar{\hat{\underline{x}}})/\sigma_{\underline{x}_0}\sigma_{\hat{\underline{x}}}$ with original digit image. **Right:** Visual comparison of reconstructions for a single digit image ($\rho = 0.25$) for small values of α.

strategy albeit using the deterministic mean-field training for RBMs discussed at the beginning of this Chapter. Finally, we consider the strategy of combination derived in the previous Section with a GRBM with truncated Gauss-Bernoulli visible units. The hyperparameters of training can be found in [151].

In the reconstruction experiments we consider the testing samples of MNIST, which were not seen by the RBMs at learning time. We recall that the measurement matrix has i.i.d Gaussian entries with zero mean and variance $1/N$. We choose a channel noise of $\Delta = 10^{-8}$. We compare the performances of three approaches on Figure 4.9, as a function of the sparsity level $\rho = K/N$ and the measurement ratio $\alpha = M/N$. We observe an improvement of the reconstruction as the priors become more informative. In particular, for the GRBM, a near perfect reconstruction can be reached even below the 'information theoretic threshold'. Not only the support is correct, but the values of the pixels as well are well recovered below the oracle $K = M$. The parameters of the RBM learned on the training set of MNIST are pointing towards a reconstructed signal very close to the ground-truth signal even when the information available is strictly insufficient to determine it exactly. On the right of the Figure, we also provide images for the reconstruction of a given MNIST digit with the three different approaches as a function of the measurement ratio. Both reconstruction algorithms involving RBMs manage to clearly reconstruct a '6' for $\alpha \approx 0.25\rho$. The GRBM further offers a reconstructed signal visually better than the BRBM.

Here we took advantage of the mean-field inference on real-valued BMs to derive a practical algorithm incorporating a learned prior in a Bayesian reconstruction problem. When a set of training examples of the signal is available, such a strategy should enable great economies in terms of the new information to acquire to reconstruct new samples, pushing further the idea of compressed sensing. There is a growing body of works investigating such scenarios. In the last paragraph of the following perspectives we discuss why we expect mean-field approaches to be especially relevant here.

4.4 Perspectives

Statistical physics theory of Restricted Boltzmann Machines

We presented in this Chapter contributions leveraging mean-field approximations to exploit the representational power of Boltzmann machines with hidden units in practical machine learning applications. Meanwhile, the recent craze for deep learning also motivated theoretical studies of

RBMs following the statistical physics tradition and using similar tools.

Studying an ensemble of RBMs with random parameters, Tubiana and Monasson [152] evidenced different regimes of typical pattern of activations in the hidden units controlled by parameters as the sparsity of the weights, their strength (playing the role of an effective temperature) or the type of prior for the hidden layer. Their study contributes to the understanding of the conditions under which the RBMs can represent high-order correlations between input units, albeit without including data and learning in their model. Barra and collaborators [10, 11], exploited the connections between the Hopfield model and RBMs to characterize RBM learning understood as an associative memory. They characterize the retrieval phase of RBMs. Mézard [87] also re-examined retrieval in the Hopfield model using its RBM representation, showing in particular that the addition of correlations between memorized patterns could still allow for a mean-field treatment at the price of a supplementary hidden layer in the BMs representation. This result remarkably draws a theoretical link between correlations in the training data and the necessity of depth in neural network models.

While the above results do not include characterization of the learning, a few others were able to discuss the dynamics of training. Huang [62] studied the Bayesian leaning of a RBM with a single hidden unit and binary weights. Decelle and collaborators [32, 31] introduced an ensemble of RBMs characterized by the spectral properties of the weight matrix and derived the typical dynamics of the corresponding order parameters during learning driven by data. Finally, Barra and collaborators [10] empirically study a teacher-student scenario of unsupervised learning by maximum likelihood on samples of an Hopfield model which they can compare to their theoretical characterization of the retrieval phase.

Feed forward generative models

While BMs are a natural interest of statistical physicists given its relations to inverse Ising problems and the Hopfield model, current state-of-the-art in unsupervised learning relies on feed forward neural networks used as VAEs [71, 120] or GANs [50]. These probabilistic models also define powerful but intractable probability distributions. Recent advances in mean-field theory and message-passing on multi-layer networks offer an interesting toolbox to design practical algorithms or develop a theoretical understanding of these models. Below, we give one example of practical application we started to investigate and comment on one direction of research of particular interest in our views.

An attempt at VAE training Similarly to RBMs, feed forward generative models can be trained for density modelling by maximum likelihood. Again as for RBMs, this objective is nevertheless intractable and in practice approximated. In VAEs, a variational lower bound of the likelihood is parametrized by an auxiliary neural network, itself optimized in parallel of the training of the generative model. Instead the likelihood and its gradients could be approximated by mean-field methods allowing for a learning procedure similar to the RBM training algorithm presented in Section 4.2. Indeed the likelihood of the feed forward architecture is directly related to the Bethe free energy corresponding to the recently derived multi-layer AMP of [84]. Nevertheless, running preliminary experiments in this direction on a single layer VAE, we encountered numerical problems. After a few epochs of learning the AMP algorithm had convergence issues, probably due to building correlations in the weights [116]. Although not obvious at the time, it seems that the ML-VAMP algorithm [38] could alternatively be used to compute the TAP solutions bypassing this difficulty. In [110], the authors demonstrate a successful inference for an in-painting experiment in a deep generative model trained on MNIST as a VAE.

GANs and VAEs as learned priors for pratice and theory In the line of the application presented in Section 4.3, several works have also investigated using feed forward generative models for inference tasks, and in particular for compressed sensing [18, 51, 52, 93]. In these works the inference is performed via gradient descent using back-propagation to compute gradients with respect to the input of the deep generative models. Using again the recently derived multi-layer version of AMP and VAMP, it would be interesting to replicate these propositions in the mean-field framework and compare empirical performances of the different algorithms.

A relevant property of these algorithms is that they can be characterized by State Evolution equations, paving the way to a theoretical teacher-student analysis of the reconstruction in these kind of settings, in the lines of the analysis performed for vanilla CS [77, 160]. More generally, these multi-layer models offer indeed interesting possibilities to build scenarios with non-trivial priors remaining tractable in the mean-field framework. In the next Chapter, we put in application this observation to design model experiments of deep learning trainings. Recently, [6] also took advantage of the method to extend the mean-field theory of low-rank matrix factorization to structured priors.

5 Mean-field inference for information theory in deep supervised learning

In networks made of thousands of neurons, we must find aggregated quantities to describe and hopefully explain the learning process. In particular, their generalization ability, remains a great puzzle [161]. Recently, it was proposed to consider the mutual information between random variables defined as the layers of a neural network [147]. The idea stems from the Information Bottleneck principle [146] proposing an information based objective for learning useful representations in probabilistic graphical models. Yet the verification of its relevance in practice requires the computation of mutual information for high-dimensional variables, a notoriously hard inference problem. Pioneering works in this direction focused either on small network models with discrete (continuous, eventually binned) activations [137], or on linear networks [128]. Using mean-field inference we are able to approximate mutual informations for a class of deep neural networks trained in the teacher-student scenario. This strategy allows us to capture some of the key properties of the deep learning setting that are usually difficult to include in tractable frameworks: non-linearities, arbitrary large width and depth, correlations in the input data and trainable weights.

In the first Section of this Chapter we present the extension to multi-layer networks of the mean-field approximation for entropies of [66, 135, 136]. In the following Section we describe how the resulting formula can be leveraged to study learning in a teacher-student scenario, and examine numerical experiments. The ideas presented in these two first parts were published in [40]. In a last Section we discuss more generally the potentialities and difficulties surrounding the information theoretic approach to deep learning theory, through presenting original numerical results and summarizing the debate that took place in the literature. This last part is un-published and results from a collaboration with Léon Bottou.

5.1 Mean-field entropy for multi-layer models

5.1.1 Extension of the replica formula to multi-layer networks

Multi-layer model

We consider the following statistical model involving a stochastic neural network of arbitrary depth ℓ,

$$\underline{\underline{W}}^{(k)} \sim p_W^{(k)}(\underline{\underline{W}}^{(k)}) \quad \forall k = 1, \cdots \ell \tag{5.1}$$

$$\underline{t}^{(0)} = \underline{x} \sim p_x(\underline{x}) = \prod_{i=1}^{N_0} p_x(x_i) \tag{5.2}$$

$$\underline{t}^{(k)} \sim p_{\text{out}}^{(k)}(\underline{t}^{(k)}|\underline{\underline{W}}^{(k)}\underline{t}^{(l-1)}) = \prod_{\mu=1}^{N_l} p_{\text{out}}^{(k)}(t_\mu^{(k)}|\underline{w}_\mu^{(k)\top}\underline{t}^{(l-1)}) \quad \forall k = 1, \cdots \ell \tag{5.3}$$

where the weight matrices are independent samples from the orthogonal invariant ensemble. This means that in their singular value decomposition $\underline{\underline{W}}^{(k)} = \underline{\underline{U}}^{(k)}\underline{\underline{S}}^{(k)}\underline{\underline{V}}^{(k)\top} \in \mathbb{R}^{N_l \times N_{l-1}}$, the basis matrices are Haar distributed: drawn uniformly from the set of orthogonal matrices $\mathrm{O}(N_l)$ and $\mathrm{O}(N_{l-1})$. We further assume the ratios $\tilde{\alpha}_k = N_k/N_0$ to be fixed when N_0 is varied. In the following we consider a fixed realization of the weight matrices, playing the role of quenched disorder. The model then defines a Markov Chain following the graphical model represented on Figure 5.1a. The distribution of a given variable $\underline{t}^{(k)}$ is intractable since it requires to marginalize over the inputs and the preceding layers in the Markov Chain.

Replica formula

One-layer For a single layer, the described model corresponds to a GLM (see 3.3.1) of output $\underline{t}^{(1)}$ with orthogonally invariant weight matrix, for which the posterior $p(\underline{x}|\underline{\underline{W}}^{(1)}, \underline{t}^{(1)})$ given by Bayes

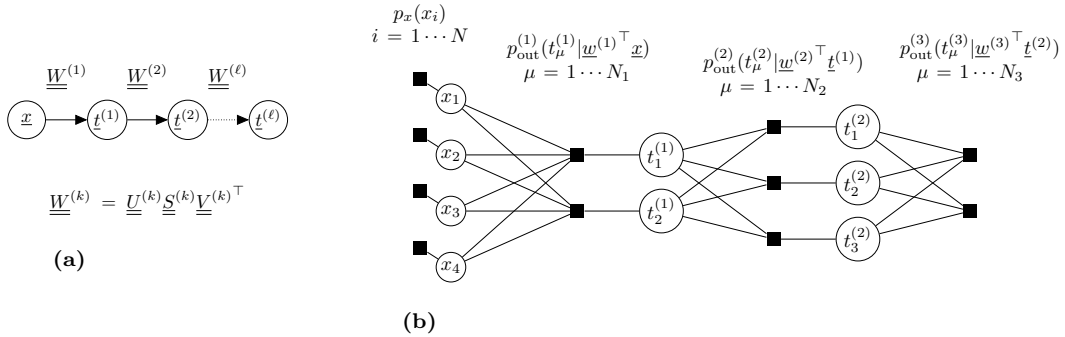


Figure 5.1: **(a)** Multi-layer stochastic neural network with orthogonally invariant weights: the matrices $\underline{\underline{U}}^{(k)}$ and $\underline{\underline{V}}^{(k)}$ are independent samples of the Haar measure. **(b)** Factor graph representation of the 3-layer GLM posterior distribution $p(\underline{x}|\underline{\underline{W}}^{(1)}, \underline{\underline{W}}^{(2)}, \underline{\underline{W}}^{(3)}, t^{(3)})$.

rule is explicit up to an intractable partition function $\mathcal{Z}(\underline{t}^{(1)}, \underline{\underline{W}}^{(1)})$. The corresponding quenched free energy, averaged over the realizations of the weight matrices, can however be approximated using a replica computation in the thermodynamic limit $N_0 \to +\infty$ following [66, 135, 136] as already discussed in Section 3.5.3. In the Bayes optimal setting, where the model generating the observations and the model used for reconstruction are identical, this free energy is in fact equal to the entropy of the output averaged over the disorder:

$$\mathcal{Z}(\underline{t}^{(1)}, \underline{\underline{W}}^{(1)}) = \int \mathrm{d}\underline{x}\; p_{\text{out}}(\underline{t}^{(1)}|\underline{\underline{W}}^{(1)}\underline{x}) p_x(\underline{x}) = p(\underline{t}^{(1)}|\underline{\underline{W}}^{(1)}), \tag{5.4}$$

$$\begin{aligned}\mathbb{E}_{\underline{\underline{W}}^{(1)}, \underline{t}^{(1)}}\left[\log \mathcal{Z}(\underline{t}^{(1)}, \underline{\underline{W}}^{(1)})\right] &= \mathbb{E}_{\underline{\underline{W}}^{(1)}}\left[\int \mathrm{d}\underline{t}^{(1)}\; p(\underline{t}^{(1)}|\underline{\underline{W}}^{(1)}) \log p(\underline{t}^{(1)}|\underline{\underline{W}}^{(1)})\right] \\ &= -\mathbb{E}_{\underline{\underline{W}}^{(1)}}\left[H(\underline{t}^{(1)}|\underline{\underline{W}}^{(1)})\right]. \end{aligned} \tag{5.5}$$

In the infinite-size limit the self-averaging property of the free energy ensures that the entropy for a given realization of the weight concentrates, this remains approximately true for large models. Hence the replica symmetric formula of [66, 135, 136] enables an asymptotic evaluation of $H(\underline{t}^{(1)}|\underline{\underline{W}}^{(1)})$. Note that there is no replica symmetry breaking in a Bayes optimal setting.

Compare to [66, 135, 136] and Section 3.5.3 we start by rewriting the single layer free energy (3.81) as

$$f = \lim_{N_0 \to \infty} \frac{1}{N_0} H(\underline{t}^{(1)}|\underline{\underline{W}}^{(1)}) \tag{5.6}$$

$$= -\text{extr}_{q,\hat{q},u,\hat{u}}\left[-\frac{1}{2}q\hat{q} + \frac{\hat{u}q_0}{2} - \frac{\alpha\hat{u}u}{2\lambda_0} + F_{\rho_\lambda,\alpha}(q_0 - q, \hat{u}/\lambda_0) + \mathcal{I}_x(\hat{q}) + \alpha\mathcal{I}_z(q_0\lambda_0/\alpha, u)\right] \tag{5.7}$$

$$= \text{extr}_{A,V,\tilde{A},\tilde{V}}\left[\phi(A, V, \tilde{A}, \tilde{V})\right] \text{ [1]} \tag{5.8}$$

with

$$\phi(A, V, \tilde{A}, \tilde{V}) = \frac{1}{2}\left[\tilde{A}(q_0 - V) - \alpha A\tilde{V} + \tilde{F}_W(AV)\right] - \mathcal{I}_x(\tilde{A}) - \alpha\mathcal{I}_z(q_0\lambda_0/\alpha, \tilde{V} - q_0\lambda_0/\alpha)$$

where [2]

$$\tilde{F}_{W^{(1)}}(x) = \min_\theta \left\{2\alpha\theta + (\alpha - 1)\log(1 - \theta) + \mathbb{E}_{\lambda_W} \log[x\lambda_W + (1 - \theta)(1 - \alpha\theta)]\right\}. \tag{5.9}$$

[1] The change of variable to recover the equivalence of the potentials is $A = \hat{u}/\lambda_0$, $V = q_0 - q$, $\tilde{A} = \hat{q}$ and $\tilde{V} = u - \lambda_0 q_0/\alpha$.

[2] The new auxiliary function is related to the former as $\tilde{F}_W(AV) = \tilde{F}_W(\hat{u}(q_0 - q)/\lambda_0) = F_{\rho_\lambda,\alpha}(q_0 - q, \hat{u}/\lambda_0)$.

The replica symmetric potential can furthermore be restated in terms of a scalar mutual information and conditional entropy

$$\phi(A,V,\tilde{A},\tilde{V}) = \frac{1}{2}\Big[-\tilde{A}V - \alpha A\tilde{V} + \tilde{F}_W(AV)\Big] + I(x; x+\frac{\xi_0}{\sqrt{A}}) + H(t_1|\tilde{\xi}_1;\tilde{V};\tilde{\rho})\,, \tag{5.10}$$

where ξ_0 and $\tilde{\xi}_1$ are standard Gaussian variables, $\tilde{\rho} = \lambda_0 q_0/\alpha$, x follows $p_x(x)$ and t_1 follows $p(t_1|\xi_1;V,q_0) = \int \mathcal{D}\tilde{\xi}\; p_{\rm out}(t_1|\sqrt{q_0-V}+\sqrt{V}\tilde{\xi})$.

Multi-layer We wish now to estimate the entropy at the next layer $H(\underline{t}^{(2)}|\underline{\underline{W}}^{(1)},\underline{\underline{W}}^{(2)})$. The replica potential for the reconstruction of $\underline{t}^{(1)}$ knowing $\underline{t}^{(2)}$ is according to the single-layer GLM formula

$$\phi_2(A_2,V_2,\tilde{A}_2,\tilde{V}_2) = \frac{1}{2}\Big[-\tilde{A}_2V_2 - \alpha_2 A_2\tilde{V}_2 + \tilde{F}_W(A_2V_2)\Big] + I(t_1; t_1+\frac{\xi_1}{\sqrt{A_2}}) + H(t_2|\tilde{\xi}_2;V_2;\tilde{\rho}_2)\,,$$

with ξ_1 and $\tilde{\xi}_2$ standard Gaussian variables, $\tilde{\rho}_1 = \lambda_1 q_1/\alpha_1$, $\alpha_1 = N_1/N_0$, $\lambda_1 = \mathbb{E}_{\lambda_{W^{(1)}}}[\lambda_{W^{(1)}}]$ and $q_1 = \int \mathrm{d}t_1\; p^{(1)}_{\rm out}(t_1)$. Using the identity $I(x;y) = H(x) - H(x|y)$ and the analytical expression of the entropy of a Gaussian distribution we have

$$I(t_1; t_1+\frac{\xi_1}{\sqrt{\tilde{A}_2}}) = H\big(t_1+\frac{\xi_1}{\sqrt{\tilde{A}_2}}\big) - H\big(\frac{\xi_1}{\sqrt{\tilde{A}_2}}\big) = H\big(t_1+\frac{\xi_1}{\sqrt{\tilde{A}_2}}\big) - \frac{1}{2}\log(2\pi e\tilde{A}_2^{-1}). \tag{5.11}$$

Yet the variable t_1 is itself the output of a GLM so that the entropy $H\big(t_1+\xi_1/\sqrt{\tilde{A}_2}\big)$ is again given by the single layer formula simply with the additional noise ξ_1. Applying this recursive heuristic argument leads to a replica formula with an additive potential over the layers,

$$\lim_{N_0\to\infty}\frac{1}{N_0}H(\underline{t}^{(\ell)}) = \min \operatorname{extr}_{\underline{A},\underline{V},\underline{\tilde{A}},\underline{\tilde{V}}}\phi_\ell(\underline{A},\underline{V},\underline{\tilde{A}},\underline{\tilde{V}}), \tag{5.12}$$

with four ℓ-dimensional vectors $\underline{A},\underline{V},\underline{\tilde{A}},\underline{\tilde{V}}$, and

$$\begin{aligned}\phi_\ell(\underline{A},\underline{V},\underline{\tilde{A}},\underline{\tilde{V}}) = {}& I\Big(t_0; t_0+\frac{\xi_0}{\sqrt{\tilde{A}_1}}\Big) - \frac{1}{2}\sum_{k=1}^{\ell}\tilde{\alpha}_{k-1}\big[\tilde{A}_kV_k + \alpha_kA_k\tilde{V}_k - F_{W^{(k)}}(A_kV_k)\big] \\ & + \sum_{k=1}^{\ell-1}\tilde{\alpha}_k\Big[H(t_k|\tilde{\xi}_k;\tilde{A}_{k+1},\tilde{V}_k,\tilde{\rho}_k) - \frac{1}{2}\log(2\pi e\tilde{A}_{k+1}^{-1})\Big] + \tilde{\alpha}_\ell H(t_\ell|\tilde{\xi}_\ell;\tilde{V}_\ell,\tilde{\rho}_\ell),\end{aligned} \tag{5.13}$$

where $\alpha_k = N_k/N_{k-1}$, $\tilde{\alpha}_k = N_k/N_0$, $\rho_k = \int \mathrm{d}t\; p^{(k-1)}_{\rm out}(t)\,t^2$, $\tilde{\rho}_k = \mathbb{E}_{\lambda_{W^{(k)}}}[\lambda_{W^{(k)}}]\rho_k/\alpha_k$, and $\tilde{\xi}_k$-s are standard Gaussian variables. In the computation of the conditional entropies in (5.13), the scalar t_k-variables are generated from $p(t_0) = p_x(t_0)$ and

$$p(t_k|\tilde{\xi}_k;\tilde{A},\tilde{V},\rho) = \mathbb{E}_{\tilde{\xi},\tilde{z}}\,p^{(k)}_{\rm out}(t_k+\xi/\sqrt{\tilde{A}}|\sqrt{\tilde{\rho}-\tilde{V}}\tilde{\xi}_k+\sqrt{\tilde{V}}\tilde{z}), \quad \forall k=1,\dots,\ell-1, \tag{5.14}$$

$$p(t_\ell|\tilde{\xi}_\ell;\tilde{V},\tilde{\rho}) = \mathbb{E}_{\tilde{z}}\,p^{(\ell)}_{\rm out}(t_\ell|\sqrt{\tilde{\rho}-\tilde{V}}\tilde{\xi}_\ell+\sqrt{\tilde{V}}\tilde{z}), \tag{5.15}$$

where ξ and $\tilde{z}$ are independent standard Gaussian random variables. Finally, the function $F_{W^{(k)}}(x)$ depends on the distribution of the eigenvalues of ${W^{(\ell)}}^\top W^{(\ell)}$ following the definition (5.9).

The heuristic procedure of derivation presented here gives an intuition for the result of the careful replica computation of [66] applied to the multi-layer GLM factor graph of $p(\underline{x}|\{\underline{\underline{W}}^{(k)}\},t^{(\ell)})$ represented on Figure 5.1b. The computation of the entropy of the output of a multi-layer network in the large dimensional limit, a computationally difficult task, has thus been reduced to an extremization of a function of 4ℓ variables, that requires evaluating single or bidimensional integrals. This extremization can be done efficiently by means of a fixed-point iteration starting from different initial conditions, as detailed in the Supplementary Material of [40] and supported in the dedicated package [83].

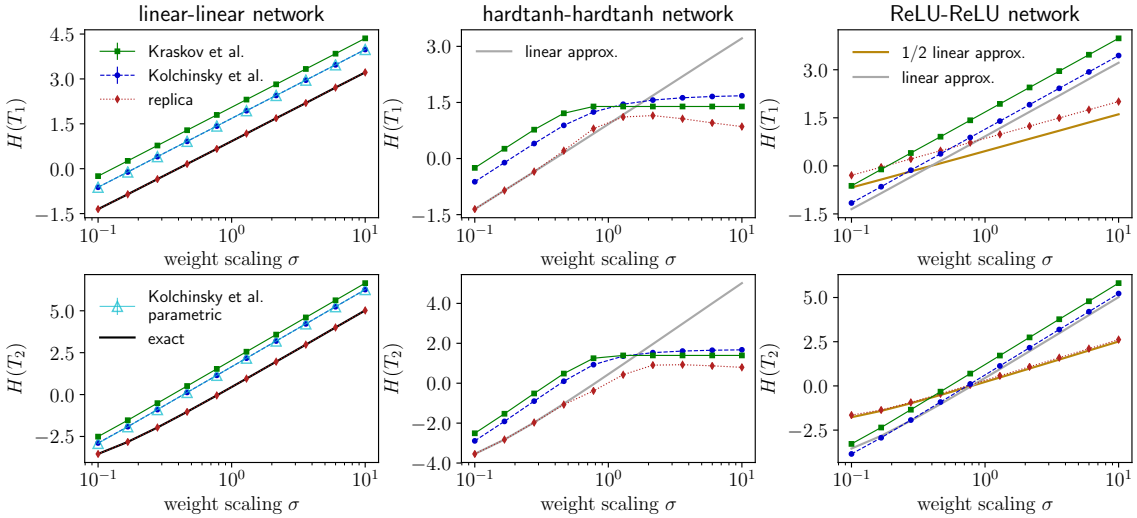


Figure 5.2: Entropy of latent variables in stochastic networks with equally sized layers $N_0 = N_1 = N_2 = 1000$, standard Gaussian inputs $p_x = \mathcal{N}(\underline{x}; 0, I_N)$, i.i.d weight entries drawn from $\mathcal{N}(W_{i\mu}; 0, \sigma/N)$, as a function of the weight scaling parameter σ. **Left column:** linear network. **Center column**: hardtanh-hardtanh network. **Right column:** relu-relu network.

Validity of the formula The formula (5.12) is conjecture to give an exact limit of the entropy density given weight matrices are independent samples of the orthogonally invariant ensemble. Co-authors in [40] rigorously proved the replica formula in 2-layer networks, for the restricted ensemble of weight matrices with Gaussian i.i.d entries. Although a general proof remains today out of reach, there are reasons to believe that the presented proof can be extended to an arbitrary number of layers, still in the case of Gaussian matrices. Further credentials to the result of the replica computation are coming from its agreement with different heuristics. The saddle point equations associated with the extremization of the replica potential (5.13) can be shown to recover fixed points of the ML-AMP [84] and ML-VAMP [38] algorithms for inference in multi-layer GLMs. In a related contribution, Reeves [118] also proposed a formula for the mutual information between adjacent layers (closely related to the entropy of the output) in deep networks, using different information-theoretic arguments. It similarly exhibits layer-wise additivity, and the two formulas are conjectured to be equivalent.

5.1.2 Comparison with non-parametric entropy estimators

The proposed strategy to approximate the entropy in multi-layer networks exploits the knowledge of the underlying model (input distribution, weight matrices, stochastic activations etc.) and produces an estimator of great accuracy for large neural networks.

The approach of non parametric estimation is entirely different. It relies only on a set of samples drawn from the distribution of interest to evaluate its entropy. The kernel-density approach of Kolchinsky et. al. [74] consists in fitting a mixture of Gaussians (MoG) to the samples and subsequently compute an upper bound on the entropy of the MoG [72]. The method of Kraskov et al. [76] uses nearest neighbor distances between samples to directly build an estimate of the entropy. Both methods require the computation of the matrix of distances between samples. It is unfortunately computationally hard to test how these estimators behave in high dimension as even for a known distribution the computation of the entropy is intractable in most cases. The replica method proposed here offers a valuable point of comparison for cases where it is rigorously exact.

We place ourselves in the setting of the rigorous Theorem of [40] to numerically test the different estimators (see Figure 5.2). The weight matrices entries are i.i.d. Gaussian with mean 0. Their standard-deviation is rescaled by a factor $1/\sqrt{N}$ and then multiplied by a coefficient σ varying between 0.1 and 10, i.e. around the recommended value for initialization in the training of deep neural networks. An additive white Gaussian noise of small variance ($\Delta = 10^{-5}$) is added right

before the activation function

$$\underline{t}^{(1)} = f(W_1 \underline{\underline{X}} + \underline{\epsilon}_1) \quad \text{with } \underline{\epsilon}_1 \sim \mathcal{N}(0, \Delta I_N), \tag{5.16}$$

$$\underline{t}^{(2)} = f(W_2 f(W_1 \underline{\underline{X}}) + \underline{\epsilon}_2) \quad \text{with } \underline{\epsilon}_2 \sim \mathcal{N}(0, \Delta I_N). \tag{5.17}$$

The non-parametric estimators [74, 76] were evaluated using 1000 samples, as the cost of computing pairwise distances is significant in high dimension. The entropy estimate is stable over independent draws of a sample of this size (error bars smaller than marker size). On Figure 5.2, we compare the different estimates of $H(\underline{t}^{(1)})$ and $H(\underline{t}^{(2)})$ for different activation functions: linear, hardtanh or relu. The hardtanh activation is a piecewise linear approximation of the tanh, $\text{hardtanh}(x) = -1$ for $x < -1$, x for $-1 < x < 1$, and 1 for $x > 1$, for which the integrals in the replica formula can be evaluated faster than for the tanh.

In the linear and hardtanh case, the non-parametric methods are following the tendency of the replica estimate when σ is varied, but appear to systematically over-estimate the entropy. For linear networks with Gaussian inputs and additive Gaussian noise, every layer is also a multivariate Gaussian and therefore entropies can be directly computed in closed form (*exact* in the plot legend). When using the Kolchinsky estimate in the linear case we also check the consistency of two strategies, either fitting the MoG to the noisy sample or fitting the MoG to the deterministic part of the $\underline{t}^{(\ell)}$ and augment the resulting variance with Δ, as done in [74] (*Kolchinsky et al. parametric* in the plot legend). In the network with hardtanh non-linearities, we check that for small weight values, the entropies are the same as in a linear network with same weights (*linear approx* in the plot legend, computed using the exact analytical result for linear networks and therefore plotted in a similar color to *exact*). Lastly, in the case of the relu-relu network, we note that non-parametric methods are predicting an entropy increasing as the one of a linear network with identical weights, whereas the replica computation reflects its knowledge of the cut-off and accurately features a slope equal to half of the linear network entropy (*1/2 linear approx* in the plot legend). While non-parametric estimators are powerful tools able to approximate entropies from the mere knowledge of samples, they inevitably introduce estimation errors. The replica method is taking the opposite view. While being restricted to a class of models, it can leverage its knowledge of the neural network structure to provide a reliable estimate. To our knowledge, there is no other entropy estimator able to incorporate this information about the underlying multi-layer model.

5.2 Mean-field information trajectories over training of deep networks

The information bottleneck theory [146] applied to neural networks consists in computing the mutual information between the data and the hidden learned representations on the one hand, and between labels and again hidden learned representations on the other hand [147, 137]. A successful training will maximize the information with respect to the labels it learns to predict. Tishby and collaborators argue that it should simultaneously minimize the information with respect to the input data. The idea is that by getting rid of unecessary information from the input to predict the output, deep neural networks would prevent overfitting and lead to a good generalization. While this intuition suggests new learning algorithms and regularizers [23, 2, 4, 1, 74, 12, 162], we can also hypothesize that this mechanism is already at play in a priori unrelated commonly used optimization methods, such as the simple stochastic gradient descent (SGD), which already yield very good capacities of generalization. This fact was first claimed by [137]. In this work, numerical experiments were run on very small neural networks, to allow for entropy estimation by binning of the hidden neurons activities. Afterwards, the authors of [128] reproduced the results of [137] on small networks using the continuous entropy estimator of [74], but found that the overall behavior of mutual information during learning is greatly affected when changing the nature of non-linearities. Additionally, these authors investigate the training of larger linear networks on i.i.d. normally distributed inputs where entropies at each hidden layer can be computed analytically for an additive Gaussian noise. The replica estimator we derived allows us to evaluate entropies and mutual informations in non-linear networks larger than in [128, 137], provided that the training data has a distribution belonging to a certain class and that weight matrices verify the assumptions of the formula.

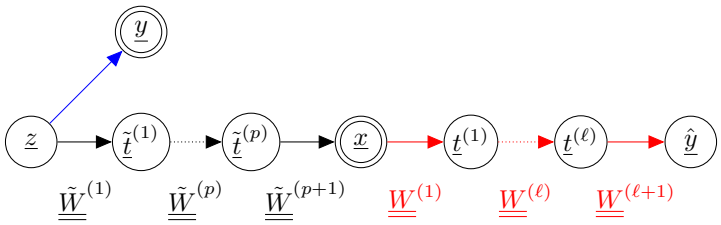


Figure 5.3: Graphical representation of a teacher model generating a synthetic data set of input-output pairs $\{\underline{x}, \underline{y}\}$ and a student model with learnable parameters trained to recover the input-output mapping.

5.2.1 Tractable deep learning models

Revisited teacher-student scenario The multi-layer model presented above can be leveraged to simulate a prototypical setting of deep supervised learning on synthetic data sets amenable to the replica computation of entropies and mutual informations. We consider a teacher model generating input-output pairs $\{\underline{x}, \underline{y}\}$ from a vector latent variable $\underline{z}$ with separable prior distribution $p_z(\underline{z}) = \prod_{i=1}^{N_0} p_z(z_i)$. The rule generating the target outputs $\underline{y}$ can be chosen arbitrarily (in blue on the Figure 5.3). However we impose that the model transforming $\underline{z}$ into $\underline{x}$ is a multi-layer neural networks with orthogonally invariant weight matrices (in black). This teacher model can be sampled from to generate a training data set. Subsequently, a student deep neural network (in red) is trained on the synthetic data set, under the weight constraint described below, to predict at its output $\underline{\hat{y}}$ the target $\underline{y}$ from the input $\underline{x}$.

A more traditional teacher-student scenario would consist in a matched setting; where the teacher would be itself a feed forward neural network from the input $\underline{x}$, following a simple prior distribution $p_x(\underline{x})$, to the target output $\underline{y}$. The advantage of the proposed setting is that it allows to consider synthetic inputs $\underline{x}$ featuring some structure. In fact the *generative network* from $\underline{z}$ to $\underline{x}$ is corresponds to the common architecture of VAEs [71, 120] and GANs [50]. Via unsupervised learning adjusting the weights $\{\tilde{W}^{(1)}, \cdots, \tilde{W}^{(p)}\}$ these generative models can generate samples mimicking real images.

Weight constraint At the start of a neural network training, weight matrices initialized as i.i.d. Gaussian random matrices satisfy the necessary assumptions of the replica formula. In their singular value decomposition $W^{(\ell)} = U^{(\ell)} S^{(\ell)} {V^{(\ell)}}^\top$ the matrices $U^{(\ell)} \in \mathrm{O}(N_\ell)$ and $V^{(\ell)} \in \mathrm{O}(N_{\ell-1})$, are typical independent samples from the Haar measure across all layers. To make sure weight matrices remain close enough to this assumption during learning, we define a custom weight constraint which consists in keeping $U^{(\ell)}$ and $V^{(\ell)}$ fixed while only the matrix $S^{(\ell)}$, constrained to be diagonal, is updated. The number of parameters is thus reduced from $N_\ell \times N_{\ell-1}$ to $\min(N_\ell, N_{\ell-1})$. We refer to layers following this weight constraint as USV-layers.

The relatively small number of degrees of freedom implies that USV-layers are harder to train than usual. In practice, we notice that they tend to require more parameter updates. Also, interleaving linear USV-layers to increase the number of parameters between non-linearities can significantly improve the final result of training. We conduct an experiment on the MNIST data set. The best train and test, losses and accuracies, for the different architectures are given in Table 5.1 and some learning curves are displayed on Figure 5.4. As expected we observe that USV-layers are achieving better classification success than the random projections, yet worse than the unconstrained fully connected layer. Stacking USV-layers to increase the number of trainable parameters allows to reach very good training accuracies, nevertheless, the testing accuracies do not benefit to the same extent from these additional parameters. On Figure 5.4, we observe that the version of the experiment with many (6) USV-layers overfits the training set (green curves with testing losses growing towards the end of learning). Adding explicit regularizers might allow to improve the generalization performances of models with USV-layers. This experiment confirms that USV-layers can learn to represent complex functions despite their restriction.

We further note that such a product decomposition is reminiscent of a series of works on

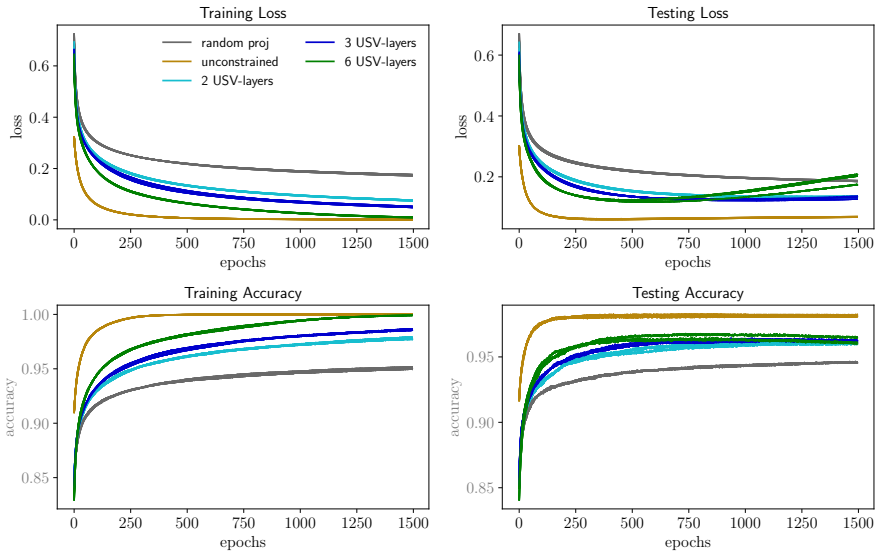


Figure 5.4: Training and testing curves for the training of a two-layer neural net on the classification of MNIST (60 000 training images and 10 000 testing images), featuring one non-linear (relu) hidden layer of 500 neurones and a softmax output layer. The 500×10 parameters of the weight matrix of the output layer are all being learned in all the versions of the experiments. Conversely, before the relu layer, we either (1) do not learn at all the 784×500 parameters which then define random projections, (2) learn all of them as a traditional fully connected network, (3) use a combination of 2 (3a), 3 (3b) or 6 (3c) consecutive USV-layers (without any intermediate non-linearity). For each version of the experiment the outcomes of two independent runs are plotted with the same color, it is not always possible to distinguish the two runs as they overlap.

First layer type	#train params	Train loss	Test loss	Train acc	Test acc
Random (1)	0	0.1745	0.1860	95.05 (0.09)	94.61 (0.02)
Unconstrained (2)	784×500	0.0012	0.0605	100. (0.00)	98.18 (0.06)
2-USV (3a)	2×500	0.0758	0.1326	97.80 (0.07)	96.10 (0.03)
3-USV (3b)	3×500	0.0501	0.1238	98.62 (0.05)	96.35 (0.04)
6-USV (3c)	6×500	0.0092	0.1211	99.93 (0.01)	96.54 (0.17)

Table 5.1: Training results for MNIST classification of a fully connected 784-500-10 neural net with a ReLU non linearity. The different rows correspond to different specifications of trainable parameters in the first layer (1, 2, 3a, 3b, 3c) describe in the caption of Figure 5.4. We use plain SGD to minimize the cross-entropy loss. All experiments use the same learning rate 0.01 and batchsize of 100 samples. Results are averaged over 5 independent runs, and standard deviations are reported in parentheses.

adaptive structured efficient linear layers (SELLs and ACDC) [94, 158] motivated this time by speed gains. In these layers only diagonal matrices are learned and the matrices $U^{(\ell)}$ and $V^{(\ell)}$ are permutations of Fourier or Hadamard matrices, so that the matrix multiplication can be replaced by fast transforms.

Following information during training Combining the teacher-student approach with the learning of USV-layer in the student, we are able to evaluate the entropy $H(\underline{t}^{(k)})$ at any layer of the student model during training. Indeed the Markov Chain starting at the latent variable $\underline{z}$, running through the teacher generative model and the student network (see Figure 5.3) fulfills the assumptions of the multi-layer model defined in Section 5.1.1. In particular, the initial distribution $p_z(\underline{z})$ is separable.

The mutual information between two continuous random variables x and y with joint density

$p_{x,y}(x,y)$ and marginals $p_x(x)$ and $p_y(y)$ is defined by

$$I(x;y) = \int dxdy p_{x,y}(x,y) \log \frac{p_{x,y}(x,y)}{p_x(x)p_y(y)}. \tag{5.18}$$

It quantifies the level of dependence of its two arguments. It is always positive and equal to zero if and only x and y are independent. The above definition can be considered as a generalization of the formula for discrete random variables following the definition of the differential entropy. An equivalent expression is $I(x;y) = H(y) - H(y|x)$. The extension to continuous random variable may however be ill-defined as the previous integral can diverge.

To access mutual informations in deep neural networks, from the replica formula for entropies, we need to compute the conditional entropies between adjacent layers. Considering neural networks of large width, we use the central limit theorem to reduce their evaluation to a 2-dimensional integral

$$H(\underline{t}^{(k)}|\underline{t}^{(k-1)}) = \int_{\mathbb{R}^{N_k}} \mathrm{d}\underline{t} \int_{\mathbb{R}^{N_{k-1}}} \mathrm{d}\underline{t}' \; p_{\text{out}}\left(\underline{t}|\underline{\underline{W}}\underline{t}'\right) p^{(k-1)}(\underline{t}') \log p_{\text{out}}(\underline{t}|\underline{\underline{W}}\underline{t}') \tag{5.19}$$

$$= N_k \int \mathrm{d}z \, \mathcal{N}(z;0,\tilde{\rho}^{(k)}) \int \mathrm{d}h \; p_{\text{out}}(h|z), \tag{5.20}$$

from which we deduce $I(\underline{t}^{(k)};\underline{t}^{(k-1)}) = H(\underline{t}^{(k)}) - H(\underline{t}^{(k)}|\underline{t}^{(k-1)})$.

While we have defined our student model as a priori stochastic, traditional feed forward neural networks are deterministic. In the numerical experiments presented in the next Section, we train and test networks without injecting noise, and only assume a noise model in the computation of information-theoretic quantities. We choose to add a white Gaussian noise of small variance ($\Delta = 10^{-5}$) right before the last activation function of the Markov Chain. This addition is necessary for conditional entropies, and consequently mutual informations, to remain finite. Our choice attempts to stay as close as possible to deterministic neural networks, yet remains inevitably somewhat arbitrary. This issue will be discussed in Section 5.3. Note that for the computation of $H(\underline{t}^{(k)})$ the mapping between $\underline{x}$ and $\underline{t}^{(k-1)}$ is kept deterministic, which allows to follow the mutual information with the input layer given the identity

$$I(\underline{t}^{(k)};\underline{x}) = H(\underline{t}^{(k)}) - H(\underline{t}^{(k)}|\underline{x}) = H(\underline{t}^{(k)}) - H(\underline{t}^{(k)}|\underline{t}^{(k-1)}) = I(\underline{t}^{(k)};\underline{t}^{(k-1)}). \tag{5.21}$$

5.2.2 Training experiments

Using a dedicated python package [41], interfacing the package [83] for replica computations, we run training experiments on different instances of the deep learning models defined above. We seek to study the simplest possible training strategies achieving good generalization. Hence for all experiments we use plain stochastic gradient descent (SGD) with constant learning rates, without momentum and without any explicit form of regularization. The sizes of the training and testing sets are taken equal and scale typically as a few hundreds times the size of the input layer.

Linear trainings

In our first experiment we check on linear networks a potential caveat of our strartegy. For the replica formula to be correct, additionally to the restriction to USV-layers, the different matrices $S^{(\ell)}$ should remain sufficiently uncorrelated during the learning. By focusing on linear networks trained on Gaussian i.i.d input data, the replica prediction can be compared to a closed form expression. We consider the training of a 4-layer linear network, on a synthetic data set of Gaussian inputs and outputs generated by a linear teacher (see Figure 5.5 for details). We repeat the training experiment for increasing layer width N, and report the maximum and the mean value of the squared error on the estimation of the $I(\underline{x};\underline{t}^{(\ell)})$ over all epochs of 5 independent training runs. We find that even if errors tend to increase with the number of layers, they remain objectively very small and decrease drastically as the size of the layers increases. Results therefore demonstrate (i) that the replica formula remains correct throughout the learning of the USV-layers and (ii) that the replica method gets closer and closer to the exact result in the limit of large networks, as

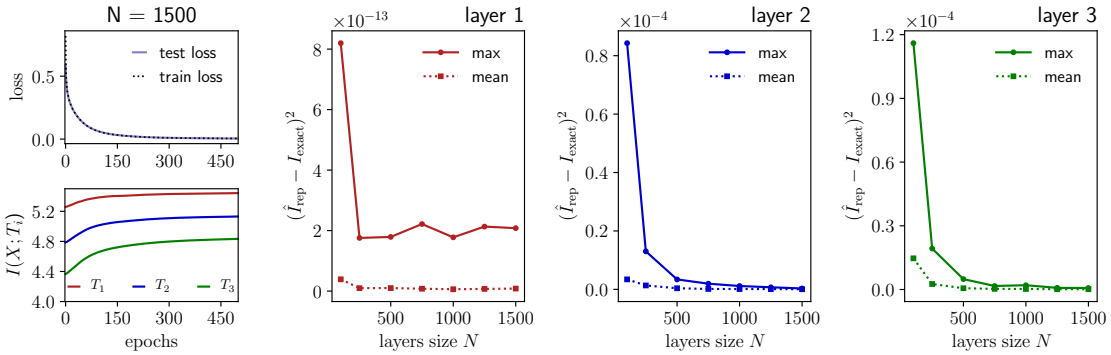


Figure 5.5: Training of a 4-layer linear students of varying size on a regression task generated by a linear teacher. The teacher of output$N_y = 4$ is $\underline{y} = \tilde{W}\underline{x} + \underline{\epsilon}$, with input $\underline{x} \sim \mathcal{N}(\underline{x}; 0, I_N)$ of size N. The random teacher matrix $\tilde{W}$ has i.i.d. normally distributed entries of mean 0 and variance $1/N$. The noise$\underline{\epsilon}$ follow s$\mathcal{N}(\underline{\epsilon}; 0, 0.01 I_N)$. The student network has 3 USV-layers, plus one fully connected unconstrained layer $\underline{x} \to \underline{t}^{(1)} \to \underline{t}^{(2)} \to \underline{t}^{(3)} \to \underline{\hat{y}}$. The training on the regression task is done using plain SGD for the MSE loss $(\underline{\hat{y}} - \underline{y})^2$. **Upper-left:** MSE loss on the training and testing sets during training by plain SGD for layers of size $N = 1500$. Best training loss is 0.004735, best testing loss is 0.004789. **Lower-left:** Corresponding mutual information evolution between hidden layers and input. **Center-left, center-right, right:** maximum and squared error of the replica estimation of the mutual information as a function of layers size N, over the course of 5 independent trainings for each value of N for the first, second and third hidden layer.

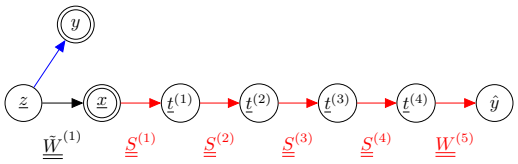


Figure 5.6: Graphical model representation of the teacher and student model for the learning experiments of non-linear networks, corresponding to results presented on Figure 5.7 and Figure 5.8. The simple generative model is $\underline{x} = \underline{\underline{\tilde{W}}}\underline{z} + \underline{\epsilon}$ with normally distributed latent representation $\underline{z} \sim \mathcal{N}(0, I_{N_0})$ of size $N_0 = 100$, data $\underline{x}$ of size $N_1 = 500$ generated with matrix $\underline{\underline{\tilde{W}}}_{\rm gen}$ with i.i.d. normally distributed entries of variance as $1/N_0$ and noise $\underline{\epsilon} \sim \mathcal{N}(\underline{\epsilon}; \underline{0}, 0.01 I_{N_1})$. We then train a student to solve the binary classification problem of recovering the label $y = \text{sign}(z_1)$ (in blue). Note that the rest of the initial code $(z_2, ..z_{N_0})$ acts as noise/nuisance with respect to the learning task. The dimensions of the student layers (in red) are 500-1000-500-250-100-2, with 4-USV layers with only matrices $S^{(l)}$ as trainable parameters and one unconstrained fully-connected layer before the output.

theoretically predicted (5.12). In the following, we assume that it is also the case for non-linear networks.

Concerning the application of the information-bottleneck ideas to deep learning, a similar experiment of linear network training was studied in [128]. In agreement with their observations, we find that the mutual informations $I(\underline{x}; \underline{t}^{(\ell)})$ keep on increasing throughout the learning (bottom left pannel of Figure 5.5), without compromising the generalization ability of the student.

Non linear trainings

Finally, we apply the replica formula to estimate mutual informations during the training of non-linear networks on correlated input data. We design a synthetic problem of classification described on Figure 5.6. In the following we vary activation functions and weight initializations for the student of fixed architecture. For training, we use plain SGD but this time to minimize the cross-entropy loss[1]. We start with the comparison of two students with activations either linear-relu-linear-

[1]The cross-entropy loss has been found to be more effective than the mean square error in classification tasks. For a problem with M categories, the training outputs are encoded in vectors $\underline{y} \in \{0, 1\}^M$ with only one non-zero

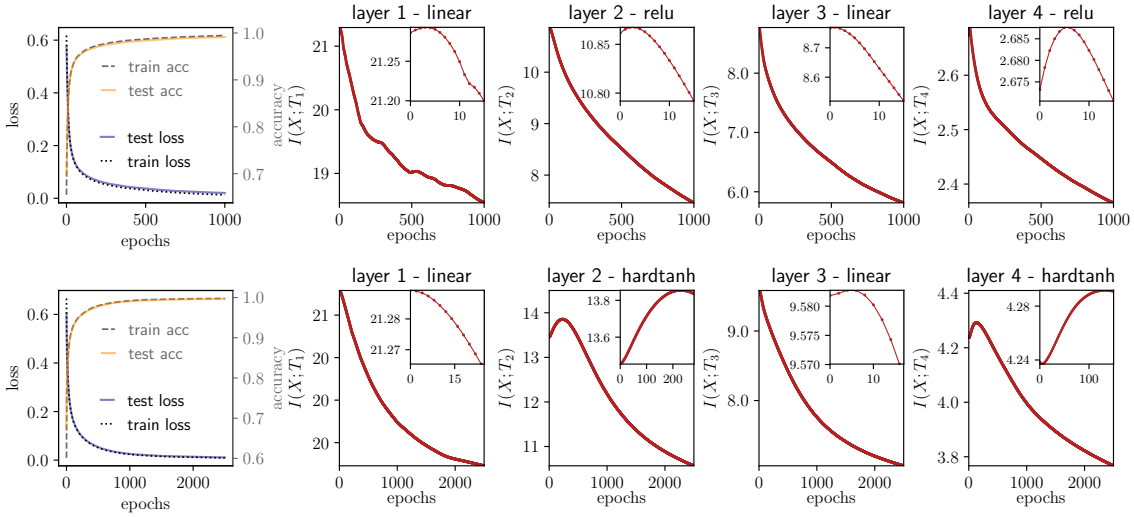


Figure 5.7: Training of two student models on a binary classification task with correlated input data and either relu (top) or hardtanh (bottom) non-linearities. **Left:** training and generalization cross-entropy loss (left axis) and accuracies (right axis) during learning. Best training-testing accuracies are 0.995 - 0.991 for relu version (top row) and 0.998 - 0.996 for hardtanh version (bottom row). **Remaining columns:** mutual information between the input and successive hidden layers. Insets zoom on the first epochs.

relu-softmax[2] (top row of Figure 5.7) or linear-hardtanh-linear-hardtanh-softmax (bottom row). Because USV-layers only feature $O(N)$ parameters instead of $O(N^2)$ we observe that they require more iterations to train in general. In the case of the relu network, adding interleaved linear layers was key to successful training with 2 non-linearities, which explains the somewhat unusual architecture proposed. For the student model using hardtanh, this was actually not an issue (see the following experiment), however, we consider a similar architecture for fair comparison. This experiment is reminiscent of the setting of [137], yet now tractable for networks of larger sizes. For both types of non-linearities we observe that the mutual information between the input and all hidden layers decrease during the learning, except for the very beginning of training where we can sometimes observe a short phase of increase (see zoom in insets). For the hardtanh layers this phase is longer and the initial increase of noticeable amplitude. In this particular experiment, the claim of [137] that compression can occur during training even with non double-saturated activation seems corroborated (a phenomenon that was not observed by [128]). Yet we do not observe that the compression is more pronounced in deeper layers and its link to generalization remains elusive. For instance, we do not see a delay in the generalization w.r.t. training accuracy/loss in the student model with hardtanh despite of an initial phase without compression in two layers.

Futhermore, we find that changing the weight initialization can drastically change the behavior of mutual informations during training while resulting in identical training and testing final performances. In an additional experiment, we consider a similar setting to the classification on correlated data presented above. On Figure 5.8 we compare 3 identical students with activations hardtanh-hardtanh-hardtanh-hardtanh-softmax. Initial weight entries are sampled from a zero-mean Gaussian with variance $4/N_{\ell-1}$ (for the model presented at the top row), $1/N_{\ell-1}$ (middle row), and $0.25/N_{\ell-1}$ (bottom row). The first column shows that training is delayed for the weight initialized at smaller values, but eventually catches up and reaches accuracies superior to 0.97 both

entry at the position of the target class. The trained neural network has an ouput $\underline{\hat{y}} \in [0,1]^M$, with components summing to one, interpreted as a probability distribution over the classes. The cross-entropy loss is then defined as

$$\ell(\underline{y}, \underline{\hat{y}}) = -\sum_{\mu=1}^{M} y_\mu \log \hat{y}_\mu .$$

[2]The softmax activation precisely allows the interpretation of the output as a probability vector. For a pre-activation vector $\underline{z} \in \mathbb{R}^M$, $\underline{\hat{y}} = \text{softmax}(\underline{z}) = \mathrm{e}^{\underline{z}} / \sum_{\mu=1}^{\mathrm{M}} \mathrm{e}^{z_\mu}$.

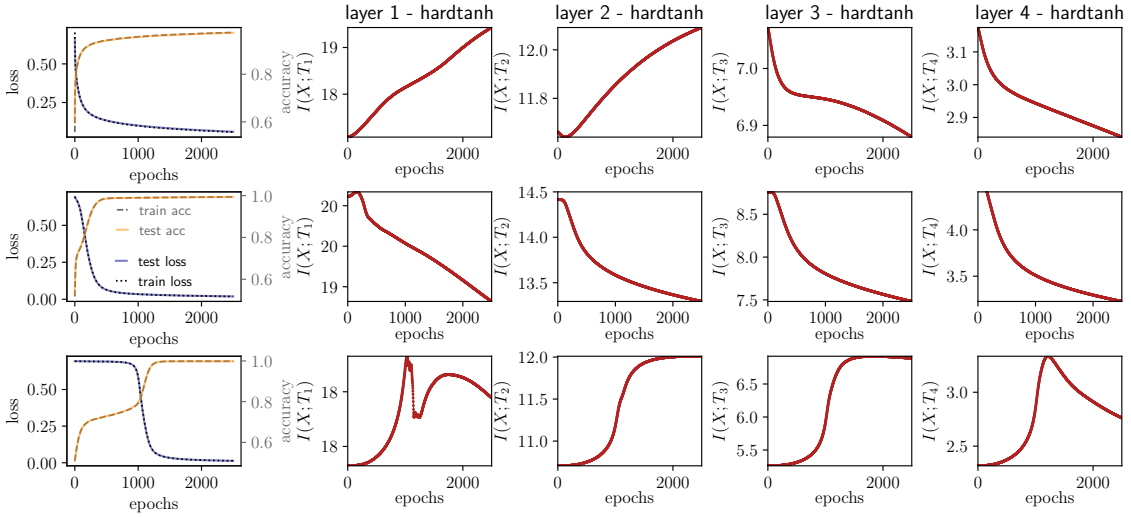


Figure 5.8: Learning and hidden-layers mutual information curves for a classification problem with a student using hardtanh activations. **Top:** Initial weights at layer ℓ of variance $4/N_{\ell-1}$, best training accuracy 0.999, best test accuracy 0.994. **Middle:** Initial weights at layer ℓ of variance $1/N_{\ell-1}$, best train accuracy 0.994, best test accuracy 0.9937. **Bottom:** Initial weights at layer ℓ of variance $0.25/N_{\ell-1}$, best train accuracy 0.975, best test accuracy 0.974. The overall direction of evolution of the mutual information can be flipped by a change in weight initialization without changing drastically final performance in the classification task.

in training and testing. Meanwhile, mutual informations have different initial values for the different weight initializations and follow very different paths. They either decrease during the entire learning, or on the contrary are only increasing, or actually feature an hybrid path. We further note that it is to some extent surprising that the mutual information would increase at all in the first row if we expect the hardtanh saturation to instead induce compression.

These observed differences and non-trivial observations raise numerous questions, and suggest that within the examined setting, a simple information theory of deep learning remains out-of-reach. In the next Section, we will discuss more thoroughly hurdles in the success of this strategy and in particular the necessity of a noise regularization of the mutual information. To conclude this Section, we check that qualitative behaviors discussed above are robust to a modification of the amplitude of the injected noise a posteriori. In all the previous experiments we had added a Gaussian noise of variance $\Delta = 10^{-5}$ to regularize the computation of the mutual information. Here we verify on still the same problem described on Figure 5.6, with hardtanh activations, that going to smaller levels of noise does not change the qualitative picture. On Figure 5.9, we compare the mutual information evolution during learning while varying Δ. As Δ decreases we observe that the average level of mutual information is increasing as expected (it should diverge at zero noise), yet the relative evolution is qualitatively unchanged.

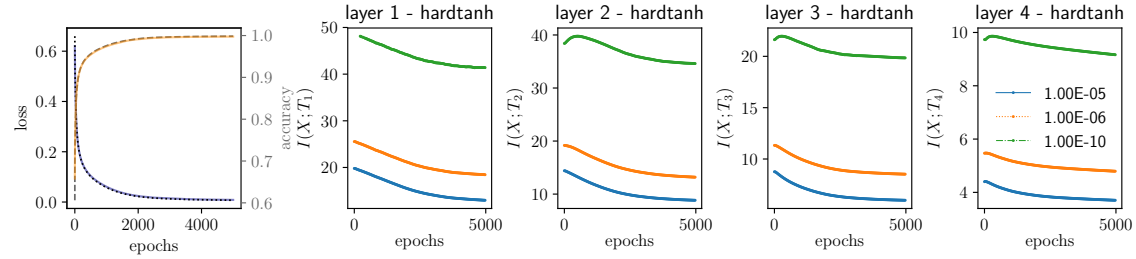


Figure 5.9: Learning and hidden-layers mutual information curves for a classification problem with correlated input data, using a 4-USV hardtanh layers and 1 unconstrained softmax layer for different levels of regularizing noise: $\Delta = 10^{-5}$ (blue), $\Delta = 10^{-6}$ (orange), $\Delta = 10^{-10}$ (green).

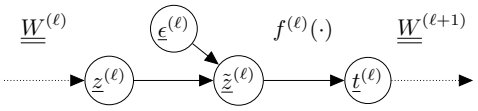


Figure 5.10: Representation as graphical model of the assumed noise injection at each layer of the stochastic neural networks studied in Section 5.3.1.

5.3 Further investigation of the information theoretic analysis of deep learning

While a few experimental works claimed to have verified that SGD training of deep neural networks leads to compression of the information in hidden layers [137, 102], others have nonetheless pointed out caveats in the application of these information theoretic ideas to the traditional supervised deep learning setting [128, 48, 73, 40]. As already mentioned, the key issue lies in the definition of the mutual information in deterministic neural networks.

In a feed forward deep neural network used for supervised learning the mapping between the entry $\underline{x}$ and the ℓ-th hidden layer $\underline{t}^{(\ell)}$ is typically deterministic. Consequently the conditional density $p(\underline{t}^{(\ell)}|\underline{x})$ is a Dirac with entropy $H(\underline{t}^{(\ell)}|\underline{x})$ diverging to $-\infty$ and, as noted by [128, 48], $I(\underline{x};\underline{t}^{(\ell)})$ is effectively $+\infty$. An intuitive way to attempt to regularize this mutual information is to quantize the continuous random variables and resort to the definition in the discrete case, which is always well defined. This is the strategy adopted by [137]. Nevertheless [128] showed that for the same training run a compression phase could be either present or absent when using different conventions for partitioning continuous intervals. Hence the regularization of the mutual information a posteriori of learning does not yield robust observations.

Conversely we can examine the training of neural networks for which mutual information are directly well defined. While discrete activations are problematic to study gradient based training algorithms, the addition of noise in a continuous neural network during the learning phase is a common form of regularization to promote generalization that also avoids the divergence of mutual information.

5.3.1 Mutual information in noisy trainings

We consider the addition of noise in the layers of neural networks during training aiming at regularization. The intuition behind this strategy is to force the learning towards a solution that is robust to variations around the training samples and therefore will also be effective on unseen test samples. Dropout [59, 142] and batch-normalization [64] are two specific protocols of noise injection during training that lead to significant improvements of performances of deep learning models and are widely adopted by practitioners. They involve both multiplicative and additive noises. We imagine a small experiment with different stochastic deep neural networks and a non-parametric estimator of mutual information. Resorting to the latter, rather than the replica formula considered in Section 5.1, allows to relax the restrictions to synthetic data sets and USV-layers, yet we will see at the cost of a loss in sensitivity. We describe how the estimator of [72] can be used to compute information observables for a variety of noises, before examining numerical results on the MNIST data set.

Non-parametric estimation

We assume that noise is added at each layer to the pre-activations denoted $\underline{z}^{(\ell)}$, in the form of a standard Gaussian white noise $\underline{\epsilon}^{(\ell)}$ scaled by a function of $\underline{z}^{(\ell)}$. Formally we have the recursion

$$\underline{z}^{(\ell)} = \underline{\underline{W}}^{(\ell)}\underline{t}^{(\ell-1)}\,, \quad \tilde{\underline{z}}^{(\ell)} = \underline{z}^{(\ell)} + n(\underline{z}^{(\ell)})\underline{\epsilon}^{(\ell)}\,, \quad \underline{t}^{(\ell)} = f(\tilde{\underline{z}}^{(\ell)}), \tag{5.22}$$

where the function $n(\cdot)$ fixing the amplitude of the noise and the activation $f(\cdot)$ are applied component-wise (see Figure 5.10).

We can compute an upper and lower bound of $I(\underline{x}, \tilde{\underline{z}}_{(\ell)})$ following [72, 128] from empirical samples. The injected noise need not be additive Gaussian and these bounds can be derived for an arbitrary noise scaling function. We consider a set of P training examples $\{\underline{x}_{(k)}\}_{k=1}^{P}$, that are

transformed by the neural network up to the layer ℓ into $z^{(\ell)}_{(k)}$. The distribution of the following noised pre-activation $\underline{z}^{(\ell)}$ can be interpreted as a mixture of distributions with centroids $\{\underline{z}^{(\ell)}_{(k)}\}_{k=1}^{P}$,

$$p_{\tilde{z}^{(\ell)}}(\tilde{z}^{(\ell)}) = \frac{1}{P}\sum_{k=1}^{P} p\left(\underline{\tilde{z}}^{(\ell)}|\underline{z}^{(\ell)}_{(k)}\right) = \frac{1}{P}\sum_{k=1}^{P}\prod_{i=1}^{N_\ell}\mathcal{N}\left(\tilde{z}^{(\ell)}_i;\, z^{(\ell)}_{k,i}, n(z^{(\ell)}{}_{k,i})^2\right). \tag{5.23}$$

Following [128] we therefore use the mutual information estimator for mixtures of [72],

$$\hat{I}(\underline{z}^{(\ell)};\underline{\tilde{z}}^{(\ell)}) = -\sum_{k=1}^{P}\frac{1}{P}\log\sum_{k'=1}^{P}\frac{1}{P}\exp\left(-D\left(p(\underline{\tilde{z}}^{(\ell)}|\underline{z}^{(\ell)}_{(k)})\,||\,p(\underline{\tilde{z}}^{(\ell)}|\underline{z}^{(\ell)}_{(k)})\right)\right), \tag{5.24}$$

where D is a generalized distance between probability densities. We further use the results of [72] showing that the estimator is an upper bound when using as a distance the Kullback-Leibler divergence, and is a lower bound for the Chernoff α-divergence. Plugging-in our notations and the definitions of distances we obtain the following expressions, for the upper bound

$$\hat{I}_{\text{up}}(\underline{z}^{(\ell)};\underline{\tilde{z}}^{(\ell)}) = -\sum_{k=1}^{P}\frac{1}{P}\log\sum_{k'=1}^{P}\exp\left(-\left[\sum_{i=1}^{N_\ell}\log\left|\frac{n(z^{(\ell)}_{k,i})}{n(z^{(\ell)}_{k',i})}\right| + \frac{1}{2}\left(\frac{n(z^{(\ell)}_{k,i})^2}{n(z^{(\ell)}_{k',i})^2} - 1\right) + \frac{(z^{(\ell)}_{k,i} - z^{(\ell)}_{k',i})^2}{2n(z^{(\ell)}_{k',i})^2}\right]\right), \tag{5.25}$$

and for the lower bound

$$\hat{I}_{\text{low}}(\underline{z}^{(\ell)};\underline{\tilde{z}}^{(\ell)}) = -\sum_{k=1}^{P}\frac{1}{K}\log\sum_{k'=1}^{P}\exp\left(-\left[\sum_{i=1}^{N_\ell}\frac{1}{4}\frac{(z^{(\ell)}_{k,i} - z^{(\ell)}_{k',i})^2}{n(z^{(\ell)}_{k',i})^2 + n(z^{(\ell)}_{k',i})^2} + \frac{1}{2}\log\frac{1}{2}\frac{n(z^{(\ell)}_{k',i})^2 + n(z^{(\ell)}_{k',i})^2}{|n(z^{(\ell)}_{k',i})n(z^{(\ell)}_{k',i})|}\right]\right) \tag{5.26}$$

where we considered the Chernoff 1/2-divergence for the lower bound.

We will only consider noises with zero mean so that, along the feed forward pass, the successive linear mixing of the products with weight matrices approximately makes $\underline{z}^{(\ell)}$ a deterministic function of $\underline{x}$ (that is the transformation of $\underline{x}$ by the neural network without the noise injection). Hence we expect, following the same argument as (5.21), that $I(\underline{\tilde{z}}^{(\ell)};\underline{z}^{(\ell)}) \simeq I(\underline{\tilde{z}}^{(\ell)};\underline{x})$. Beyond this heuristic argument, the data processing inequality [29] ensures that $I(\underline{\tilde{z}}^{(\ell)};\underline{z}^{(\ell)}) \geq I(\underline{\tilde{z}}^{(\ell)};\underline{x})$, so that a drop of the first mutual information is reflected in the second if they are of comparable order of magnitude.

Thus we have a strategy to estimate mutual informations between layers from a set of P samples. This allows to consider 'real' data sets, of natural images for instance, and arbitrary architectures and training algorithm. However, the non-parametric strategy will suffer from two limitations. First, its approximation of the density as a mixture over the P samples limits its sensitivity to large mutual informations

$$\hat{I}(\underline{z}^{(\ell)};\underline{\tilde{z}}^{(\ell)}) \leq H(\underline{z}^{(\ell)}) = -\sum_{k=1}^{P} p\left(\underline{z}^{(\ell)}_{(k)}\right)\log p\left(\underline{z}^{(\ell)}_{(k)}\right) = -\sum_{k=1}^{P}\frac{1}{P}\log\frac{1}{P} = \log P. \tag{5.27}$$

The bound is saturated when the noised representations $\underline{\tilde{z}}^{(\ell)}$ of the different samples are well separated, as also noted by [48]. Otherwise the estimator (5.24) examines the overlaps of the mixture components from the pairwise distances to evaluate the mutual information. This points at the second limitation of the procedure: the memory (and computational) burden in P^2. While a large number of samples P is desirable to increase the sensitivity of the estimator upper-bounded by $\log P$, the cost scales quadratically. This issue worsens as the dimension of variables increase as overlaps are typically vanishing.

Numerical experiments on MNIST

We run experiments training deep neural networks to solve the benchmark task of classifying the handwritten digits of the MNIST data set. We adopt a unique architecture with 5 hidden layers and 20 fully connected neurons in all hidden layers, with either tanh or relu activations, and 10 output softmax neurons to predict the 10 digit classes. The trainings are done with mini-batch stochastic gradient descent (batch-size of 100 and learning rate of 0.01) without other forms of regularization besides the noise injection under study.

We consider both additive and multiplicative noises via different noise scaling functions: either constant $n(z_i) = \sigma_0$, linear $n(z_i) = \sigma_0|z_i|$, quadratic $n(z_i) = \sigma_0 z_i^2$, or exponential $n(z_i) = \sigma_0(e^{|z_i|} - 1)$. We select the optimal value for the parameter σ_0 in terms of the best accuracy achieved on the MNIST test data set, and report the values of the upper bound and lower bounds (5.25)-(5.26) for the last hidden layer during the training on Figure 5.11 for tanh activations and Figure 5.12 for relu activations. For stochastic deep networks with tanh activation, we observe a clear compression during the learning for multiplicative noises, and a slight compression in a second phase of training for the constant noise. Unfortunately, for the unbounded relu activation the estimator hits the limitation of the non-parametric approach induced by the size of the sample $P = 5000$. Increasing the sample size comes with costs scaling as P^2 while only yielding a sensitivity growing as $\log(P)$. Therefore, the experiment is non-conclusive for relus: there may or may not be compressions happening above the $\log(P)$ bound. We are unable to further comment on the necessity for saturated activations to observe compressions, but we further investigate the case of the tanh activations.

Saxe and collaborators [128] argued that the loss of information in stochastic tanh networks results form the interplay of the saturation of the tanh activation and the blurring induced by noise. During neural network training, the weight magnitudes, initialized at small values, typically grow and with them the pre-activations as well. In turns, the amplitude of multiplicative noises follows this evolution. Therefore the greater information drops observed for the latter types of noises are expected. Similarly the initial phase of increase of the mutual information for the constant noise can be understood as the improvement of an effective signal-to-noise ratio as the pre-activations grow at the beginning of training while the noise is fixed.

To assess whether this geometric interpretation of information trajectories is sufficient we also report mutual informations for a network of similar architecture but with Gaussian random weight matrices of same variance. Thereby we test whether the mere increase of the weights amplitude explains the compression. For all noises, the random network transmits more information than the trained one, and the upper bound of the estimator is almost always saturated for the random network. Hence, the compression seen during training may be due to saturations yet 'organized' by the learning.

5.3.2 Discussion

The concepts and preliminary results reported by [147, 137] have spurred a significant interest in the deep learning community to the information theoretic analysis of neural network training. Nevertheless, the observables proposed as relevant - mutual informations between hidden layers and inputs or outputs - are ill-defined in deterministic neural networks that are commonly used for prediction in supervised learning. The application of the information bottleneck concepts to learning of purely deterministic rules also suffer some caveats investigated by [73]. Otherwise, it is possible to regularize the definition of mutual informations by discretization [137] or by assuming noisy activations [74, 128]. Yet the outcome of experiments sometimes depends on this arbitrary choice of regularization [128]. It was argued that the observed compressions in this context can nonetheless be interpreted as a measure of the clustering of the representations learned by the network [48]. Indeed, using discretization or the non-parametric estimate with added noise, we understand that overlaps in the hidden activation patterns of different inputs induce a drop of mutual information. The choice of characteristic strength of the regularization (bins spacing, noise amplitude etc.) then controls the scale of the clustering being probed. The above cited works could not observe compressions in this setting for unbounded activations, but we note that their estimator suffers from a limited sensitivity to high mutual information and interesting phenomenology may be hidden above the limiting upper bound. In particular, in Section 5.2.2, we report compressions

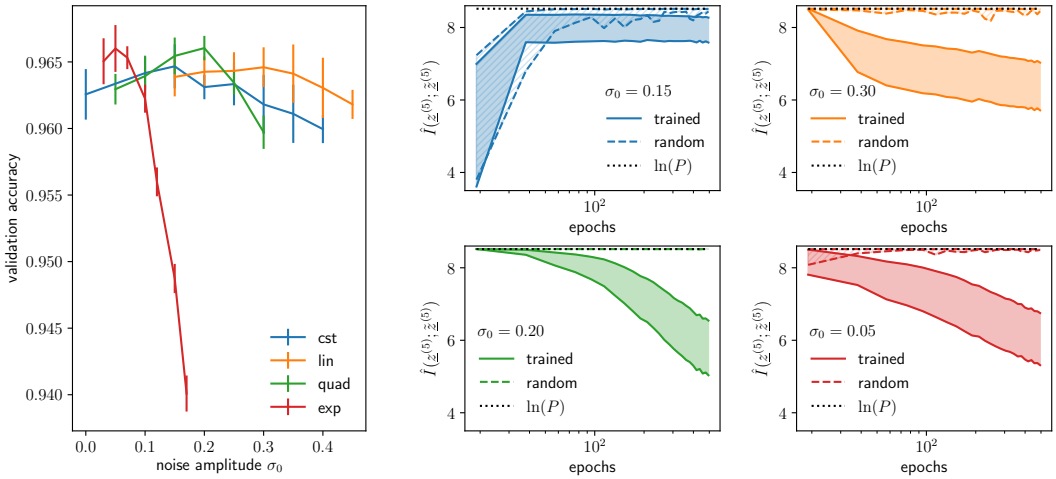


Figure 5.11: Results for 5-hidden layer network with tanh activation on MNIST **Left:** Average and standard deviation of the best accuracy on the MNIST test set over 10 runs for the different types of noise as a function of the noise amplitude parameter σ_0. **Right:** Evolution the mutual information bounds for $P = 5000$ at the last hidden layer over the course of training for the different noises at the selected value of σ_0 (solid lines with solid colored region in between). The dashed lines and regions corresponds to the bounds computed for an identical network with random Gaussian weight matrices of same variance as the trained weight matrix at the same epoch.

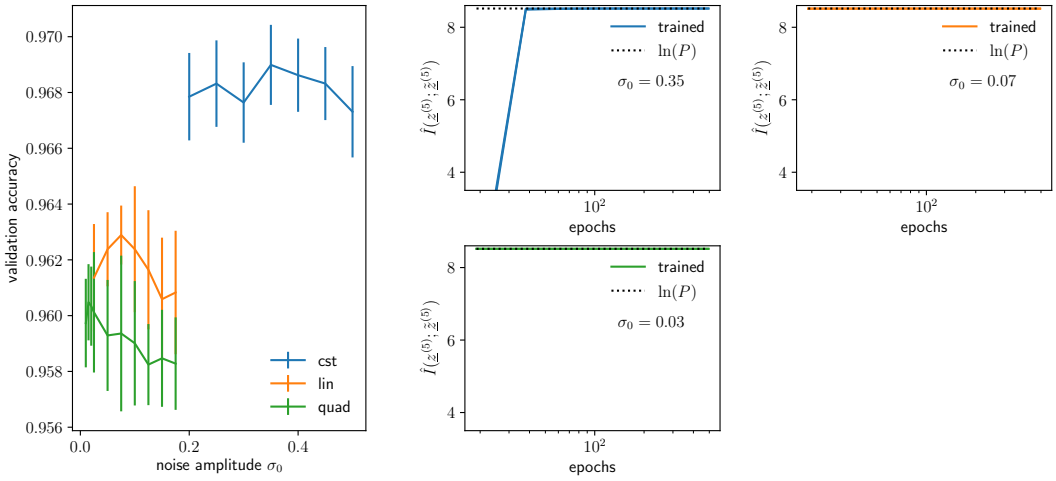


Figure 5.12: Results for 5-hidden layer network with relu activation on MNIST **Left:** Average and standard deviation of the best accuracy on the test set over 10 runs for the different types of noise as a function of the noise amplitude parameter σ_0. Here, there is no training with exponential noise, unstable for the unbounded relu activations. **Right:** Evolution of the mutual information bounds for $P = 5000$ at the last hidden layer over the course of training for the different noises at the selected value of σ_0 (solid lines with solid colored region in between). The upper bound of the non-parametric estimator is saturated.

during the training of relu networks on synthetic data sets, using the mutual information estimator based on the replica method. The issue of definition can also be circumvented by considering neural networks in the learning phase regularized by the addition of noise. On the example of the MNIST classification, using the non-parametric estimator [72], we observed that tanh activated network display varying level of compression but we are not able to draw conclusions for the case of relu activations.

Therefore we are recalled to the complexity of the estimation of mutual informations that appears as a serious obstacle to the validation of a theory of deep learning based on information theoretic principles. Non parametric methods are generic strategy of evaluation yet they are found

insufficient to investigate all the interesting cases and suffer in particular when the dimension of the problem (i.e. the size of the hidden layers) increases. Using advanced methods from statistical physics we were able to follow precisely information trajectories for certain models of classification experiments - for different non-linear activations and for hidden layer of high dimensions. Still, these models are constrained to have weight matrices with a reduced number of degrees of freedom, while the concept of over-parametrization emerges as a key component to the success of deep learning (see e.g. [141, 46] for a discussion of the state-of-understanding and recent results). Another a priori promising non-parametric estimator, MINE [12], relies on the optimization of a deep neural network. Yet its accuracy in high-dimension remains unverified. Albeit the replica formula could offer an interesting point of comparison in cases where it is known to be exact, it seems difficult to obtain convincing guarantees for the MINE estimator while it relies on neural network optimization. Our aim would precisely be to use it to explain what remains to this day mysterious in deep learning.

The concept of progressive invariance learned by the hidden representation remains a key intuition of the community to justify the advantage of deep neural networks. Yet the information theoretic lense as formulated by [147, 137], which could have yielded a quantitative verification, incurs conceptual and practical problems hard to resolve. A related yet different proposition was made by [140], with interesting results for the unsupervised learning of a certain class of deep neural networks with binary units. It remains to understand if the theory can be robustly generalized and tested on real valued feed forward neural networks used in supervised learning.

6 Towards a model for deep Bayesian (online) learning

One of the great successes of the statistical physics approach to neural networks learning theory initiated by Gardner [44] is the prediction of learning curves in the teacher-student scenario. It corresponds to the computation of the optimal generalization errors under the Bayesian learning paradigm as a function of parameters of the model such as the size of the training set or the level of noise in the teacher mapping from input to outputs. Following early works on the single layer perceptron, variations of a simple two-layer architecture were also considered: the (tree) committee machine [156, 95, 106, 37, 96, 7]. In this model, only the first layer connecting the inputs to the hidden units is trained while the second layer has fixed weights. A current challenge in learning theory is to conduct similar analyses for neural networks with learnable parameters distributed in multiple layers.

To this aim, the possibility of decomposing complicated inference tasks into simpler well-known blocks is a promising lever. Let us rapidly examine the supervised learning of a two-layer feed forward neural network. A training set of P input-output pairs $\{\underline{x}_{(k)}, \underline{y}_{(k)}\}_{k=1}^{P}$ is generated using two weight matrices and noisy activation functions:

$$\forall k = 1 \cdots P, \quad \underline{x}_{0,(k)} \sim p_{x_0}(x_{(k)}) \tag{6.1}$$

$$\underline{y}_{(k)} = g(\underline{\underline{W_0}}^{(2)} g(\underline{\underline{W_0}}^{(1)} \underline{x}_{(k)}; \underline{\epsilon}_1); \underline{\epsilon}_2), \qquad \underline{\epsilon}_{1,2} \sim p_\epsilon(\underline{\epsilon}_{1,2}) \tag{6.2}$$

with dimensions $\underline{x}_{0,(k)} \in \mathbb{R}^N$, $\underline{y}_{(k)} \in \mathbb{R}^Q$, $\underline{\underline{W_0}}^{(1)} \in \mathbb{R}^{M\times N}$, $\underline{\underline{W_0}}^{(2)} \in \mathbb{R}^{Q\times M}$. Introducing the intermediate layer variables $\underline{t} \in \mathbb{R}^M$, and the matrix notations $\underline{\underline{X}} \in \mathbb{R}^{N\times P}$, $\underline{\underline{T}} \in \mathbb{R}^{M\times P}$ and $\underline{\underline{Y}} \in \mathbb{R}^{Q\times P}$ gathering the P samples, we can decompose the student model as

$$\underline{\underline{Y}} = g(\underline{\underline{W}}^{(2)} \underline{\underline{T}}; \underline{\epsilon}_2) \qquad \textit{factorization to recover } \underline{\underline{T}} \textit{ and } \underline{\underline{W}}^{(2)} \textit{ from observed } \underline{\underline{Y}}, \tag{6.3}$$

$$\underline{\underline{T}} = g(\underline{\underline{W}}^{(1)} \underline{\underline{X}}; \underline{\epsilon}_1) \qquad \textit{vectorized GLM to recover } \underline{\underline{W}}^{(1)} \textit{ from observations } \underline{\underline{T}}. \tag{6.4}$$

The GLM model is rather straightforward to solve and combine following the recent multi-layer developments of AMP [84] and VAMP [38]. However, the matrix factorization problem, arising from the ignorance of the hidden states as the first layer weights are unknown, is more complicated. At finite rank, it can be solved by message passing and rigorously characterized in the thermodynamic limit [80, 81, 67], which corresponds in our problem to a number M of hidden units remaining finite as the other dimensions are growing to infinity. Assuming few hidden units should therefore allow us to analyse the learning in the teacher-student scenario of this two-layer neural network by composing the low-rank matrix factorization analysis with a GLM. However, to be able to consider a number of hidden units of same order as the number of inputs and outputs, which is a situation closer to the practical usage of deep neural networks, a surrogate to the analysis of high-rank matrix factorization is needed. We formulate here a proposition towards this goal.

In this Chapter, we present in a first Section new results for GLMs with calibration variables. In a second Section we illustrate the newly derived algorithms and analyses with the example of gain calibration. Finally, we propose in a last Section a direction of research that should allow us to leverage these advances to treat in a constrained setting a high rank matrix factorization and the learning of multi-layer neural networks.

6.1 Cal-AMP revisited

We recall the blind calibration problem for the GLM already discussed in Chapter 3 Section 3.5.2. We have access to P observations $\underline{y}_{(k)} \in \mathbb{R}^M$ gathered in a matrix $\underline{\underline{Y}} \in \mathbb{R}^{M\times P}$. They were generated from the P unknown signals $\underline{x}_{0,(k)}\mathbb{R}^N$, similarly noted $\underline{\underline{X}}_0 \in \mathbb{R}^{N\times P}$, linearly mixed by a known weight matrix $\underline{\underline{W}}$, and pushed through a noisy channel denoted $p^s_{\text{out},0}$ (equivalent to a stochastic

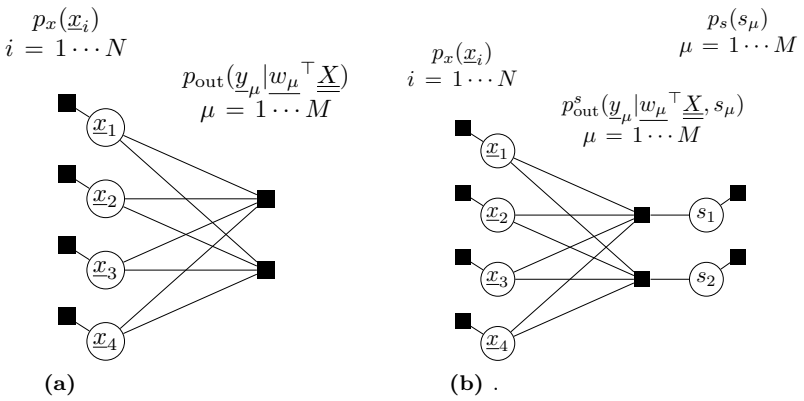


Figure 6.1: **(a)** Factor graph of the GLM on vector variables corresponding to the joint distribution (6.10). **(b)** Factor graph for the GLM on vector variables with not perfectly known channel including calibration variables, corresponding to the joint distribution (6.9).

activation $g_0(\cdot)$ further down this Section). The channel is not perfectly known and depends on the realization of a calibration $\underline{s}_0 \in \mathbb{R}^M$. Under the teacher model including prior distributions,

$$\underline{s}_0 \sim p_{s_0}(\underline{s}_0) \tag{6.5}$$

$$\forall k = 1 \cdots P, \quad \underline{x}_{0,(k)} \sim p_{x_0}(\underline{x}_{0,(k)}) = \prod_{i=1}^{N} p_{x_0}(x_{0,i,(k)}) \tag{6.6}$$

$$\underline{y}_{(k)} \sim p^s_{\text{out},0}(\underline{y}_{(k)}|\underline{\underline{W}}\underline{x}_{0,(k)}, \underline{s}_0) = \prod_{\mu=1}^{M} p^s_{\text{out},0}(y_{(k),\mu}|\underline{\underline{W}}\underline{x}_{0,(k)}, s_{0,\mu}) \tag{6.7}$$

we are interested in the estimation of the unknown signal and calibration variable under a student model for priors and channel,

$$p(\underline{\underline{X}}, \underline{s}|\underline{\underline{Y}}, \underline{\underline{W}}) = \frac{1}{\mathcal{Z}(\underline{\underline{Y}}, \underline{\underline{W}})} p_x(\underline{\underline{X}}) p_s(\underline{s}) p^s_{\text{out}}(\underline{\underline{Y}}|\underline{\underline{W}}\underline{\underline{X}}, \underline{s}) \tag{6.8}$$

$$= \frac{1}{\mathcal{Z}(\underline{\underline{Y}}, \underline{\underline{W}})} \prod_{i=1}^{N} p_x(\underline{x}_i) \prod_{\mu=1}^{M} p_s(s_\mu) p^s_{\text{out}}(\underline{y}_\mu|\underline{w}_\mu^\top \underline{\underline{X}}, s_\mu), \tag{6.9}$$

with $\underline{x}_i \in \mathbb{R}^P$, and $\underline{y}_\mu \in \mathbb{R}^P$. Note that the distribution could be further factorized over the index k of the P observations given the teacher model. The corresponding message passing was derived in [130, 131]. Here we instead restrict to the level of factorization above and consider the AMP algorithm on vector variables.

6.1.1 Derivation through AMP on vector variables

AMP for reconstruction of multiple samples

Before specializing to the calibration problem we are interested in, we present the AMP algorithm on vector variables. Without the calibration variable, the posterior measure we are interested in is

$$p(\underline{\underline{X}}|\underline{\underline{Y}}, \underline{\underline{W}}) = \frac{1}{\mathcal{Z}(\underline{\underline{Y}}, \underline{\underline{W}})} \prod_{i=1}^{N} p(\underline{x}_i) \prod_{\mu=1}^{M} p_{\text{out}}(\underline{y}_\mu|\underline{w}_\mu^\top \underline{\underline{X}}/\sqrt{N}), \quad \underline{x}_i \in \mathbb{R}^P, \quad \underline{y}_\mu \in \mathbb{R}^P. \tag{6.10}$$

where the known entries of matrix $\underline{\underline{W}}$ are drawn i.i.d from a standard normal distribution (note that the scaling in $1/\sqrt{N}$ is here made explicit). The corresponding factor graph is given on Figure 6.1a. This setting was actually recently treated for a model of the committee machine in [7]. The major difference with the fully factorized version on scalar variables is that we consider covariance matrices

between variables coming from the P observations instead of assuming complete independence. We shall recall here the main steps of the derivation, which will be useful both in the following derivation of the State Evolution, and in the presentation of the derivation of the AMP algorithm in combined models.

Belief propagation The belief propagation algorithm is identical to the algorithm presented in Section 3.3.2, except that messages are functions of variables in $\mathbb{R}^P$:

$$\tilde{m}^{(t)}_{\mu\to i}(\underline{x}_i) = \frac{1}{\mathcal{Z}_{\mu\to i}} \int \prod_{i'\neq i} \mathrm{d}\underline{x}_{i'}\; p_{\text{out}}\left(\underline{y}_\mu | \sum_j \frac{W_{\mu j}}{\sqrt{N}}\underline{x}_j\right) \prod_{i'\neq i} m^{(t)}_{i'\to\mu}(\underline{x}_{i'}) \tag{6.11}$$

$$m^{(t+1)}_{i\to\mu}(\underline{x}_i) = \frac{1}{\mathcal{Z}_{i\to\mu}} p_x(\underline{x}_i) \prod_{\mu'\neq\mu} \tilde{m}^{(t)}_{\mu'\to i}(\underline{x}_i). \tag{6.12}$$

To improve readability we drop the time indices in the following derivation, and only specify them in the final algorithm.

Relaxed BP We will now develop messages keeping only terms up to order $O(1/N)$ as we take the thermodynamic limit $N\to+\infty$ (at fixed $\alpha = M/N$). At this order, we will find that it is consistent to consider the messages to be approximately Gaussian, i.e. characterized by their means and co-variances. Thus we define

$$\begin{cases} \hat{\underline{x}}_{i\to\mu} = \int \mathrm{d}\underline{x}\; \underline{x}\; m_{i\to\mu}(\underline{x}) \\ \underline{\underline{C}}^x_{i\to\mu} = \int \mathrm{d}\underline{x}\; \underline{x}\underline{x}^T\; m_{i\to\mu}(\underline{x}) \end{cases} \qquad \begin{cases} \underline{\omega}_{\mu\to i} = \sum_{i'\neq i} \frac{W_{\mu i'}}{\sqrt{N}} \hat{\underline{x}}_{i'\to\mu} \\ \underline{\underline{V}}_{\mu\to i} = \sum_{i'\neq i} \frac{W^2_{\mu i'}}{N} \underline{\underline{C}}^x_{i'\to\mu}, \end{cases} \tag{6.13}$$

where $\underline{\omega}_{\mu\to i}$ and $\underline{\underline{V}}_{\mu\to i}$ are related to the intermediate variable $\underline{z}_\mu = \underline{w}_\mu^\top \underline{\underline{X}}$.

Expansion of $\tilde{m}_{\mu\to i}$ We defined the Fourier transform $\hat{p}_{\text{out}}$ of $p_{\text{out}}(\underline{y}_\mu|\underline{z}_\mu)$ with respect to its argument $\underline{z}_\mu = \underline{w}_\mu^\top \underline{\underline{X}}$,

$$\hat{p}_{\text{out}}(\underline{y}_\mu|\underline{\xi}_\mu) = \int \mathrm{d}\underline{z}_\mu\; \hat{p}_{\text{out}}(\underline{y}_\mu|\underline{z}_\mu)\, e^{-i\underline{\xi}_\mu^\top \underline{z}_\mu}. \tag{6.14}$$

Using reciprocally the Fourier representation of $p_{\text{out}}(\underline{y}_\mu|\underline{z}_\mu)$,

$$p_{\text{out}}(\underline{y}_\mu|\underline{z}_\mu) = \frac{1}{(2\pi^M)} \int \mathrm{d}\underline{\xi}_\mu\; \hat{p}_{\text{out}}(\underline{y}_\mu|\underline{\xi}_\mu)\, e^{i\underline{\xi}_\mu^\top \underline{z}_\mu}, \tag{6.15}$$

we decouple the integrals over the different $\underline{x}_{i'}$ in (6.11),

$$\tilde{m}_{\mu\to i}(\underline{x}_i) \propto \int \mathrm{d}\underline{\xi}_\mu\; \hat{p}_{\text{out}}\left(\underline{y}_\mu|\underline{\xi}_\mu\right) e^{i\frac{W_{\mu i}}{\sqrt{N}}\underline{\xi}_\mu^\top \underline{x}_i} \prod_{i'\neq i} \int \mathrm{d}\underline{x}_{i'}\; m_{i'\to\mu}(\underline{x}_{i'}) e^{i\frac{W_{\mu i'}}{\sqrt{N}}\underline{x}_i \underline{\xi}_\mu^\top \underline{x}_{i'}} \tag{6.16}$$

$$\propto \int \mathrm{d}\underline{\xi}_\mu\; \hat{p}_{\text{out}}\left(\underline{y}_\mu|\underline{\xi}_\mu\right) e^{i\underline{\xi}^\top\left(\frac{W_{\mu i}}{\sqrt{N}}\underline{x}_i + \underline{\omega}_{\mu\to i}\right) - \frac{1}{2}\underline{\xi}^\top \underline{\underline{V}}^{-1}_{\mu\to i}\underline{\xi}} \tag{6.17}$$

where developing the exponentials of the product in (6.11) allows to express the integrals over the $\underline{x}_{i'}$ as a function of the definitions (6.13), before re-exponentiating to obtain the final result (6.17). Now reversing the Fourier transform and performing the integral over $\underline{\xi}$ we can further rewrite

$$\tilde{m}_{\mu\to i}(\underline{x}_i) \propto \int \mathrm{d}\underline{z}_\mu\; p_{\text{out}}\left(\underline{y}_\mu|\underline{z}_\mu\right) e^{-\frac{1}{2}\left(\underline{z}_\mu - \frac{W_{\mu i}}{\sqrt{N}}\underline{x}_i - \underline{\omega}_{\mu\to i}\right)^\top \underline{\underline{V}}^{-1}_{\mu\to i}\left(\underline{z}_\mu - \frac{W_{\mu i}}{\sqrt{N}}\underline{x}_i - \underline{\omega}_{\mu\to i}\right)} \tag{6.18}$$

$$\propto \int \mathrm{d}\underline{z}_\mu\; \mathbb{P}_{\text{out}}(\underline{z}_\mu; \underline{\omega}_{\mu\to i}, \underline{\underline{V}}_{\mu\to i}) e^{\left(\underline{z}_\mu - \underline{\omega}_{\mu\to i}\right)^\top \underline{\underline{V}}^{-1}_{\mu\to i}\frac{W_{\mu i}}{\sqrt{N}}\underline{x}_i - \frac{W^2_{\mu i}}{2N}\underline{x}_i^\top \underline{\underline{V}}^{-1}_{\mu\to i}\underline{x}_i}, \tag{6.19}$$

where we are led to introduce the output update functions

$$\mathbb{P}_{\text{out}}(\underline{z}_\mu;\underline{\omega}_{\mu\to i},\underline{\underline{V}}_{\mu\to i}) = p_{\text{out}}\left(\underline{y}_\mu|\underline{z}_\mu\right)\mathcal{N}(\underline{z}_\mu;\underline{\omega}_{\mu\to i},\underline{\underline{V}}_{\mu\to i})\,, \tag{6.20}$$

$$\mathcal{Z}_{\text{out}}(\underline{y}_\mu,\underline{\omega}_{\mu\to i},\underline{\underline{V}}_{\mu\to i}) = \int \mathrm{d}z_\mu\; p_{\text{out}}\left(\underline{y}_\mu|\underline{z}_\mu\right)\mathcal{N}(\underline{z}_\mu;\underline{\omega}_{\mu\to i},\underline{\underline{V}}_{\mu\to i})\,, \tag{6.21}$$

$$\underline{g_{\text{out}}}(\underline{y}_\mu,\underline{\omega}_{\mu\to i},\underline{\underline{V}}_{\mu\to i}) = \frac{1}{\mathcal{Z}_{\text{out}}}\frac{\partial \mathcal{Z}_{\text{out}}}{\partial \underline{\omega}} \quad \text{and} \quad \partial_{\underline{\omega}}\underline{g_{\text{out}}} = \frac{\partial \underline{g_{\text{out}}}}{\partial \underline{\omega}}, \tag{6.22}$$

where we recall that $\mathcal{N}(\underline{z};\underline{\omega},\underline{\underline{V}})$ is the multivariate Gaussian distribution of mean $\underline{\omega}$ and covariance $\underline{\underline{V}}$. Further expanding the exponential in (6.19) up to order $O(1/N)$ leads to the Gaussian parametrization

$$\tilde{m}_{\mu\to i}(\underline{x}_i) \propto 1 + \frac{W_{\mu i_{\mu i}}}{\sqrt{N}}\underline{g_{\text{out}}}^1\underline{x}_i + \frac{W_{\mu i_{\mu i}}^2}{2N}\underline{x}_i{}^T(\underline{g_{\text{out}}}\underline{g_{\text{out}}}^T + \partial_{\underline{\omega}}\underline{\underline{g_{\text{out}}}}^1)\underline{x}_i \tag{6.23}$$

$$\propto e^{\underline{B}_{\mu\to i}{}^T\underline{x}_i - \frac{1}{2}\underline{x}_i{}^T\underline{\underline{A}}_{\mu\to i}\underline{x}_i}, \tag{6.24}$$

with

$$\begin{cases} \underline{B}_{\mu\to i} = \frac{W_{\mu i_{\mu i}}}{\sqrt{N}}\underline{g_{\text{out}}}^2(\underline{y}_\mu,\underline{\omega}_{\mu\to i},\underline{\underline{V}}_{\mu\to i}) \\ \underline{\underline{A}}_{\mu\to i} = -\frac{W_{\mu i_{\mu i}}}{N}\partial_{\underline{\omega}}\underline{\underline{g_{\text{out}}}}(\underline{y}_\mu,\underline{\omega}_{\mu\to i},\underline{\underline{V}}_{\mu\to i}). \end{cases} \tag{6.25}$$

Consistency with $m_{i\to\mu}$ Inserting the Gaussian approximation of $\tilde{m}_{\mu\to i}$ in the definition of $m_{i\to\mu}$, we get the parametrization

$$m_{i\to\mu}(\underline{x}_i) \propto p_x(\underline{x}_i)\prod_{\mu'\neq\mu} e^{\underline{B}_{\mu'\to i}{}^T\underline{x}_i - \frac{1}{2}\underline{x}_i{}^T\underline{\underline{A}}_{\mu'\to i}\underline{x}_i} \propto p_x(\underline{x}_i)e^{-\frac{1}{2}(\underline{x}_i-\underline{\lambda}_{i\to\mu})^T\underline{\underline{\sigma}}_{i\to\mu}^{-1}(\underline{x}_i-\underline{\lambda}_{i\to\mu})} \tag{6.26}$$

with

$$\begin{cases} \underline{\lambda}_{i\to\mu} = \underline{\underline{\sigma}}_{i\to\mu}\left(\sum_{\mu'\neq\mu}\underline{B}_{\mu'\to i}\right) \\ \underline{\underline{\sigma}}_{i\to\mu} = \left(\sum_{\mu'\neq\mu}\underline{\underline{A}}_{\mu'\to i}\right)^{-1}. \end{cases} \tag{6.27}$$

Closing the equations Ensuring the consistency with the definitions (6.13) of mean and covariance of $m_{i\to\mu}$ we finally close our set of equations by defining the input update functions

$$\mathcal{Z}^x = \int \mathrm{d}x\; p_x(\underline{x})e^{-\frac{1}{2}(\underline{x}-\underline{\lambda})^\top\sigma^{-1}(\underline{x}-\underline{\lambda})} \tag{6.28}$$

$$\underline{f}_1^x(\underline{\lambda},\underline{\sigma}) = \frac{1}{\mathcal{Z}^x}\int \mathrm{d}\underline{x}\;\underline{x}\,p_x(\underline{x})e^{-\frac{1}{2}(\underline{x}-\underline{\lambda})^\top\underline{\underline{\sigma}}^{-1}(\underline{x}-\underline{\lambda})} \tag{6.29}$$

$$\underline{\underline{f}}_2^x(\underline{\lambda},\underline{\sigma}) = \frac{1}{\mathcal{Z}^x}\int \mathrm{d}x\;\underline{x}\underline{x}^\top\,p_x(\underline{x})e^{-\frac{1}{2}(\underline{x}-\underline{\lambda})^\top\underline{\underline{\sigma}}^{-1}(\underline{x}-\underline{\lambda})} - \underline{f}_1^x(\underline{\lambda},\underline{\sigma})\underline{f}_1^x(\underline{\lambda},\underline{\sigma})^\top, \tag{6.30}$$

so that

$$\begin{cases} \hat{\underline{x}}_{i\to\mu} = \underline{f}_1^x(\underline{\lambda}_{i\to\mu},\underline{\sigma}_{i\to\mu}) \\ \underline{\underline{C}}^x_{i\to\mu} = \underline{\underline{f}}_2^x(\underline{\lambda}_{i\to\mu},\underline{\sigma}_{i\to\mu}). \end{cases} \tag{6.31}$$

Approximate message passing Given the scaling of the weights it is possible to further simplify the algorithm by considering the approximated marginals

$$m_i(\underline{x}_i) = \frac{1}{\mathcal{Z}_i}p_x(\underline{x}_i)e^{-\frac{1}{2}(\underline{x}-\underline{\lambda}_i)^\top\underline{\underline{\sigma}}_i^{-1}(\underline{x}-\underline{\lambda}_i)} \qquad \text{with} \qquad \begin{cases} \underline{\lambda}_i = \underline{\underline{\sigma}}_i\left(\sum_{\mu=1}^M\underline{B}_{\mu\to i}\right) \\ \underline{\underline{\sigma}}_i = \left(\sum_\mu^M\underline{\underline{A}}_{\mu\to i}\right)^{-1}. \end{cases} \tag{6.32}$$

Defining likewise parameters $\underline{\omega}_\mu$, $\underline{\underline{V}}_\mu$ and $\hat{\underline{x}}_i$, $\underline{\underline{C}}^x_i$, and considering their relations to the original $\underline{\lambda}_{i\to\mu}$, $\underline{\underline{\sigma}}_{i\to\mu}$, $\underline{\omega}_{\mu\to i}$, $\underline{\underline{V}}_{\mu\to i}$, $\hat{\underline{x}}_{i\to\mu}$ and $\underline{\underline{C}}^x_{i\to\mu}$ we get the vectorized AMP for the GLM presented in Algorithm 7. Note that this step was detailed for Boltzmann machines in Section 4.1.2.

Algorithm 7 Approximate Message Passing for vectors

Input: matrix $\underline{\underline{Y}} \in \mathbb{R}^{M\times P}$ and matrix $\underline{\underline{W}} \in \mathbb{R}^{M\times N}$:
Initialize: $\hat{\underline{x}}_i$, $\underline{\underline{C}}^x_i$ $\forall i$ and $\underline{g_{\text{out}}}_\mu$, $\partial_\omega \underline{\underline{g_{\text{out}}}}_\mu$ $\forall \mu$

repeat

1) Estimate mean and variance of $\underline{z}_\mu$ given current $\hat{\underline{x}}_i$

$$\underline{\underline{V}}^{(t)}_\mu = \sum_{i=1}^N \frac{W^2_{\mu i}}{N} \underline{\underline{C}}^{x\ (t)}_i \tag{6.33}$$

$$\underline{\omega}^{(t)}_\mu = \sum_{i=1}^N \frac{W_{\mu i}}{\sqrt{N}} \hat{\underline{x}}^{(t)}_i - \sum_{i=1}^N \frac{W^2_{\mu i}}{N} (\underline{\underline{\sigma}}^{(t)}_i)^{-1} \underline{\underline{C}}^{x(t)}_i \underline{\underline{\sigma}}_i \underline{g_{\text{out}}}^{(t-1)}_\mu \tag{6.34}$$

2) Estimate mean and variance of the gap between optimal $\underline{z}_\mu$ and $\underline{\omega}_\mu$ given $\underline{y}_\mu$

$$\partial_\omega \underline{\underline{g_{\text{out}}}}^{(t)}_\mu = \partial_\omega \underline{\underline{g_{\text{out}}}}(\underline{y}_\mu, \underline{\omega}^{(t)}_\mu, \underline{\underline{V}}^{(t)}_\mu) \tag{6.35}$$

$$\underline{g_{\text{out}}}^{(t)}_\mu = \underline{g_{\text{out}}}(\underline{y}_\mu, \underline{\omega}^{(t)}_\mu, \underline{\underline{V}}^{(t)}_\mu) \tag{6.36}$$

3) Estimate mean and variance of $\underline{x}_i$ given current optimal $\underline{z}_\mu$

$$\underline{\underline{\sigma}}^{(t)}_i = \left(-\sum_{\mu=1}^M \frac{W^2_{\mu i}}{N} \partial_\omega \underline{\underline{g_{\text{out}}}}^{(t)}_\mu \right)^{-1} \tag{6.37}$$

$$\underline{\lambda}^{(t)}_i = \hat{\underline{x}}^{(t)}_i + \underline{\underline{\sigma}}^{(t)}_i \left(\sum_{\mu=1}^M \frac{W_{\mu i}}{\sqrt{N}} \underline{g_{\text{out}}}^{(t)}_\mu \right) \tag{6.38}$$

4) Estimate of mean and variance of $\underline{x}_i$ augmented of the information about the prior

$$\underline{\underline{C}}^{x\ (t+1)}_i = \underline{\underline{f}}^x_2(\underline{\lambda}^{(t)}_i, \underline{\underline{\sigma}}^{(t)}_i) \tag{6.39}$$

$$\hat{\underline{x}}^{(t+1)}_i = \underline{f}^x_1(\underline{\lambda}^{(t)}_i, \underline{\underline{\sigma}}^{(t)}_i) \tag{6.40}$$

until convergence

Treatment of calibration variables

Heuristic derivation To include calibration variables, we need to consider the factor graph on Figure 6.1b and augment the belief propagation with a set of messages related to the s_μ. Without going back through the entire derivation the final algorithm can easily be guessed. The posterior distribution on $\underline{\underline{X}}$ in the presence of the calibration variable $\underline{s}$ is a special case of the GLM on vector variables examined above with the effective channel

$$p_{\text{out}}(\underline{y}_\mu | \underline{w}_\mu{}^\top \underline{\underline{X}}) = \int \mathrm{d}s_\mu \; p^s_{\text{out}}(\underline{y}_\mu | \underline{w}_\mu{}^\top \underline{\underline{X}}, s_\mu) p_s(s_\mu). \tag{6.41}$$

Thus to reconstruct $\underline{\underline{X}}$ one can directly use Algorithm 7, with output functions (6.20)-(6.22) that will include a marginalization over $\underline{s}$:

$$\mathcal{Z}_{\text{out}}(\underline{y}_\mu, \underline{\omega}_\mu, \underline{\underline{V}}_\mu) = \int \mathrm{d}\underline{z}_\mu \int \mathrm{d}s_\mu \; p^s_{\text{out}}\left(\underline{y}_\mu | \underline{z}_\mu, s_\mu\right) p_s(s_\mu) \mathcal{N}(\underline{z}_\mu; \underline{\omega}_\mu, \underline{\underline{V}}_\mu). \tag{6.42}$$

The estimate of a variable constructed by (sum-product) AMP is always the mean of the approximate posterior marginal distribution. For the calibration variable, the AMP posterior is already

displayed in the above $\mathcal{Z}_{\text{out}}$,

$$m^s_\mu(s_\mu) = \frac{1}{\mathcal{Z}_{\text{out}}} \int \mathrm{d}\underline{z}_\mu \; p^s_{\text{out}}\left(\underline{y}_\mu|\underline{z}_\mu, s_\mu\right) p_s(s_\mu)\mathcal{N}(\underline{z}_\mu; \underline{\omega}_\mu, \underline{\underline{V}}_\mu). \tag{6.43}$$

So that at convergence of Algorithm 7, we can compute the estimate and uncertainty on the calibration variable

$$\hat{s}_\mu = f^s_1(\underline{y}_\mu, \underline{\omega}_\mu, \underline{\underline{V}}_\mu) = \frac{1}{\mathcal{Z}_{\text{out}}} \int \mathrm{d}s_\mu \; s_\mu \int \mathrm{d}\underline{z}_\mu \; p^s_{\text{out}}\left(\underline{y}_\mu|\underline{z}_\mu, s_\mu\right) p_s(s_\mu)\mathcal{N}(\underline{z}_\mu; \underline{\omega}_\mu, \underline{\underline{V}}_\mu) \tag{6.44}$$

$$C^s_\mu = f^s_2(\underline{y}_\mu, \underline{\omega}_\mu, \underline{\underline{V}}_\mu) = \frac{1}{\mathcal{Z}_{\text{out}}} \int \mathrm{d}s_\mu \; s^2_\mu \int \mathrm{d}\underline{z}_\mu \; p^s_{\text{out}}\left(\underline{y}_\mu|\underline{z}_\mu, s_\mu\right) p_s(s_\mu)\mathcal{N}(\underline{z}_\mu; \underline{\omega}_\mu, \underline{\underline{V}}_\mu) - \hat{s}^2_\mu. \tag{6.45}$$

Relation to original Cal-AMP derivation In [130, 131], the Cal-AMP algorithm was derived from the belief propagation equations on the scalar variables of the fully factorized distribution (over N, M and P). It is equivalent to Algorithm 7 if covariance matrices $\underline{\underline{V}}_\mu$, $\partial_{\underline{\underline{\omega}}} \underline{\underline{g}}_{\text{out}}{}_\mu$, $\underline{\underline{\sigma}}_i$, $\underline{\underline{C}}^x_i$ are assumed to be diagonal. However, we recall that BP is only exact on a tree, where the incoming message at each node are truly independent. On the dense factor graph of the GLM on scalar variables, this is approximately exact in the thermodynamic limit due to the random mixing, and small scaling, of the weight matrix $\underline{\underline{W}}$. Here by considering the reconstruction of P samples at the same time, sharing a given realization $\underline{s}$ of the calibration variable, additional correlations may arise. Writing the message passing on the vector variables in $\mathbb{R}^P$ allows not to neglect them.

6.1.2 State Evolution for Cal-AMP

In the thermodynamic limit, recall that the performance of the AMP algorithm can be characterized by a set of simpler equations corresponding to an averaging over the disorder (here $\underline{\underline{X}}_0$, $\underline{\underline{Y}}$ and $\underline{\underline{W}}$), referred to as State Evolution. In Chapter 3 we have seen that there are two ways of deriving the State Evolution, either from the direct averaging of the AMP steps or from the saddle point equations of the replica free energy associated with the problem under the Replica Symmetric (RS) ansatz. In [7], the teacher-student matched setting of the GLM on vectors is examined through the replica approach and the Bayes optimal State Evolution equations are obtained through this second strategy. In the following we present the alternative derivation of the State Evolution equations from the message passing and without assuming a priori matched teacher and student. To this end, our starting point will be the r-BP equations. Ultimately, we also introduce new State Evolution equations following the reconstruction of calibration variables.

State Evolution derivation in mismatched setting

Defintion of the overlaps The overlaps are here $P \times P$ matrices

$$\underline{\underline{q}} = \frac{1}{N}\sum_{i=1}^N \underline{\hat{x}}_i \underline{\hat{x}}_i^T, \quad \underline{\underline{m}} = \frac{1}{N}\sum_{i=1}^N \underline{\hat{x}}_i \underline{x}_{0,i}{}^T, \quad \underline{\underline{q}}_0 = \frac{1}{N}\sum_{i=1}^N \underline{x}_{0,i} \underline{x}_{0,i}{}^T. \tag{6.46}$$

Output parameters Under Gaussian statistics of the entries of $\underline{\underline{W}}$, the variable $\underline{\omega}_{\mu\to i}$ defined in (6.13) is a sum of independent Gaussian variables and follows itself a Gaussian distribution. Its first and second moments are

$$\mathbb{E}_{\underline{\underline{W}}}\left[\underline{\omega}_{\mu\to i}\right] = \frac{1}{\sqrt{N}} \sum_{i'\neq i} \mathbb{E}_{\underline{\underline{W}}}\left[W_{\mu i}\right] \underline{\hat{x}}_{i'\to\mu} = 0\,, \tag{6.47}$$

$$\mathbb{E}_{\underline{\underline{W}}}\left[\underline{\omega}_{\mu\to i}\underline{\omega}_{\mu\to i}^T\right] = \frac{1}{N}\sum_{i'\neq i}\mathbb{E}_{\underline{\underline{W}}}\left[W_{\mu i}^2\right]\underline{\hat{x}}_{i'\to\mu}\underline{\hat{x}}_{i'\to\mu}^T = \frac{1}{N}\sum_{i=1}^{N}\underline{\hat{x}}_{i\to\mu}\underline{\hat{x}}_{i\to\mu}^T + O\left(1/N\right) \tag{6.48}$$

$$= \frac{1}{N}\sum_{i=1}^{N}\underline{\hat{x}}_i\underline{\hat{x}}_i^T - \partial_{\underline{\lambda}}\underline{f}_1^x\underline{\underline{\sigma}}_i B_{\mu\to i}\underline{\hat{x}}_i^T - \left(\partial_{\underline{\lambda}}\underline{f}_1^x\underline{\underline{\sigma}}_i B_{\mu\to i}\underline{\hat{x}}_i^T\right)^T + O\left(1/N\right) \tag{6.49}$$

$$= \frac{1}{N}\sum_{i=1}^{N}\underline{\hat{x}}_i\underline{\hat{x}}_i^T + O\left(1/\sqrt{N}\right) \tag{6.50}$$

where we used the fact that $B_{\mu\to i}$ defined in (6.25) is of order $O(1/\sqrt{N})$. Similarly, the variable $\underline{z}_{\mu\to i} = \sum_{i'\neq i}\frac{W_{\mu i'}}{\sqrt{N}}\underline{x}_{i'}$ is Gaussian with first and second moments

$$\mathbb{E}_{\underline{\underline{W}}}\left[\underline{z}_{\mu\to i}\right] = \frac{1}{\sqrt{N}}\sum_{i'\neq i}\mathbb{E}_{\underline{\underline{W}}}\left[W_{\mu i}\right]\underline{x}_{0,i'} = 0\,, \tag{6.51}$$

$$\mathbb{E}_{\underline{\underline{W}}}\left[\underline{z}_{\mu\to i}\underline{z}_{\mu\to i}^T\right] = \frac{1}{N}\sum_{i=1}^{N}\underline{x}_{0,i}\underline{x}_{0,i}{}^T + O\left(1/\sqrt{N}\right). \tag{6.52}$$

Furthermore, their covariance is

$$\mathbb{E}_{\underline{\underline{W}}}\left[\underline{z}_{\mu\to i}\underline{\omega}_{\mu\to i}^T\right] = \frac{1}{N}\sum_{i'\neq i}\mathbb{E}_{\underline{\underline{W}}}\left[W_{\mu i}^2\right]\underline{x}_{0,i'}\underline{\hat{x}}_{i'\to\mu}^T = \frac{1}{N}\sum_{i=1}^{N}\underline{x}_{0,i'}\underline{\hat{x}}_{i\to\mu}^T + O\left(1/N\right) \tag{6.53}$$

$$= \frac{1}{N}\sum_{i=1}^{N}\underline{x}_{0,i'}\underline{\hat{x}}_i^T - \underline{x}_{0,i'}\partial_{\underline{\lambda}}\underline{f}_1^x\underline{\underline{\sigma}}_i\underline{B}_{\mu\to i}^T + O\left(1/N\right) \tag{6.54}$$

$$= \frac{1}{N}\sum_{i=1}^{N}\underline{x}_{0,i'}\underline{\hat{x}}_i^T + O\left(1/\sqrt{N}\right). \tag{6.55}$$

Hence we find that for all μ-s and all i-s, $\underline{\omega}_{\mu\to i}$ and $\underline{z}_{\mu\to i}$ are approximately jointly Gaussian in the thermodynamic limit following a unique distribution $\mathcal{N}\left(\underline{z}_{\mu\to i}, \underline{\omega}_{\mu\to i};\, \underline{0},\, \underline{\underline{Q}}\right)$ with the bloc covariance matrix

$$\underline{\underline{Q}} = \begin{bmatrix} \underline{\underline{q}}_0 & \underline{\underline{m}} \\ \underline{\underline{m}}^\top & \underline{\underline{q}} \end{bmatrix}. \tag{6.56}$$

For the variance message $\underline{\underline{V}}_{\mu\to i}$, also defined in (6.13), we have

$$\mathbb{E}_{\underline{\underline{W}}}\left[\underline{\underline{V}}_{\mu\to i}\right] = \sum_{i'\neq i}\mathbb{E}_{\underline{\underline{W}}}\left[\frac{W_{\mu i}}{N}^2\right]\underline{\underline{C}}_{i'\to\mu}^x = \sum_{i=1}^{N}\frac{1}{N}\underline{\underline{C}}_{i\to\mu}^x + O\left(1/N\right) \tag{6.57}$$

$$= \sum_{i=1}^{N}\frac{1}{N}\underline{\underline{C}}_i^x + O\left(1/\sqrt{N}\right), \tag{6.58}$$

where using the developments of $\underline{\lambda}_{i\to\mu}$ and $\underline{\underline{\sigma}}_{i\to\mu}$ (6.27), along with the scaling of $\underline{B}_{\mu\to i}$ in $O(1/\sqrt{N})$ we replaced

$$\underline{\underline{C}}_{i\to\mu}^x = \underline{\underline{f}}_2^x(\underline{\lambda}_{i\to\mu}, \underline{\underline{\sigma}}_{i\to\mu}) = \underline{\underline{f}}_2^x(\underline{\lambda}_i, \underline{\underline{\sigma}}_i) - \partial_{\underline{\lambda}}\underline{\underline{f}}_2^x\underline{\underline{\sigma}}_i\underline{B}_{\mu\to i}^T = \underline{\underline{f}}_2^x(\underline{\lambda}_i, \underline{\underline{\sigma}}_i) + O\left(1/\sqrt{N}\right). \tag{6.59}$$

Futhermore, we can check that

$$\lim_{N\to+\infty}\mathbb{E}_{\underline{\underline{W}}}\left[\underline{\underline{V}}_{\mu\to i}^2 - \mathbb{E}_{\underline{\underline{W}}}\left[\underline{\underline{V}}_{\mu\to i}\right]^2\right] = 0, \tag{6.60}$$

meaning that all $\underline{\underline{V}}_{\mu\to i}$ concentrate on their identical mean in the termodynamic limit, which we note

$$\underline{\underline{V}} = \sum_{i=1}^{N} \frac{1}{N} \underline{\underline{C}}^x_i. \tag{6.61}$$

Input parameters Here we use the re-parametrization trick to express $\underline{y}_\mu$ as a function $g_0(\cdot)$ taking a noise $\underline{\epsilon}_\mu \sim p_\epsilon(\underline{\epsilon}_\mu)$ as a second input: $\underline{y}_\mu = g_0(\underline{w}_\mu^\top \underline{\underline{X}}_0, s_{0,\mu}, \underline{\epsilon}_\mu)$. Following (6.25) and (6.32),

$$\underline{\underline{\sigma}}_i^{-1}\underline{\lambda}_i = \sum_{\mu=1}^{M} \frac{W_{\mu i}}{\sqrt{N}} \underline{\underline{g_{\text{out}}}} \left(\underline{y}_\mu, \underline{\omega}_{\mu\to i}, \underline{\underline{V}}_{\mu\to i}\right) \tag{6.62}$$

$$= \sum_{\mu=1}^{M} \frac{W_{\mu i}}{\sqrt{N}} \underline{\underline{g_{\text{out}}}} \left(g_0 \left(\sum_{i'\neq i} \frac{W_{\mu i'}}{\sqrt{N}} \underline{x}_{0,i'} + \frac{W_{\mu i}}{\sqrt{N}} \underline{x}_{0,i}, \underline{\epsilon}_\mu \right), \underline{\omega}_{\mu\to i}, \underline{\underline{V}}_{\mu\to i} \right) \tag{6.63}$$

$$= \sum_{\mu=1}^{M} \frac{W_{\mu i}}{\sqrt{N}} \underline{\underline{g_{\text{out}}}} \left(g_0 \left(\sum_{i'\neq i} \frac{W_{\mu i'}}{\sqrt{N}} \underline{x}_{0,i'}, s_\mu, \underline{\epsilon}_\mu \right), \underline{\omega}_{\mu\to i}, \underline{\underline{V}}_{\mu\to i} \right)$$
$$+ \sum_{\mu=1}^{M} \frac{W_{\mu i}^2}{N} \partial_z \underline{\underline{g_{\text{out}}}} \left(g_0 \left(\underline{z}_{\mu\to i}, s_\mu, \underline{\epsilon}_\mu \right), \underline{\omega}_{\mu\to i}, \underline{\underline{V}}_{\mu\to i} \right) \underline{x}_{0,i}. \tag{6.64}$$

The first term is again a sum of independent random variables, given the $W_{\mu i}$ are i.i.d. with zero mean, of which the messages of type $\mu \to i$ are assumed independent. The second term has non-zero mean and can be shown to concentrate. Finally recalling that all $\underline{\underline{V}}_{\mu\to i}$ also concentrate on $\underline{\underline{V}}$ we obtain the distribution

$$\underline{\underline{\sigma}}_i^{-1}\underline{\lambda}_i \sim \mathcal{N}(\underline{\underline{\sigma}}_i^{-1}\underline{\lambda}_i;\ \alpha \underline{\underline{\hat{m}}}\underline{x}_{0,i}, \sqrt{\alpha}\underline{\underline{\hat{q}}} I_P) \tag{6.65}$$

with

$$\underline{\underline{\hat{q}}} = \int \mathrm{d}\underline{\epsilon}\ p_\epsilon(\underline{\epsilon}) \mathrm{d}s_0\ p_{s_0}(s_0) \int \mathrm{d}\underline{\omega}\ \mathrm{d}\underline{z}\ \mathcal{N}(\underline{z}, \underline{\omega}; \underline{0}, \underline{\underline{Q}}) \underline{\underline{g_{\text{out}}}}(g_0\left(\underline{z}, s_0, \underline{\epsilon}\right), \underline{\omega}, \underline{\underline{V}}) \times \tag{6.66}$$
$$\underline{\underline{g_{\text{out}}}}(g_0\left(\underline{z}, s_0, \underline{\epsilon}\right), \underline{\omega}, \underline{\underline{V}})^T,$$

$$\underline{\underline{\hat{m}}} = \int \mathrm{d}\underline{\epsilon}\ p_\epsilon(\underline{\epsilon}) \mathrm{d}s_0\ p_{s_0}(s_0) \int \mathrm{d}\underline{\omega}\ \mathrm{d}\underline{z}\ \mathcal{N}(\underline{z}, \underline{\omega}; \underline{0}, \underline{\underline{Q}}) \partial_z \underline{\underline{g_{\text{out}}}}(g_0\left(\underline{z}, s_0, \underline{\epsilon}\right), \underline{\omega}, \underline{\underline{V}}). \tag{6.67}$$

For the inverse variance $\underline{\underline{\sigma}}_i^{-1}$ one can check again that it concentrates on its mean

$$\underline{\underline{\sigma}}_i^{-1} = \sum_{\mu=1}^{M} \frac{{W_{\mu i}}^2}{N} \partial_\omega \underline{\underline{g_{\text{out}}}}(\underline{y}_\mu, \underline{\omega}_{\mu\to i}, \underline{\underline{V}}_{\mu\to i}) \simeq \alpha \underline{\underline{\hat{\chi}}} \tag{6.68}$$

$$\underline{\underline{\hat{\chi}}} = -\int \mathrm{d}\underline{\epsilon}\ p_\epsilon(\underline{\epsilon}) \mathrm{d}s_0\ p_{s_0}(s_0) \int \mathrm{d}\underline{\epsilon}\ \mathrm{d}\underline{z}\ \mathcal{N}(\underline{z}, \underline{\omega}; \underline{0}, \underline{\underline{Q}}) \partial_\omega \underline{\underline{g_{\text{out}}}}(g_0\left(\underline{z}, s_0, \underline{\epsilon}\right), \underline{\omega}, \underline{\underline{V}}). \tag{6.69}$$

Closing the equations These statistics of the input parameters must ensure that consistently

$$\underline{\underline{V}} = \frac{1}{N} \sum_{i=1}^{N} \underline{\underline{C}}^x_i = \mathbb{E}_{\underline{\lambda}, \underline{\underline{\sigma}}} \left[\underline{\underline{f}}^x_2(\underline{\lambda}, \underline{\underline{\sigma}}) \right], \tag{6.70}$$

$$\underline{\underline{q}} = \frac{1}{N} \sum_{i=1}^{N} \underline{\hat{x}}_i \underline{\hat{x}}_i^\top = \mathbb{E}_{\underline{\lambda}, \underline{\underline{\sigma}}} \left[\underline{\underline{f}}^x_1(\underline{\lambda}, \underline{\underline{\sigma}}) \underline{\underline{f}}^x_1(\underline{\lambda}, \underline{\underline{\sigma}})^\top \right], \tag{6.71}$$

$$\underline{\underline{m}} = \frac{1}{N} \sum_{i=1}^{N} \underline{\hat{x}}_i \underline{x}_{0,i}{}^\top = \mathbb{E}_{\underline{\lambda}, \underline{\underline{\sigma}}} \left[\underline{\underline{f}}^x_1(\underline{\lambda}, \underline{\underline{\sigma}}) \underline{x}_{0,i}{}^\top \right], \tag{6.72}$$

which gives upon expressing the computation of the expectations

$$\underline{\underline{V}} = \int \mathrm{d}\underline{x}_0 \, p_{x_0}(\underline{x}_0) \int \mathcal{D}\underline{\xi} \, \underline{\underline{f}}_2^x \left((\alpha \underline{\underline{\hat{\chi}}})^{-1} \left(\sqrt{\alpha \underline{\underline{\hat{q}}}} \underline{\xi} + \alpha \underline{\underline{\hat{m}}} \underline{x}_0 \right) ; (\alpha \underline{\underline{\hat{\chi}}})^{-1} \right) , \tag{6.73}$$

$$\underline{\underline{m}} = \int \mathrm{d}\underline{x}_0 \, p_{x_0}(\underline{x}_0) \int \mathcal{D}\underline{\xi} \, \underline{f}_1^x \left((\alpha \underline{\underline{\hat{\chi}}})^{-1} \left(\sqrt{\alpha \underline{\underline{\hat{q}}}} \underline{\xi} + \alpha \underline{\underline{\hat{m}}} \underline{x}_0 \right) ; (\alpha \underline{\underline{\hat{\chi}}})^{-1} \right) \underline{x}_0{}^\top , \tag{6.74}$$

$$\underline{\underline{q}} = \int \mathrm{d}\underline{x}_0 \, p_{x_0}(\underline{x}_0) \int \mathcal{D}\underline{\xi} \, \underline{f}_1^x \left((\alpha \underline{\underline{\hat{\chi}}})^{-1} \left(\sqrt{\alpha \underline{\underline{\hat{q}}}} \underline{\xi} + \alpha \underline{\underline{\hat{m}}} \underline{x}_0 \right) ; (\alpha \underline{\underline{\hat{\chi}}})^{-1} \right) \times \underline{f}_1^x \left((\alpha \underline{\underline{\hat{\chi}}})^{-1} \left(\sqrt{\alpha \underline{\underline{\hat{q}}}} \underline{\xi} + \alpha \underline{\underline{\hat{m}}} \underline{x}_0 \right) ; (\alpha \underline{\underline{\hat{\chi}}})^{-1} \right)^\top . \tag{6.75}$$

The State Evolution analysis of the GLM on the vector variables finally consists in iterating the equations (6.66), (6.67), (6.69), (6.73), (6.74) and (6.75) until convergence.

Reconstruction of the calibration variable In parallel, one can follow the reconstruction of $\underline{s}$ by introducing the scalar overlaps

$$r = \frac{1}{M} \sum_{\mu=1}^{M} \hat{s}_\mu^2, \quad \nu = \frac{1}{M} \sum_{\mu=1}^{M} \hat{s}_\mu s_{0,\mu}, \quad r_0 = \frac{1}{M} \sum_{\mu=1}^{M} s_{0,\mu}^2. \tag{6.76}$$

Recalling the definition of the estimator $\hat{\underline{s}}$ (6.44), and after following the steps of the above derivation, one can see that the overlaps can be computed from the State Evolution variables

$$r = \int \mathrm{d}\underline{\epsilon} \, p_\epsilon(\underline{\epsilon}) \mathrm{d}s_0 \, p_{s_0}(s_0) \int \mathrm{d}\underline{\omega} \, \mathrm{d}\underline{z} \, \mathcal{N}(\underline{z}, \underline{\omega}; \underline{0}, \underline{\underline{Q}}) \, \hat{s} \left(g_0 \left(\underline{z}, s_0, \underline{\epsilon} \right), \underline{\omega}, \underline{\underline{V}} \right)^2 , \tag{6.77}$$

$$\nu = \int \mathrm{d}\underline{\epsilon} \, p_\epsilon(\underline{\epsilon}) \mathrm{d}s_0 \, p_{s_0}(s_0) \int \mathrm{d}\underline{\omega} \, \mathrm{d}\underline{z} \, \mathcal{N}(\underline{z}, \underline{\omega}; \underline{0}, \underline{\underline{Q}}) \, \hat{s} \left(g_0 \left(\underline{z}, s_0, \underline{\epsilon} \right), \underline{\omega}, \underline{\underline{V}} \right) s_0 . \tag{6.78}$$

Performance analysis The mean squared error (MSE) on the reconstruction of $\underline{\underline{X}}$ by the AMP algorithm is then predicted by

$$\mathrm{MSE}(\underline{\underline{X}}) = q - 2m + q_0, \tag{6.79}$$

where the scalar values used here correspond to the (unique) value of the diagonal elements of the corresponding overlap matrices. The fixed points of the iterated equations can subsequently be plugged into (6.77) and (6.78) to compute the MSE in the reconstruction of the calibration variable

$$\mathrm{MSE}(\underline{s}) = r - 2\nu + r_0 . \tag{6.80}$$

Bayes optimal State Evolution

Similarly to the scalar case, the equation can be greatly simplified in the Bayes optimal setting where the statistical model used by the student (priors p_x and p_s, and channel p_{out}) is known to match the teacher. In this case, one can prove that

$$\underline{\underline{q}} = \underline{\underline{m}}, \quad \underline{\underline{V}} = \underline{\underline{q}}_0 - \underline{\underline{m}}, \quad \hat{\underline{\underline{q}}} = \hat{\underline{\underline{m}}} = \hat{\underline{\underline{\chi}}} \quad \text{and} \quad r = \nu. \tag{6.81}$$

And the State Evolution can be reduced to a set of three equations

$$r = \int \mathrm{d}\underline{\epsilon} \, p_\epsilon(\underline{\epsilon}) \mathrm{d}s_0 \, p_{s_0}(s_0) \int \mathrm{d}\underline{\omega} \, \mathrm{d}\underline{z} \, \mathcal{N}(\underline{z}, \underline{\omega}; \underline{0}, \underline{\underline{Q}}) \, \hat{s} \left(g_0 \left(\underline{z}, s_0, \underline{\epsilon} \right), \underline{\omega}, \underline{\underline{q}}_0 - \underline{\underline{m}} \right)^2 , \tag{6.82}$$

$$\underline{\underline{m}} = \int \mathrm{d}\underline{x}_0 \, p_{x_0}(\underline{x}_0) \int \mathcal{D}\underline{\xi} \, \underline{f}_1^x \left((\alpha \underline{\underline{\hat{m}}})^{-1} \left(\sqrt{\alpha \underline{\underline{\hat{m}}}} \underline{\xi} + \alpha \underline{\underline{\hat{m}}} \underline{x}_0 \right) ; (\alpha \underline{\underline{\hat{m}}})^{-1} \right) \underline{x}_0{}^\top \tag{6.83}$$

$$\underline{\underline{\hat{m}}} = \int \mathrm{d}\underline{\epsilon} \, p_\epsilon(\underline{\epsilon}) \mathrm{d}s_0 \, p_{s_0}(s_0) \int \mathrm{d}\underline{\omega} \, \mathrm{d}\underline{z} \, \mathcal{N}(\underline{z}, \underline{\omega}; \underline{0}, \underline{\underline{Q}}) \underline{g}_{\mathrm{out}} \left(g_0 \left(\underline{z}, s_0, \underline{\epsilon} \right), \underline{\omega}, \underline{\underline{q}}_0 - \underline{\underline{m}} \right) \times \underline{g}_{\mathrm{out}} \left(g_0 \left(\underline{z}, s_0, \underline{\epsilon} \right), \underline{\omega}, \underline{\underline{q}}_0 - \underline{\underline{m}} \right) , \tag{6.84}$$

with the bloc covariance matrix

$$\underline{\underline{Q}} = \begin{bmatrix} \underline{\underline{q}}_0 & \underline{\underline{m}} \\ \underline{\underline{m}}^\top & \underline{\underline{m}} \end{bmatrix}. \tag{6.85}$$

6.1.3 Online algorithm and analysis

Additionally, we consider the analysis of the online reconstruction of the calibration variable as the observations $\underline{y}_{(k)} \in \mathbb{R}^M$ are treated successively. For the problem of the reconstruction of $\underline{x}_0$ in the classic GLM problem, we have already discussed in Chapter 3, Section 3.5.1 the streaming AMP algorithm and corresponding State Evolution proposed in [85]. The idea was to use as an effective prior at step $k+1$ the approximate posterior marginal constructed by AMP at step k. Here we readily adapt this strategy by using at step $k+1$ as an effective prior on s_μ the approximate posterior at step k.

Approximate message passing The AMP algorithm consists here in restarting Cal-AMP for a single sample at each new observation $\underline{y}_{(k)}$, while updating the prior used for the calibration variable. From the definition of the approximate posterior (6.43), we obtain the recursion on the effective prior $p_{s_\mu}^{(k+1)}$ on s_μ:

$$p_{s_\mu}^{(k+1)}(s_\mu) = m_\mu^{s,(k)}(s_\mu) \tag{6.86}$$

$$= \frac{1}{\mathcal{Z}_{\text{out}}^{(k)}} \int \mathrm{d}z_{(k),\mu}\, p_{\text{out}}^s\left(y_{(k),\mu}|z_{(k),\mu}, s_\mu\right) p_{s_\mu}^{(k)}(s_\mu) \mathcal{N}(z_{(k),\mu}; \omega_{(k),\mu}, V_{(k),\mu}), \tag{6.87}$$

where the output variables $\omega_{(k),\mu}$ and $V_{(k),\mu}$ correspond to the values at convergence (or at the last iteration $t_{\max}$) of the Cal-AMP algorithm at the previous step (k). In the Section 6.2, we will examine the gain calibration problem and specify effective strategies to implement this recursion within the AMP algorithm.

State Evolution The streaming State Evolution analysis of the calibration reconstruction is also adapted using the above recursion. Note that the effective prior at a given step P depends on the output variables of the algorithm for all the previously seen samples - expending the recursion above we have:

$$p_{s_\mu}^{(P)}(s_\mu) = \frac{1}{\mathcal{Z}_{\text{out}}^{(P)}} \int \prod_{k=1}^{P} \left(\mathrm{d}z_{(k),\mu}\, p_{\text{out}}^s\left(y_{(k),\mu}|z_{(k),\mu}, s_\mu\right) \mathcal{N}(z_{(k),\mu}; \omega_{(k),\mu}, V_{(k),\mu}) \right) p_s(s_\mu) \tag{6.88}$$

$$\mathcal{Z}_{\text{out}}^{(P)} = \mathcal{Z}_{\text{out}}^{(P)}(\{y_{(k),\mu}, \omega_{(k),\mu}, V_{(k),\mu}\}_{k=1}^P) \tag{6.89}$$

where again for each k the output variables $\omega_{(k),\mu}$ and $V_{(k),\mu}$ are the converged values for the corresponding step k. The dependence of the normalization of $\mathcal{Z}_{\text{out}}^{(P)}$ on the output variables (6.89) is reflected in the definitions at step P of ${g_{\text{out}}}^{(P)}$ and $\hat{s}^{(P)}$. Therefore, the State Evolution involving the output functions will feature an averaging on all the output variables relative to the already

processed samples. In the Bayes optimal setting, it becomes

$$r_{(P)} = \int \mathrm{d}\underline{\epsilon}\; p_\epsilon(\underline{\epsilon})\mathrm{d}s_0\; p_{s_0}(s_0) \int \mathrm{d}\omega\; \mathrm{d}z\; \mathcal{N}(z,\omega;\underline{0},\underline{\underline{Q}}_{(P)}) \tag{6.90}$$
$$\prod_{k=1}^{P-1} \int \mathrm{d}\omega_{(k)}\; \mathrm{d}z_{(k)}\; \mathcal{N}(z_{(k)},\omega_{(k)};\underline{0},\underline{\underline{Q}}_{(k)})\; \hat{s}^{(P)}\left(g_0\left(z,s_0,\epsilon\right),\omega,q_0-m\right)^2,$$

$$m_{(P)} = \int \mathrm{d}x_0\; p_{x_0}(x_0) \int \mathcal{D}\xi\; f_1^x\left((\alpha\hat{\chi})^{-1}\left(\sqrt{\alpha\hat{m}}\xi+\alpha\hat{m}_{(P)}x_0\right);(\alpha\hat{m}_{(P)})^{-1}\right)x_0\,, \tag{6.91}$$

$$\hat{m}_{(P)} = \int \mathrm{d}\epsilon\; p_\epsilon(\epsilon)\mathrm{d}s_0\; p_{s_0}(s_0) \int \mathrm{d}\omega\; \mathrm{d}z\; \mathcal{N}(z,\omega;\underline{0},\underline{\underline{Q}}_{(P)})$$
$$\prod_{k=1}^{P-1} \int \mathrm{d}\omega_{(k)}\; \mathrm{d}z_{(k)}\; \mathcal{N}(z_{(k)},\omega_{(k)};\underline{0},\underline{\underline{Q}}_{(k)})\; {g_{\text{out}}}^{(P)}\left(g_0\left(z,s_0,\epsilon\right),\omega,q_0-m\right)^2, \tag{6.92}$$

where the bloc covariance matrices similar to (6.85) are function of the fixed points of each step. In the following Section we will discuss how to implement this State Evolution in practice, focusing on the specific problem of gain calibration.

6.2 Experimental validation on gain calibration

6.2.1 Setting and update functions

As a test case, we consider the following problem of gain calibration. The input signal is known to be ρ-sparse and distributed according to a Gauss-Bernoulli distribution. Each component of the output includes a division by a calibration variable that is uniformly distributed in a positive interval $[a,b]$:

$$\forall i = 1\cdots N\,,\quad p_{x_0}(\underline{x}_i) = \prod_{k=1}^{P}\left(\rho\,\mathcal{N}(x_{(k),i};0,1)+(1-\rho)\,\delta(x_{(k),i})\right)\,, \tag{6.93}$$

$$\forall \mu = 1\cdots M\,,\quad p_{s_0}(s_\mu) = \mathbb{1}_{[a,b]}/(b-a)\,,\quad 0<a<b, \tag{6.94}$$

$$\underline{y}_\mu = \frac{1}{s_{0,\mu}}(\underline{w}_\mu^\top \underline{\underline{X}}_0 + \underline{\epsilon})\,,\quad \underline{\epsilon}\sim\sqrt{\Delta}\mathcal{N}(\underline{\epsilon},\underline{0},I_P)\,. \tag{6.95}$$

Output functions In this setting the output functions have analytical expressions. The channel distribution and partition function are

$$p_{\text{out}}(\underline{y}_\mu|\underline{z}_\mu) = \int_a^b \mathrm{d}s_\mu\; p_s(s_\mu)(s_\mu)^P\mathcal{N}(\underline{z}_\mu;s_\mu y_\mu,\Delta), \tag{6.96}$$

$$\mathcal{Z}_{\text{out}}(\underline{y},\underline{\omega},\underline{\underline{V}}) = \int \mathrm{d}\underline{z}\; p_{\text{out}}(\underline{y}|\underline{z}) = \int_a^b \mathrm{d}s\; p_s(s)(s)^P\mathcal{N}(\underline{z};sy,\Delta) \tag{6.97}$$

which gives

$$\underline{g_{\text{out}}}(\underline{y},\underline{\omega},\underline{\underline{V}}) = \frac{1}{\mathcal{Z}_{\text{out}}}\partial_{\underline{\omega}}\mathcal{Z}_{\text{out}} = (\underline{\underline{V}}+\Delta I_P)^{-1}(\hat{s}(\underline{y},\underline{\omega},\underline{\underline{V}})\underline{y}-\underline{\omega}), \tag{6.98}$$

$$\partial_{\underline{\omega}}\underline{g_{\text{out}}}(\underline{y},\underline{\omega},\underline{\underline{V}}) = C^s(\underline{y},\underline{\omega},\underline{\underline{V}})(\underline{\underline{V}}+\Delta I_P)^{-1}\underline{y}\underline{y}^\top(\underline{\underline{V}}+\Delta I_P)^{-1} - (\underline{\underline{V}}+\Delta I_P)^{-1}. \tag{6.99}$$

These expressions involve the estimate and variance of the calibration variables under the posterior which can be computed as

$$\hat{s}(\underline{y},\underline{\omega},\underline{\underline{V}}) = f_1^s(\underline{y},\underline{\omega},\underline{\underline{V}}) = \frac{\int_a^b \mathrm{d}s\;(s)^{P+1}\mathcal{N}(\underline{z};sy,\Delta)}{\int_a^b \mathrm{d}s\;(s)^P\mathcal{N}(\underline{z};sy,\Delta)} = \frac{\mathcal{I}(P+1,\nu,\delta,a,b)}{\mathcal{I}(P,\nu,\delta,a,b)}, \tag{6.100}$$

$$C^s(\underline{y},\underline{\omega},\underline{\underline{V}}) = f_2^s(\underline{y},\underline{\omega},\underline{\underline{V}}) = \frac{\mathcal{I}(P+2,\nu,\delta,a,b)}{\mathcal{I}(P,\nu,\delta,a,b)} - \hat{s}(\underline{y},\underline{\omega},\underline{\underline{V}})^2, \tag{6.101}$$

where

$$\mathcal{I}(P,\nu,\delta,a,b) = \int_a^b \mathrm{d}s\; s^P e^{-\frac{1}{2\delta}(s-\nu)^2}\,, \tag{6.102}$$

$$\delta = (\underline{y}^\top(\underline{\underline{V}} + \Delta I_P)^{-1}\underline{y})^{-1}\,, \tag{6.103}$$

$$\delta^{-1}\nu = \underline{y}^\top(\underline{\underline{V}} + \Delta I_P)^{-1}\underline{\omega} \tag{6.104}$$

and $\mathcal{I}(P,\nu,\delta,a,b)$ can be computed using gamma functions as explained in [130, 131].

Input functions For a Gauss-Bernoulli prior on the entries of $\underline{\underline{X}}$, and assuming the AMP variances $\underline{\underline{\sigma}}_i$ are diagonal matrices, the input update functions can be written component-wise with scalar arguments:

$$f_1^x(\lambda,\sigma) = \left(\rho\frac{\lambda}{(1+\sigma)^{3/2}}e^{-\frac{\lambda^2}{2(1+\sigma)}}\right)\Bigg/\left(\rho\frac{e^{-\frac{\lambda^2}{2(1+\sigma)}}}{(1+\sigma)^{1/2}} + (1-\rho)\frac{e^{-\frac{\lambda^2}{2\sigma}}}{\sigma^{1/2}}\right)\,, \tag{6.105}$$

and

$$f_2^x(\lambda,\sigma) = \left(\rho\frac{1\sigma(1+\sigma)+(1\lambda)^2}{(1+\sigma)^{5/2}}e^{-\frac{\lambda^2}{2(1+\sigma)}}\right)\Bigg/\left(\rho\frac{e^{-\frac{\lambda^2}{2(1+\sigma)}}}{(1+\sigma)^{1/2}} + (1-\rho)\frac{e^{-\frac{\lambda^2}{2\sigma}}}{\sigma^{1/2}}\right) - {f_{1,k}^x}^2. \tag{6.106}$$

so that $\hat{\underline{x}}_{i,k} = f_1^x(\lambda_{i,k},\sigma_{i,kk})$ and $C^x_{i,kk} = f_2^x((\lambda_{i,k},\sigma_{i,kk})$.

Algorithm 8 Offline Gain Calibration State Evolution

Input: matrix $\underline{\underline{Y}} \in \mathbb{R}^{M\times P}$ and matrix $\underline{\underline{W}} \in \mathbb{R}^{M\times N}$:

Initialize:

$t=0$, $m^{(0)}=0$, $V^{(0)}=q_0=\rho$,

$\forall\mu = 1\cdots N_{\mathrm{MC}}$ $s_{0,\mu}\sim p_{s_0}(s_{0,\mu})$

repeat

1) Draw Monte Carlo samples for update of $\hat{m}$ (6.92) and r (6.90)

$\forall\mu = 1\cdots N_{\mathrm{MC}}$

$\underline{z}_\mu,\underline{\omega}_\mu \sim \mathcal{N}(\underline{z}_\mu,\underline{\omega}_\mu;\underline{0},\underline{\underline{Q}}^{(t)})$

$\underline{\epsilon}\sim p_\epsilon(\underline{\epsilon})$

$\underline{y}_\mu = g_0(\underline{z}_\mu, s_{0,\mu},\underline{\epsilon})$

with $\underline{\underline{Q}}^{(t)} = \begin{bmatrix} q_0 I_P & m^{(t)} I_P \\ m^{(t)} I_P & m^{(t)} I_P \end{bmatrix} \in R^{2P\times 2P}$

2) Compute integrands

2.2) Compute $\hat{s}_\mu(\underline{y}_\mu,\underline{z}_\mu,\underline{\omega}_\mu,\underline{\underline{V}}^{(t)})$ (6.100)

2.3) Compute $\underline{g_{\mathrm{out}}}_\mu(\underline{y}_\mu,\underline{z}_\mu,\underline{\omega}_\mu,\underline{\underline{V}}^{(t)})$ (6.98)

3) Update r and $\hat{m}$

$$r^{(t)} = \frac{1}{N_{\mathrm{MC}}}\sum_{\mu=1}^{N_{\mathrm{MC}}}\left(\hat{s}_\mu^{(t)}\right)^2 \qquad \hat{m}^{(t)} = \frac{1}{N_{\mathrm{MC}}}\sum_{\mu=1}^{N_{\mathrm{MC}}}\left(g_{\mathrm{out}\,\mu}\right)^2$$

3) Update $m^{(t+1)}$ by numerical integration using (6.91), and $V^{(t+1)} = q_0 - m^{(t+1)}$.

$t = t+1$

until convergence

Output: time series $\{V^{(t)}, m^{(t)}, \hat{m}^{(t)}\ ;\ t = 1\cdots t_{\max}\}$

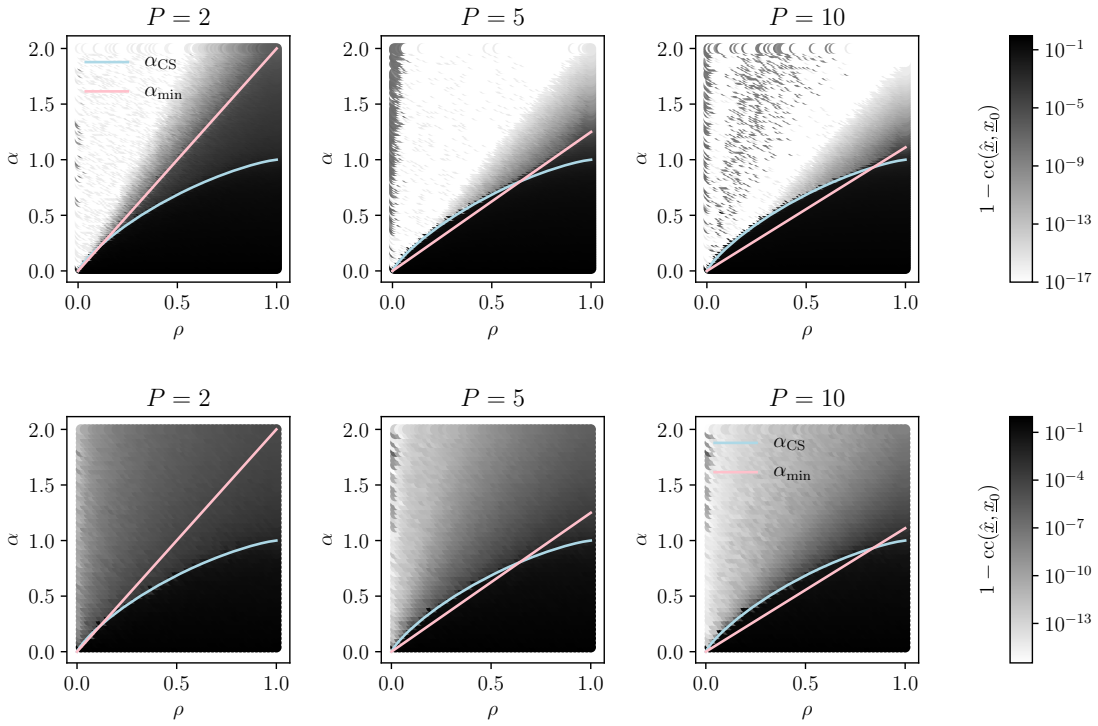


Figure 6.2: Normalized cross correlation between Cal-AMP estimate $\hat{\underline{\underline{X}}}$ and teacher signal $\underline{\underline{X}}_0$ for the gain calibration problem with calibration variables uniformly distributed in $[0.95, 1.05]$ and $N = 10^3$. Diagrams are plotted as a function of the measurement rate $\alpha = M/N$ and the sparsity level ρ for the offline **(top row)** and online **(bottow row)** algorithms, for increasing number of available samples P. The blue line $\alpha_{\rm CS}$ is the phase transition threshold for a perfectly calibrated channel. The pink line $\alpha_{\rm min}$ marks the strict lower bound on the number of measurements necessary for reconstruction. The online Cal-AMP requires more samples than the offline version to achieve comparable errors, nevertheless above the transition relatively low errors are already reached at $P = 10$. We note that the algorithms are sometimes unstable at low ρ, leading to unexpectedly high MSEs (top left corner of top right diagram).

6.2.2 Offline results

The offline AMP algorithm is directly given by Algorithm 7 replacing the values of the output and input functions presented in the previous paragraph. In our numerical tests, we additionally impose that co-variance matrices $\underline{\underline{V}}_\mu$, $\partial_\omega \underline{\underline{g}}_{\rm out}{}_\mu$ and $\underline{\underline{\sigma}}_i$ are diagonal. This assumption lightens numerics so as to consider larger number of samples P. It also allows a comparison with the results of [130, 131]. Under the assumption of diagonal covariance matrices, the State Evolution equations (6.83), (6.84) involve $P \times P$ matrices proportional to the identity (given the P examples are statistically equivalent). Therefore it is sufficient to consider the update of one diagonal element noted respectively m and $\hat{m}$. While the update of m only requires a two dimensional integral that can be performed numerically, we resort to a Monte Carlo for the update of $\hat{m}$. The procedure is described in Algorithm 8.

On the top row of Figure 6.2 and respectively of Figure 6.3, we report the performance of reconstruction by the Cal-AMP algorithm of the signal $\underline{\underline{X}}_0$ and the calibration variable $\underline{s}_0$, in the plane $\rho - \alpha$, for different values of P. These phase diagrams are similar to the ones reported in [130, 131] and will be compared with the online case described in the next Section. On Figure 6.4a, we check numerically that the derived State Evolution predicts the behavior of the AMP algorithm for gain calibration. The MSEs of the two procedures are indeed consistent along the iterations of the algorithm. On Figure 6.4b, we also report an almost perfect agreement of the fixed points of SE and AMP in terms of the MSE on $\underline{\underline{X}}$ as we vary the number of samples P.

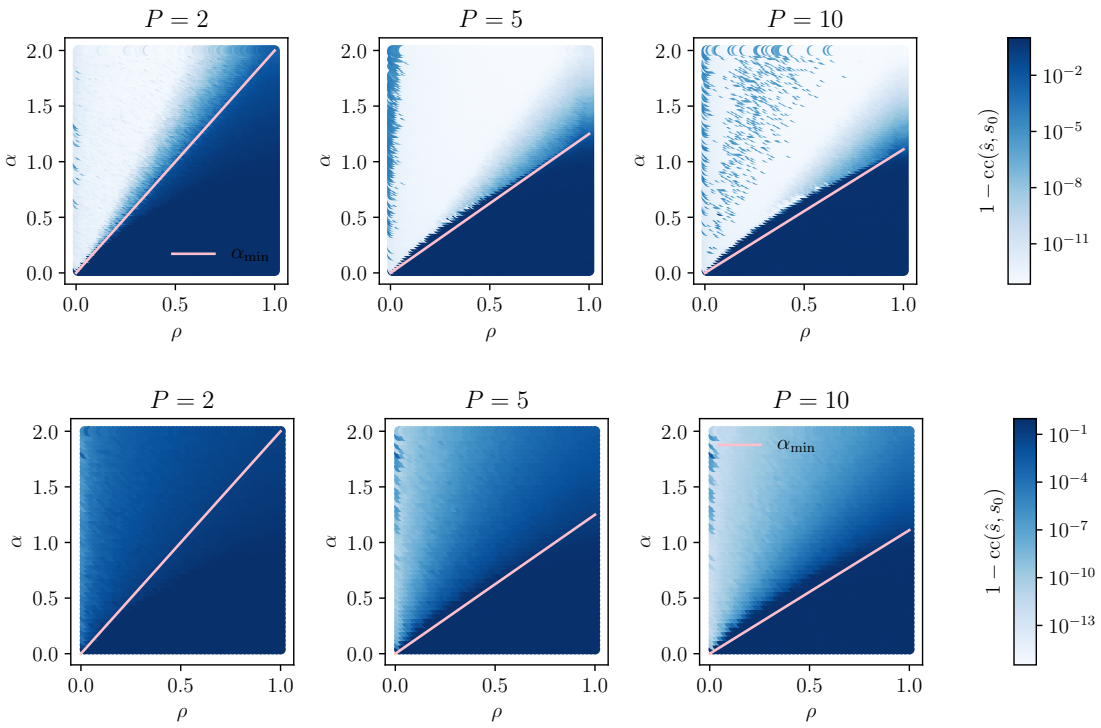


Figure 6.3: Normalized cross correlation between Cal-AMP estimate $\underline{\hat{s}}$ and teacher signal $\underline{s}_0$ for the gain calibration problem with calibration variables uniformly distributed in $[0.95, 1.05]$ and $N = 10^3$. Diagrams are plotted as a function of the measurement rate $\alpha = M/N$ and the sparsity level ρ for the offline **(top row)** and online **(bottow row)** algorithms, for increasing number of available samples P. The pink line $\alpha_{\min}$ marks the strict lower bound on the number of measurements necessary for reconstruction. Results are similar to the diagrams in terms of errors on the signal $\hat{X}$ of Figure 6.2.

6.2.3 Online results

The online algorithms include supplementary operations to update the effective prior on the calibration variable from one step to the next. In this setting the recursion is

$$p_{s_\mu}^{(k+1)}(s_\mu) = \frac{1}{\mathcal{Z}_{\text{out}}^{(k+1)}} \int \mathrm{d}z_{(k),\mu}\, \mathcal{N}(\underline{y}_{(k),\mu}; z_{(k),\mu}/s_\mu, \Delta)\mathcal{N}(z_{(k),\mu}; w_{(k),\mu}, V_{(k),\mu}) p_{s_\mu}^{(k)}(s_\mu), \tag{6.107}$$

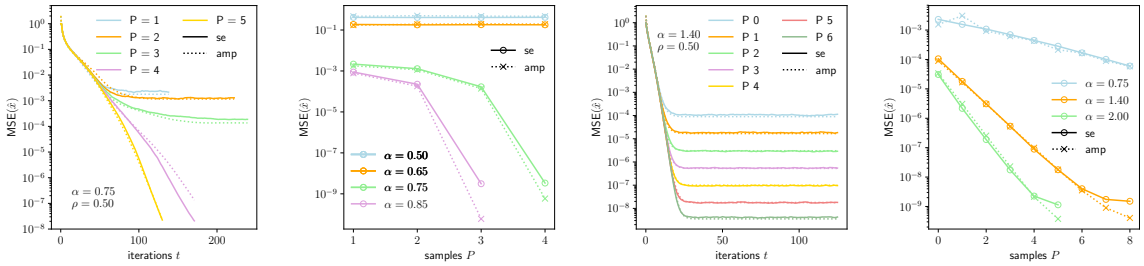
$$\propto s_\mu\, e^{-\frac{(s_\mu - \nu_{(k),\mu})^2}{2\delta_{(k),\mu}}} p_{s_\mu}^{(k)}(s_\mu) \tag{6.108}$$

so that

$$p_{s_\mu}^{(P)}(s_\mu) \propto s_\mu^P\, e^{-\frac{(s-\Lambda_{(P),\mu})^2}{2\Sigma_{(P),\mu}}} \quad \text{with} \quad \begin{cases} \Sigma_{(P),\mu}^{-1} = \sum_{k=1}^{P} \delta_{(k),\mu}^{-1} \\ \Lambda_{(P),\mu} = \Sigma_{(P),\mu} \left(\sum_{k=1}^{P} \delta_{(k),\mu}^{-1} \nu_{(k),\mu} \right) , \end{cases} \tag{6.109}$$

yielding a posterior on the calibration variable at step P with an identical form to the posterior of the offline algorithm (albeit with parameters computed differently). Therefore $\hat{s}_\mu$ and C_μ^s can still be computed analytically following (6.100) and (6.101). We provide a pseudo-code for the online Cal-AMP in Algorithm 10 and a pseudo-code for online SE in Algorithm 9.

On the bottom rows of Figure 6.2 and 6.3 we plot phase diagrams obtained with online Cal-AMP. Compared to the offline diagrams we find that the reconstruction requires more samples to achieve comparable levels of accuracy. On Figure 6.4c and 6.4d we check the consistency of the Cal-AMP and SE fixed points in terms of MSE on $\underline{\underline{X}}$.



(a) Offline **(b)** Offline **(c)** Online **(d)** Online

Figure 6.4: Comparison of mean squared errors on the signal $\underline{\underline{X}}$ obtained with Cal-AMP and predicted by SE for the gain calibration problem with calibration variables uniformly distributed in $[0.95, 1.05]$ and $N = 10^4$. Along the iterations of the algorithm and comparing fixed points we find a very good agreements of the two procedures. Nevertheless, the Monte Carlo integration in the SE does not allow for great numerical precision for low errors and we start seeing discrepancies between SE and AMP for errors below 10^{-8}.

Algorithm 9 Online Gain Calibration State Evolution

Input: matrix $\underline{\underline{Y}} \in \mathbb{R}^{M\times P}$ and matrix $\underline{\underline{W}} \in \mathbb{R}^{M\times N}$:
Initialize: $t = 0$,
$\forall \mu = 1 \cdots N_{\text{MC}}$
$s^0_\mu \sim p_{s^0}(s^0_\mu)$
$\Lambda_{0,\mu} = 0\,, \quad \Sigma_{0,\mu} = 0$

for $k = 1 \cdots P$ **do**
Initialize: $t = 0$, $m^{(0)}_{(k)} = 0$, $V^{(0)}_{(k)} = q_0 = \rho$
repeat
1) Draw Monte Carlo samples for update of $\hat{m}$ (6.92) and r (6.90)
$\forall \mu = 1 \cdots N_{\text{MC}}$

$$z_{\mu,k}, \omega_{\mu,k} \sim \mathcal{N}(z_{\mu,k}, \omega_{\mu,k}; \underline{0}, \underline{\underline{Q}}^{(t)}_{(k)})$$
$$\epsilon \sim p_\epsilon(\epsilon)$$
$$y_{\mu,k} = g^0(z_{\mu,k}, s^0_\mu, \epsilon)$$

with $\underline{\underline{Q}}^{(t)}_{(k)} = \begin{bmatrix} q_0 & m^{(t)}_{(k)} \\ m^{(t)}_{(k)} & m^{(t)}_{(k)} \end{bmatrix}$

2) Following step (2) of Algorithm 10 with
- samples of previous steps $\{y_{\mu,l}, z_{\mu,l}, \omega_{\mu,l}\}_{l\leq k-1}$ at convergence
- current $y_{\mu,k}, z_{\mu,k}, \omega_{\mu,k}$
- current $V^{(t)}_{(k)}$

2.1) Update $\Lambda_{k,\mu}, \Sigma_{k,\mu}$
2.2) Compute $\hat{s}_\mu$
2.3) Compute $g_{\text{out}\,\mu,k}$

3) Update $r_{(k)}$ and $\hat{m}_{(k)}$

$$r^{(t)}_{(k)} = \frac{1}{N_{\text{MC}}} \sum_{\mu=1}^{N_{\text{MC}}} \left(\hat{s}^{(t)}_\mu\right)^2 \qquad \hat{m}^{(t)}_{(k)} = \frac{1}{N_{\text{MC}}} \sum_{\mu=1}^{N_{\text{MC}}} \left(g_{\text{out}\,\mu,k}\right)^2$$

4) Update $m^{(t+1)}_{(k)}$ by numerical integration using (6.91), and $V^{(t+1)}_{(k)} = q_0 - m^{(t+1)}_{(k)}$.
$t = t + 1$

until convergence
end for
Output: time series $\{V^{(t)}_{(k)}, m^{(t)}_{(k)}, \hat{m}^{(t)}_{(k)}\ ;\ t = 1 \cdots t_{\max}\}_{k=1}^{P}$

Algorithm 10 Online Gain Calibration Approximate Message Passing

Input: matrix $\underline{Y} \in \mathbb{R}^{M\times P}$ and matrix $\underline{\underline{W}} \in \mathbb{R}^{M\times N}$:
Initialize: $\Lambda_{0,\mu} = 0$, $\Sigma_{0,\mu} = 0$,
for $k = 1 \cdots P$: **do**
Initialize: $\hat{x}_{i,k}$, $C^x_{i,k}$ $\quad \forall i$ and $g_{\text{out}\,\mu,k}$, $\partial_\omega g_{\text{out}\,\mu,k}$ $\quad \forall \mu$
repeat
1) Estimate mean and variance of $\underline{z}_{(k)}$ given current $\underline{\hat{x}}_{(k)}$

$$V^{(t)}_{\mu,k} = \sum_{i=1}^{N} W^2_{\mu i} C^{x\,(t)}_i \tag{6.110}$$

$$\omega^{(t)}_{\mu,k} = \sum_{i=1}^{N} W_{\mu i}\hat{x}^{(t)}_{i,k} - \sum_{i=1}^{N} W^2_{\mu i} C^{x\,(t)}_{i,k} g_{\text{out}}{}^{(t-1)}_{\mu,k} \tag{6.111}$$

2) Exploit current $\underline{y}_{(k)}$ and inherited $\Lambda_{k-1,\mu}, \Sigma_{k-1,\mu}$
2.1) Update recursion parameters given $\underline{y}_{(k)}$

$$\delta^{(t)\,-1}_{\mu,k} = y^2_{\mu,k}/(V^{(t)}_{\mu,k} + \Delta) \tag{6.112}$$

$$\nu^{(t)}_{\mu,k} = \delta_{\mu,k} y_{\mu,k} \omega^{(t)}_{\mu,k}/(V^{(t)}_{\mu,k} + \Delta) \tag{6.113}$$

$$\Sigma^{(t)}_{k,\mu} = \Sigma_{k-1,\mu} + \delta^{(t)}_{\mu,k} \tag{6.114}$$

$$\Lambda^{(t)}_{k,\mu} = \Sigma_{k,\mu}\left(\Sigma_{k-1,\mu}\Lambda_{k-1,\mu} + \delta^{(t)}_{\mu,k}\nu^{(t)}_{\mu,k}\right) \tag{6.115}$$

2.2) Update estimates $\hat{s}$ and C^s

$$\hat{s}^{(t)}_\mu = \frac{\mathcal{I}(k+1, \Lambda_{k,\mu}, \Sigma_{k,\mu}, a, b)}{\mathcal{I}(k, \Lambda_{k,\mu}, \Sigma_{k,\mu}, a, b)} \tag{6.116}$$

$$C^{s\,(t)}_{\mu} = \frac{\mathcal{I}(k+2, \Lambda_{k,\mu}, \Sigma_{k,\mu}, a, b)}{\mathcal{I}(k, \Lambda_{k,\mu}, \Sigma_{k,\mu}, a, b)} \tag{6.117}$$

2.3) Update $\underline{g_{\text{out}}}$ and $\partial_\omega \underline{\underline{g_{\text{out}}}}$

$$\partial_\omega g_{\text{out}}{}^{(t)}_{\mu,k} = C^{s\,(t)}_{\mu} y^2_{\mu,k}/(V^{(t)}_{\mu,k} + \Delta)^2 - (V^{(t)}_{\mu,k} + \Delta)^{-1} \tag{6.118}$$

$$g_{\text{out}}{}^{(t)}_{\mu,k} = (\hat{s}^{(t)}_\mu y_{\mu,k} - \omega^{(t)}_{\mu,k})/(V^{(t)}_{\mu,k} + \Delta), \tag{6.119}$$

3) Estimate mean and variance of $\underline{x}$ given current optimal $\underline{z}$

$$\sigma^{(t)}_{i,k} = \left(-\sum_{\mu=1}^{M} W^2_{\mu i}\partial_\omega g_{\text{out}}{}^{(t)}_{\mu,k}\right)^{-1} \tag{6.120}$$

$$\lambda^{(t)}_{i,k} = \hat{x}^{(t)}_{i,k} + \sigma^{(t)}_{i,k}\left(\sum_{\mu=1}^{M} W_{\mu i} g_{\text{out}}{}^{(t)}_{\mu,k}\right) \tag{6.121}$$

4) Estimate of mean and variance of $\underline{x}$ augmented of the information about the prior

$$C^x_{i,k}{}^{(t+1)} = f^x_2(\lambda^{(t)}_{i,k}, \sigma^{(t)}_{i,k}) \tag{6.122}$$

$$\hat{x}^{(t+1)}_{i,k} = f^x_1(\lambda^{(t)}_{i,k}, \sigma^{(t)}_{i,k}) \tag{6.123}$$

$t = t + 1$

until convergence
end for
Output: Estimates and variances $\underline{\hat{x}}_i$, $\underline{\underline{C^x}}_i$, $\hat{s}_\mu$, C^s_μ

6.3 Matrix factorization model and multi-layer networks

We now return to our original problem of interest, which is to find a tractable setting to perform and analyze high-rank matrix factorization, a necessary milestone on the road to the characterization of learning in multi-layer neural networks. We identify a constrained formulation of matrix factorization that can be approached as a multi-layer GLM including calibration variables. We demonstrate a principle of derivation of the AMP algorithm for multi-layer GLM and deduce from it the AMP algorithm for the constrained matrix factorization.

6.3.1 Constrained matrix factorization

We consider the problem of recovering matrices $\underline{\underline{W}}_0 \in \mathbb{R}^{Q\times N}$ and $\underline{\underline{X}}_0 \in \mathbb{R}^{N\times P}$ after observation of their product through a noisy (non-linear) channel while introducing a constraint:

$$\underline{\underline{Y}} = g_0(\underline{\underline{W}}_0\,\underline{\underline{X}}_0;\,\underline{\underline{\epsilon}}),\ \text{s.t.}\ \underline{\underline{W}}_0 = \underline{\underline{\Phi}}^1\,\underline{\underline{S}}_0\,\underline{\underline{\Phi}}^0. \tag{6.124}$$

As usual $g_0(\cdot)$ stands for the channel and incorporates a noise $\underline{\underline{\epsilon}} \in \mathbb{R}^{Q\times P}$ drawn from a simple distribution $p_\epsilon(\underline{\underline{\epsilon}})$. The matrices $\underline{\underline{\Phi}}^0 \in \mathbb{R}^{M\times N}$ and $\underline{\underline{\Phi}}^1 \in \mathbb{R}^{Q\times M}$ of the constrained form of $\underline{\underline{W}}_0$ are known, while the matrix $\underline{\underline{S}}_0 \in \mathbb{R}^{M\times M}$ is diagonal and encloses the degrees of freedom of $\underline{\underline{W}}_0$. Considering the teacher-student matched setting, the posterior joint distribution we are going to be interested in is

$$p(\underline{\underline{X}},\underline{\underline{S}}|\underline{\underline{\Phi}}^0,\underline{\underline{\Phi}}^1,\underline{\underline{Y}}) = \frac{1}{\mathcal{Z}}\int \mathrm{d}\underline{\underline{U}}\; p^2_{\text{out}}(\underline{\underline{Y}}|\underline{\underline{\Phi}}^1\underline{\underline{U}})p^2_{\text{out}}(\underline{\underline{U}}|\underline{\underline{\Phi}}^0\underline{\underline{X}},\underline{\underline{s}})p_x(\underline{\underline{X}})p_x(\underline{\underline{S}}) \tag{6.125}$$

with the distributions of the right hand-side assumed to factorize over their inputs and the corresponding factor graph represented in Figure 6.5a. In the following we will assume that N, M and Q are large but maintain fixed ratios $\alpha_0 = M/N$ and $\alpha_1 = Q/N$. Thus we are considering extensive rank matrix factorization, while easing the inference problem by reducing the number of free variables in one of the factors.

The advantage of the chosen constrained model is that it can be interpreted as the combination of two GLM models:

$$\underline{\underline{Y}} = g_{0,2}(\underline{\underline{\Phi}}^1\,\underline{\underline{U}};\,\underline{\underline{\epsilon}}) \qquad \textit{usual GLM with prior on signal } \underline{\underline{U}} \textit{ coming from the first layer,}$$

$$\underline{\underline{U}} = g_{0,1}(\underline{\underline{\Phi}}^0\,\underline{\underline{X}};\,\underline{\underline{S}}) \qquad \textit{GLM with calibration variables.}$$

The inference of $\underline{\underline{X}}$ and $\underline{\underline{S}}$ can therefore be tackled using the recent progresses around multi-layer models. In the following we start by recalling and straightforwardly extending the ML-AMP algorithm [84] introduced at $P = 1$ to arbitrary P. In a second step we re-introduce the calibration variables to obtain the AMP algorithm for the constrained matrix factorization under scrutiny.

6.3.2 Multi-layer vectorized AMP

The derivation of the multi-layer AMP on vector variables follows identical steps to the derivation of the single layer presented in Section 6.1.1 starting with a larger collection of messages. Without conducting the lengthy procedure, one can form an intuition for the resulting algorithm starting from the single-layer AMP. We discuss here this intuition focusing on the 2-layer case.

We are now considering the factor graph of Figure 6.5b and the resulting Algorithm 11. Compared to the single-layer case, an interface with a set of hidden variables $\underline{u}_\mu$ is inserted between the signals $\underline{x}_i$ and the observations $\underline{y}_a$. In the neighborhood of the inputs $\underline{x}_i$ the factor graph is identical to the single-layer and the input functions can be defined from a normalization partition identical to (6.28),

$$\mathcal{Z}^x(\underline{\lambda}_i,\underline{\underline{\sigma}}_i) = \int \mathrm{d}\underline{x}_i\; p_x(\underline{x}_i)e^{-\frac{1}{2}(\underline{x}_i-\underline{\lambda}_i)^\top\underline{\underline{\sigma}}_i^{-1}(\underline{x}_i-\underline{\lambda}_i)}, \tag{6.126}$$

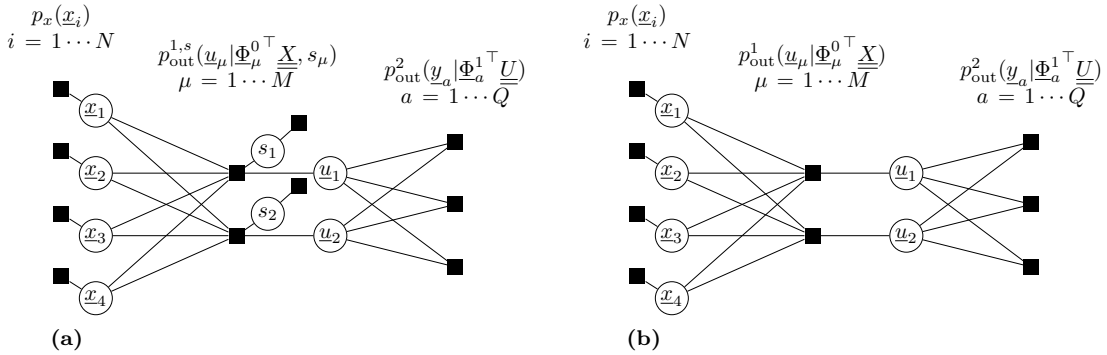


Figure 6.5: (a) Factor graph representation of the proposed constrained matrix factorization setting corresponding to the joint distribution (6.125) with factorization on N, M and Q and with the notation $s_\mu = S_{\mu\mu}$. **(b)** Factor graph representation of a generic 2-layer GLM with vector variables.

yielding updates (6.149)-(6.150). Similarly, the neighborhood of the observations $\underline{y}_a$ is also unchanged and the updates (6.139) and (6.140) are following from the definition of

$$\mathcal{Z}^{y}_{\text{out}}(\underline{\omega}^2_a, \underline{\underline{V}}^2_a) = \int \mathrm{d}\underline{z}_a \;\; p^2_{\text{out}}(\underline{y}_a|\underline{z}_a)\; e^{\frac{1}{2}\left(\underline{z}_a - \underline{\omega}^2_a\right)^\top \underline{\underline{V}}^{2\,-1}_a \left(\underline{z}_a - \underline{\omega}^2_a\right)}, \tag{6.127}$$

identical to the single layer (6.21). At the interface however, the variables $\underline{u}_\mu$ play the role of outputs for the first GLM and of inputs for the second GLM, which translates into a normalization partition function of mixed form

$$\mathcal{Z}^{u}_{\text{out}}(\underline{\omega}^1_\mu, \underline{\underline{V}}^1_\mu, \underline{\lambda}^1_\mu, \underline{\underline{\sigma}}^1_\mu) = \int \mathrm{d}\underline{z}_\mu \;\; \int \mathrm{d}\underline{u}_\mu \; p^1_{\text{out}}(\underline{u}_\mu|\underline{z}_\mu) \times \tag{6.128}$$
$$e^{-\frac{1}{2}\left(\underline{u}_\mu - \underline{\lambda}^1_\mu\right)^\top \underline{\underline{\sigma}}^{1\,-1}_\mu \left(\underline{u}_\mu - \underline{\lambda}^1_\mu\right)} e^{\frac{1}{2}\left(\underline{z}_\mu - \underline{\omega}^1_\mu\right)^\top \underline{\underline{V}}^{1\,-1}_\mu \left(\underline{z}_\mu - \underline{\omega}^1_\mu\right)}.$$

Updates (6.145) and (6.146) are obtained by considering that the second layer acts as an effective channel for the first layer, i.e. from the normalization interpreted as

$$\mathcal{Z}^{u}_{\text{out}}(\underline{\omega}^1_\mu, \underline{\underline{V}}^1_\mu, \underline{\lambda}^1_\mu, \underline{\underline{\sigma}}^1_\mu) = \int \mathrm{d}\underline{z}_\mu \;\; p^{\text{eff}}_{\text{out}}(\underline{z}_\mu)\; e^{\frac{1}{2}\left(\underline{z}_\mu - \underline{\omega}^1_\mu\right)^\top \underline{\underline{V}}^{1\,-1}_\mu \left(\underline{z}_\mu - \underline{\omega}^1_\mu\right)}. \tag{6.129}$$

Finally, update equations (6.151) and (6.152) are in turn derived considering the first layer defines an effective prior for the hidden variables and the normalization as

$$\mathcal{Z}^{u}_{\text{out}} = \int \mathrm{d}\underline{u}_\mu \;\; p^{\text{eff}}_{u}(\underline{u})\; e^{-\frac{1}{2}\left(\underline{u}_\mu - \underline{\lambda}^1_\mu\right)^\top \underline{\underline{\sigma}}^{1\,-1}_\mu \left(\underline{u}_\mu - \underline{\lambda}^1_\mu\right)}. \tag{6.130}$$

The rest of the algorithm updates follows as usual from the self-consistency between the different variables introduced as they correspond to different parametrization of the same marginals. The schedule of updates and the time indexing reported in Algorithm 11 results from the entire derivation starting from the BP messages. The generalization of the algorithm to an arbitrary number of layers is easily obtained repeating the heuristic arguments presented here. Lastly, note that the corresponding State Evolution equations can be derived following identical steps as presented in Section 6.1.2.

6.3.3 AMP for constrained matrix factorization

We have already argued that the constrained matrix factorization problem of interest here (6.124) can be interpreted as the combination of two GLM models. It is in fact covered by the two-layer AMP algorithm presented in the previous Section, for the interface channel

$$p^1_{\text{out}}(\underline{u}_\mu|\underline{z}^1_\mu) = \int \mathrm{d}s_\mu \; p^{1,s}_{\text{out}}(\underline{u}_\mu|\underline{z}^1_\mu) p_s(s_\mu) = \int \mathrm{d}s_\mu \; \delta(\underline{u}_\mu - s_\mu \underline{z}_\mu) p_s(s_\mu), \tag{6.131}$$

and to which we should add the equations of estimation of the calibration variables. The interface partition function is

$$\mathcal{Z}^u_{\text{out}} = \int d\underline{u} \int d\underline{z}^1_\mu p^1_{\text{out}}(\underline{z}^1_\mu|\underline{u})\mathcal{N}(\underline{z}^1_\mu; \underline{\lambda}^1_\mu, \underline{\underline{\sigma}}^1_\mu)\mathcal{N}(\underline{u}; \underline{\omega}^1_\mu, \underline{\underline{V}}^1_\mu) \tag{6.132}$$

$$= \int d\underline{u} \int d\underline{z}^1_\mu \left(\int ds_\mu p_s(s_\mu)\delta(\underline{z}^1_m u - s_\mu \underline{u}) \right) \mathcal{N}(\underline{z}^1_\mu; \underline{\lambda}^1_\mu, \underline{\underline{\sigma}}^1_\mu)\mathcal{N}(\underline{u}; \underline{\omega}^1_\mu, \underline{\underline{V}}^1_\mu) \tag{6.133}$$

$$= \int d\underline{u} \int ds_\mu p_s(s_\mu)\mathcal{N}(s_\mu \underline{u}; \underline{\lambda}^1_\mu, \underline{\underline{\sigma}}^1_\mu)\mathcal{N}(\underline{u}; \underline{\omega}^1_\mu, \underline{\underline{V}}^1_\mu) \tag{6.134}$$

$$= \int ds_\mu p_s(s_\mu)\mathcal{N}(\underline{\lambda}^1_\mu; s_\mu \underline{\omega}^1_\mu, \underline{\underline{\sigma}}^1_\mu + s^2_\mu \underline{\underline{V}}^1_\mu), \tag{6.135}$$

where we changed the convention by adding the normalizations of the Gaussian measures compared to (6.128) (an operation that does not affect the definition of the update functions). For the calibration variables, the approximate posterior provided by AMP is therefore

$$m^s_\mu(s_\mu) = \frac{1}{\mathcal{Z}^u_{\text{out}}} p_s(s_\mu)\mathcal{N}(\underline{\lambda}^1_\mu; s_\mu \underline{\omega}^1_\mu, \underline{\underline{\sigma}}^1_\mu + s^2_\mu \underline{\underline{V}}^1_\mu). \tag{6.136}$$

The update functions derived from $\mathcal{Z}^u_{\text{out}}$ can be written in terms of expectations over this posterior. They are given in Appendix B.

Algorithm 11 Generalized Approximate Message Passing for the 2-layer GLM

Input: matrix $\underline{\underline{Y}} \in \mathbb{R}^{M\times P}$ and matrices $\underline{\underline{\Phi}}^0 \in \mathbb{R}^{M\times N}$, $\underline{\underline{\Phi}}^1 \in \mathbb{R}^{Q\times M}$:
Initialize: $\underline{\hat{x}}_i$, $\underline{\underline{C^x}}_i \quad \forall i$, $\underline{\hat{u}}_\mu$, $\underline{\underline{C^u}}_\mu$, $\underline{g^1_{\text{out}}}_\mu$, $\partial_{\underline{\omega}}\underline{\underline{g^1_{\text{out}}}}_\mu \quad \forall \mu$, $\underline{g^2_{\text{out}}}_a$, $\partial_{\underline{\omega}}\underline{\underline{g^2_{\text{out}}}}_a \quad \forall \mu$ and $t = 0$.

repeat

1) Update auxiliary variables of second layer:

$$\underline{\omega}_a^{2\,(t)} = \sum_\mu \frac{\Phi^1_{a\mu}}{\sqrt{N}} \hat{\underline{u}}_\mu^{(t)} - \sum_\mu \frac{(\Phi^1_{a\mu})^2}{N} \underline{\underline{\sigma}}_\mu^{1\,(t-1)^{-1}} \underline{\underline{C^u}}_\mu^{\,(t)} \underline{\underline{\sigma}}_\mu^{1\,(t-1)} \underline{g^2_{\text{out}}}_a^{\,(t-1)} \tag{6.137}$$

$$\underline{\underline{V}}_a^{2\,(t)} = \sum_\mu \frac{(\Phi^1_{a\mu})^2}{N} \underline{\underline{C^u}}_\mu^{\,(t)} \tag{6.138}$$

$$\underline{g^2_{\text{out}}}_a^{\,(t)} = \underline{g^2_{\text{out}}}(\underline{y}_a, \underline{\omega}_a^{2\,(t)}, \underline{\underline{V}}_a^{2\,(t)}) \tag{6.139}$$

$$\partial_{\underline{\omega}}\underline{\underline{g^2_{\text{out}}}}_a^{\,(t)} = \partial_{\underline{\omega}}\underline{\underline{g^2_{\text{out}}}}(\underline{y}_a, \underline{\omega}_a^{2\,(t)}, \underline{\underline{V}}_a^{2\,(t)}) \tag{6.140}$$

$$\underline{\lambda}_\mu^{1\,(t)} = \underline{\underline{\sigma}}_\mu^1 \left(\sum_a \frac{\Phi^1_{a\mu}}{\sqrt{N}} \underline{g^2_{\text{out}}}_a^{\,(t)} - \frac{(\Phi^1_{a\mu})^2}{N} \partial_{\underline{\omega}}\underline{\underline{g^2_{\text{out}}}}_a^{\,(t)} \hat{\underline{u}}_\mu \right) \tag{6.141}$$

$$\underline{\underline{\sigma}}_\mu^{1\,(t)} = \left(-\sum_a \frac{(\Phi^1_{a\mu})^2}{N} \partial_{\underline{\omega}}\underline{g^2_{\text{out}}}_a^{\,(t)} \right)^{-1} \tag{6.142}$$

1) Update auxiliary variables of first layer:

$$\underline{\omega}_\mu^{1\,(t)} = \sum_i \frac{\Phi^0_{\mu i}}{\sqrt{N}} \hat{\underline{x}}_i^{(t)} - \sum_i \frac{(\Phi^0_{\mu i})^2}{N} \underline{\underline{\sigma}}_i^{0\,(t-1)^{-1}} \underline{\underline{C^x}}_i \underline{\underline{\sigma}}_i^{0\,(t-1)} \underline{g^1_{\text{out}}}_\mu^{\,(t-1)} \tag{6.143}$$

$$\underline{\underline{V}}_\mu^{1\,(t)} = \sum_i \frac{(\Phi^0_{\mu i})^2}{N} \underline{\underline{C^x}}_i^{\,(t)} \tag{6.144}$$

$$\underline{g^1_{\text{out}}}_\mu^{\,(t)} = \underline{g^1_{\text{out}}}(\underline{\omega}_\mu^{1\,(t)}, \underline{\underline{V}}_\mu^{1\,(t)}, \underline{\lambda}_\mu^{1\,(t)}, \underline{\underline{\sigma}}_\mu^{1\,(t)}) \tag{6.145}$$

$$\partial_{\underline{\omega}}\underline{\underline{g^1_{\text{out}}}}_\mu^{\,(t)} = \partial_{\underline{\omega}}\underline{\underline{g^1_{\text{out}}}}(\underline{\omega}_\mu^{1\,(t)}, \underline{\underline{V}}_\mu^{1\,(t)}, \underline{\lambda}_\mu^{1\,(t)}, \underline{\underline{\sigma}}_\mu^{1\,(t)}) \tag{6.146}$$

$$\underline{\underline{\sigma}}_i^{0\,(t)} = \left(-\sum_i \frac{(\Phi^0_{\mu i})^2}{N} \partial_{\underline{\omega}}\underline{\underline{g^1_{\text{out}}}}_\mu^{\,(t)} \right)^{-1} \tag{6.147}$$

$$\underline{\lambda}_i^0 = \underline{\underline{\sigma}}_i^0 \left(\sum_\mu \frac{\Phi^0_{\mu i}}{\sqrt{N}} \underline{g^1_{\text{out}}}_\mu^{\,(t)} - \frac{(\Phi^0_{\mu i})^2}{N} \partial_{\underline{\omega}}\underline{\underline{g^1_{\text{out}}}}_\mu^{\,(t)} \hat{\underline{x}}_i \right) \tag{6.148}$$

3) Update means and variances of variables of both layers, $\underline{x}$ and $\underline{u}$:

$$\hat{\underline{x}}_i^{(t+1)} = f_1^x \left(\underline{\lambda}_i^{0\,(t)}, \underline{\underline{\sigma}}_i^{0\,(t)} \right) \tag{6.149}$$

$$\underline{\underline{C^x}}_i^{\,(t+1)} = f_2^x \left(\underline{\lambda}_i^{0\,(t)}, \underline{\underline{\sigma}}_i^{0\,(t)} \right) \tag{6.150}$$

$$\hat{\underline{u}}_\mu^{(t+1)} = f_1^u \left(\underline{\omega}_\mu^{1\,(t)}, \underline{\underline{V}}_\mu^{1\,(t)}, \underline{\lambda}_\mu^{1\,(t)}, \underline{\underline{\sigma}}_\mu^{1\,(t)} \right) \tag{6.151}$$

$$\underline{\underline{C^u}}_\mu^{\,(t+1)} = f_2^u \left(\underline{\omega}_\mu^{1\,(t)}, \underline{\underline{V}}_\mu^{1\,(t)}, \underline{\lambda}_\mu^{1\,(t)}, \underline{\underline{\sigma}}_\mu^{1\,(t)} \right) \tag{6.152}$$

$t = t + 1$

until convergence

Output: signal estimate $\hat{\underline{x}}_1 \in \mathbb{R}^N$, and estimated covariance $\underline{\underline{C^x}}_1 \in \mathbb{R}^{N\times N}$

6.4 Towards the analysis of learning in multi-layer neural networks

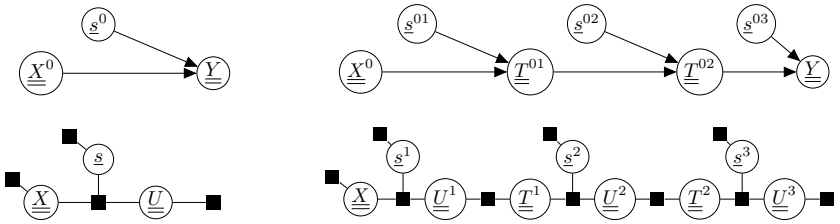
The presented elementary matrix factorization can be used in turn as a building block to study learning in multi-layer neural networks under the teacher-student paradigm. To conclude this Chapter we discuss how we believe this study could be done.

The statistical model we previously investigated as an example of matrix factorization (6.124) can be interpreted as a very simple feed forward generative model used in modern machine learning. Namely, a latent variable with a separable prior $\underline{x} \in \mathbb{R}^N$ is transformed by a stochastic single layer neural network with weight matrix $\underline{\underline{W}}$ (parametrized by $\underline{s}$) to create a data point $\underline{y} \in \mathbb{R}^Q$. Usually, such a model is trained to fit an arbitrary (real) data set by a gradient descent following either the Generative Adversarial Networks [50] interpretation and the adversarial objective or the Variational Autoencoder interpretation [71, 120] and the maximum likelihood objective. In the constrained setting we have described, with a reduced number of degrees of freedom of the weight matrix $\underline{\underline{W}}$, the AMP algorithm is also a learning algorithm returning the mean a posteriori $\underline{\underline{\hat{W}}} = \underline{\underline{\Phi}}^1 \underline{\underline{\hat{S}}} \underline{\underline{\Phi}}^0$ as appropriate network parameters.

This interpretation is interesting for several reasons. On Figure 6.6a, we represent the graphical model for the single-layer teacher generative model and the factor graph for the AMP inference of the student (only factorized at the level of the matrices). In the teacher-student scenario, the theoretical performance of AMP can be tracked on a single instance by comparing the reconstructed $\underline{\underline{\hat{W}}}$ to the teacher $\underline{\underline{W}}_0$ and cross-checked by the State Evolution. These tools will thereby allow to conduct a theoretical analysis of unsupervised learning in a simple model. Moreover, AMP can also be a practical learning algorithm on arbitrary data (not generated from a teacher) and should be compared with other learning techniques. While it is likely that a message passing implementation of learning would be cumbersome compared to SGD, it could be more efficient in various ways. For instance, AMP may need less samples than gradient descent to reach a similar quality of training.

Furthermore, the case study can be extended to deep generative neural networks with constrained weight matrices by stacking the elementary model as represented on Figure 6.6b. The juxtaposition of models can again be interpreted as a special case of the multi-layer vectorized GLM and the inference can be performed by extending the 2-layer vectorized AMP Algorithm 11 to an arbitrary number of layers. The graphical representations (see Figure 6.6b) of the teacher generative model and the posterior distribution now include variables for the hidden layers noted $\underline{\underline{T}}$, which are new interface variables in the AMP inference.

Finally, the considered model comprises deep supervised learning as a special case. In fact, by fixing the values of the inputs $\underline{\underline{X}}$ to the value of a training set with the corresponding outputs $\underline{\underline{Y}}$, the inference problem is only simpler. It consists in learning solely the weights of the deep neural network, corresponding to the supervised setting. Such scenario should therefore allow to replicate the statistical physics analyses of learning in simple architectures (perceptrons and committee machines) in deep networks. In particular it could enable the computation of learning curves predicting optimal generalization errors in the multi-layer case, albeit in the constrained regime of learning, with only the singular values of the weights as adjustable parameters. As traditional generalization bounds relying on worst case guarantees are found irrelevant in deep learning, the proposed case study appears as an interesting preliminary test to determine whether the statistical physics approach could be more informative here. Potentially, it could provide a scale to measure the performance of the currently popular training algorithms that were introduced empirically, such as SGD.



(a) single-layer **(b)** multi-layer

Figure 6.6: Teacher-student models for unsupervised learning **Top row:** Directed graphical model representations of the single-layer and multi-layer teacher generative models for synthetic data set $\underline{\underline{Y}}$. **Bottom row:** Corresponding factor graphs for inference of the latent representations and network parameters.

Conclusion and outlook

In this dissertation, we used different mean-field free energies and algorithms to answer various questions around the training of neural networks. In Chapter 4, we showed that the TAP free energy and the approximate message passing algorithms are of practical interest to train Boltzmann machines. They can also be leveraged to exploit learned RBMs as priors in Bayesian problems. A straightforward perspective of these results is to now consider, through the same lense, the feed forward models used in unsupervised learning. Here also, mean-field methods could yield alternative algorithms competitive with the state-of-the art. In Chapter 5, we used a replica computation to test the relevance of information theory to explain the generalization paradox in deep learning. With this strategy we were able to probe prototypes of learning experiments with large networks and non-linearities. Our findings both confirmed and confounded some of the previous claims in the literature, finally pointing out the caveats of the proposed approach. Chapter 6 is in itself a perspective. We identified a direction that could lead to an analysis of learning in multi-layer networks along the lines of the original statistical mechanics of the perceptron. We validated the first step of this proposed program by deriving the State Evolution corresponding to the Cal-AMP algorithm, which allows to infer parameters in the channel of a GLM.

On the edge of validity

Nonetheless, we have also touched upon the limitations of the mean-field approach. In Chapter 5, we had to consider a restricted setting of learning in order to guarantee the accuracy of the replica formula of the entropy. Only the spectrum of the weight matrices was learned, which effectively reduced the number of degrees of freedom. A similar restriction is also included in the proposed direction to study multi-layer Bayesian learning in Chapter 6. Additionally, for both of these analyses, we can only consider synthetic data sets with specific distributions.

More generally, the temptation to apply abusively results from one field to the other can be a dangerous pitfall of the interdisciplinary approach. We could mention here the characterization of the dynamics of optimization, an open question that we did not tackle in this dissertation. While physicists have extensively studied Langevin dynamics with Gaussian white noise, the continuous time limit of SGD is unfortunately not an equivalent in the general case. While some works attempt to draw insights from this analogy using strong assumptions (e.g. [25, 65]), others seek precisely to understand the differences between the two dynamics in neural networks optimization (e.g. [8, 138]). Alternatively, [82] gained insights on the dynamics of gradient descent (and Langevin) in a different high-dimensional non-convex optimization problem inspired by the physics of spin glasses. This last work illustrates the modelling tradition of theoretical physics. One can learn from models a priori far from the exact neural networks desired, but that retain some key properties, while being amenable to theoretical characterization. Another example of this approach is [155].

Thus the mean-field approach will certainly not provide complete answers to the still numerous puzzles on the way towards a deep learning theory. Yet, combined with the modelling tradition of physics, it will still probably provide many more interesting intuitions. More specifically, the different contributions presented in this dissertation, point us towards a promising set up to study the impact of structure in data, which we propose as a final outlook.

One outlook

In the complicated story behind generalization, more and more works are trying to disentangle the role played by optimization algorithms (e.g. [132]) and regularization methods (e.g. [127]), but the role played by data structure, while certainly important [161], is harder to probe. In the teacher-student scenario, one considers learning on synthetic datasets. In the classical literature, data points are typically distributed as random white noise. This choice is merely the simplest to

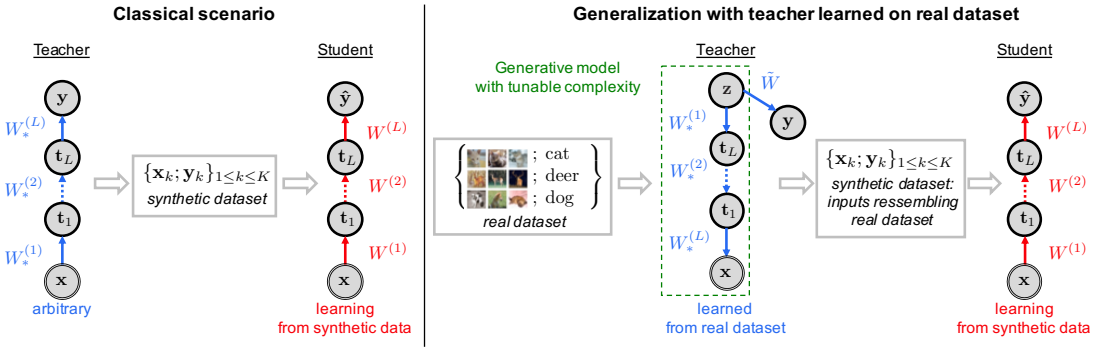


Figure 6.7: Left: The traditional teacher-student scenario consists in trying to recover the parameters W_* of a teacher from a synthetic dataset generated by the latter used to train a student with same architecture. **Right:** To study the role of structure in natural data (e.g. images), one could study the training of a student network on a dataset remaining synthetic but generated by a teacher of controlled complexity trained on real data with state-of-the art technique in unsupervised learning.

allow for analytical computations of interesting observables such as learning curves. An exciting direction is instead to introduce some tunable structure in the synthetic data. This strategy was already adopted in [69, 26, 28]. These works used mixture of Gaussians or parametric manifolds, which can be more or less clustered, to derive theoretical learning properties as a function of the data properties using mean-field methods.

In Chapter 5, we further generalized the classical matched teacher-student scenario to include generative models. Although it was not the case in the experiments run in this Chapter, these generative models can be trained following state-of-the art unsupervised learning methods to produce synthetic datasets looking like real ones (images, sounds, texts etc.), as sketched on Figure 6.7. Following ideas developed in Chapters 4 and 5, mean-field inference should remain efficient in these models during learning with usual algorithms such as SGD. Additionally, the perspectives of Chapter 6 envisage also the analysis of a Bayesian learning of the weights using message passing algorithms and State Evolutions, both in the generative model and in the student network.

The combination of these different ideas pushes the idea of modeling learning tasks in an exciting direction. The progress in mean-field inference in compositional models will possibly allow a theoretical characterization of these teacher-student settings. Even if not, the empirical study of tunable sophisticated scenarios mimicking real learning tasks should improve our understanding, among other points, of the role of data structures in deep learning.

Appendix

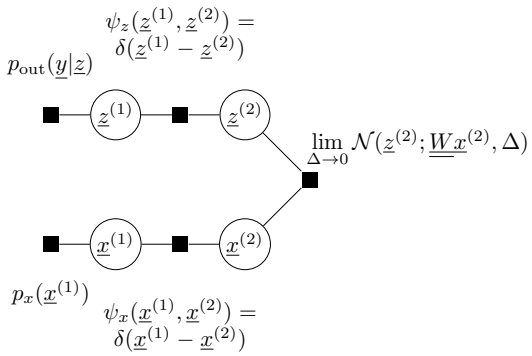


Figure 8: Factor graph representation of the GLM for the derivation of VAMP

A Vector Approximate Message Passing for the GLM

We recall here a possible derivation of G-VAMP discussed in Chapter3 Algorithm 2 . We consider a projection of the BP equations for the factor graph Figure 8.

Gaussian assumptions We start by parametrizing marginals as well as messages coming out of the Dirac factors. For $a = 1, 2$:

$$m_{x,(a)}(\underline{x}^{(a)}) = \mathcal{N}(\underline{x}^{(a)}, \hat{\underline{x}}^{(a)}, \underline{\underline{C}}^{x(a)}), \qquad m_{z,(a)}(\underline{z}^{(a)}) = \mathcal{N}(\underline{z}^{(a)}, \hat{\underline{z}}^{(a)}, \underline{\underline{C}}^{z(a)}), \tag{153}$$

and

$$\tilde{m}_{\psi_x \to \underline{x}^{(a)}}(\underline{x}^{(a)}) \propto e^{-\frac{1}{2}\underline{x}^{(a)\top} \underline{\underline{A}}_x^{(a)} \underline{x}^{(a)} + \underline{B}_x^{(a)\top} \underline{x}^{(a)}}, \tag{154}$$

$$\tilde{m}_{\psi_z \to \underline{z}^{(a)}}(\underline{z}^{(a)}) \propto e^{-\frac{1}{2}\underline{z}^{(a)\top} \underline{\underline{A}}_z^{(a)} \underline{z}^{(a)} + \underline{B}_z^{(a)\top} \underline{z}^{(a)}}. \tag{155}$$

Self consistency of the parametrizations at Dirac factor nodes Around ψ_x the message passing equations are simply

$$\tilde{m}_{\psi_x \to \underline{x}^{(2)}}(\underline{x}^{(2)}) = m_{\underline{x}^{(1)} \to \psi_x}(\underline{x}^{(2)}), \qquad \tilde{m}_{\psi_x \to \underline{x}^{(1)}}(\underline{x}^{(1)}) = m_{\underline{x}^{(2)} \to \psi_x}(\underline{x}^{(1)}) \tag{156}$$

and similarly around ψ_z. Moreover, considering that messages are marginals to which the contribution of the opposite message is retrieved we have

$$m_{\underline{x}^{(1)} \to \psi_x}(\underline{x}^{(1)}) \propto m_{x,(1)}(\underline{x}^{(1)}) / \tilde{m}_{\psi_x \to \underline{x}^{(1)}}(\underline{x}^{(1)}), \tag{157}$$

$$m_{\underline{x}^{(2)} \to \psi_x}(\underline{x}^{(2)}) \propto m_{x,(2)}(\underline{x}^{(2)}) / \tilde{m}_{\psi_x \to \underline{x}^{(2)}}(\underline{x}^{(2)}). \tag{158}$$

Combining this observation along with (156) leads to updates (3.89) and (3.85) The same reasoning can be followed for the messages around ψ_z leading to updates (3.91) and (3.87).

Input and output update functions The update functions of means and variances of the marginals are deduced from the parametrized message passing. For the variable $\underline{x}^{(1)}$ taking into account the prior p_x, the updates are very similar to GAMP input functions:

$$\hat{\underline{x}}^{(1)} \propto \int \mathrm{d}\underline{x}^{(1)}\ \underline{x}^{(1)} p_x(\underline{x}^{(1)}) \tilde{m}_{\psi_x \to \underline{x}^{(1)}}(\underline{x}^{(1)}) \tag{159}$$

$$= \frac{1}{\mathcal{Z}_x^{(1)}} \int \mathrm{d}\underline{x}^{(1)}\ \underline{x}^{(1)} p_x(\underline{x}^{(1)}) e^{-\frac{1}{2}\underline{x}^{(1)\top} \underline{\underline{A}}_x^{(1)} \underline{x}^{(1)} + \underline{B}_x^{(1)\top} \underline{x}^{(1)}} = f_1^x(\underline{B}_x^{(1)}, \underline{\underline{A}}_x^{(1)}), \tag{160}$$

$$\underline{\underline{C}}^{x(1)} = \frac{1}{\mathcal{Z}_x^{(1)}} \int \mathrm{d}\underline{x}^{(1)}\ \underline{x}^{(1)} \underline{x}^{(1)\top} p_x(\underline{x}^{(1)}) e^{-\frac{1}{2}\underline{x}^{(1)\top} \underline{\underline{A}}_x^{(1)} \underline{x}^{(1)} + \underline{B}_x^{(1)\top} \underline{x}^{(1)}} \tag{161}$$

$$- f_1^x(\underline{B}_x^{(1)}, \underline{\underline{A}}_x^{(1)}) f_1^x(\underline{B}_x^{(1)}, \underline{\underline{A}}_x^{(1)})^\top$$

$$= f_2^x(\underline{B}_x^{(1)}, \underline{\underline{A}}_x^{(1)}), \tag{162}$$

where $\mathcal{Z}_x^{(1)}$ is as usual the partition ensuring the normalization.

Similarly for the variable $\underline{z}^{(1)}$, the update functions are very similar to the GAMP output functions including the information coming from the observations:

$$\hat{\underline{z}}^{(1)} \propto \int \mathrm{d}\underline{z}^{(1)}\; p_{\mathrm{out}}(\underline{y}|\underline{z}^{(1)})\tilde{m}_{\psi_z\to\underline{z}^{(1)}}(\underline{z}^{(1)}) \tag{163}$$

$$= \frac{1}{\mathcal{Z}_z^{(1)}}\int \mathrm{d}\underline{z}^{(1)}\; p_{\mathrm{out}}(\underline{y}|\underline{z}^{(1)})e^{-\frac{1}{2}\underline{z}^{(1)\top}\underline{\underline{A}}_z^{(1)}\underline{z}^{(1)}+\underline{B}_z^{(1)\top}\underline{z}^{(1)}} = f_1^z(\underline{B}_z^{(1)},\underline{\underline{A}}_z^{(1)})\,, \tag{164}$$

$$\underline{\underline{C}}^{z(1)} = \frac{1}{\mathcal{Z}_z^{(1)}}\int \mathrm{d}\underline{z}^{(1)}\; \underline{z}^{(1)}\underline{z}^{(1)\top} p_{\mathrm{out}}(\underline{y}|\underline{z}^{(1)})e^{-\frac{1}{2}\underline{z}^{(1)\top}\underline{\underline{A}}_z^{(1)}\underline{z}^{(1)}+\underline{B}_z^{(1)\top}\underline{z}^{(1)}} \tag{165}$$

$$- f_1^z(\underline{B}_z^{(1)},\underline{\underline{A}}_z^{(1)})f_1^z(\underline{B}_z^{(1)},\underline{\underline{A}}_z^{(1)})^\top$$

$$= f_2^z(\underline{B}_z^{(1)},\underline{\underline{A}}_z^{(1)})\,. \tag{166}$$

Linear transformation For the middle factor node we consider the vector variable concatenating $\bar{\underline{x}} = [\underline{x}^{(2)}\underline{z}^{(2)}] \in \mathbb{R}^{N+M}$. The computation of the corresponding marginal with the message passing then yields

$$m_{\bar{\underline{x}}}(\bar{\underline{x}}) \propto \lim_{\Delta\to 0}\mathcal{N}(\underline{z}^{(2)};\underline{\underline{W}}\underline{x}^{(2)},\Delta\underline{\underline{I}}_M)e^{-\frac{1}{2}\underline{x}^\top\underline{\underline{A}}_x^{(2)}\underline{x}+\underline{B}_x^{(2)\top}\underline{x}}e^{-\frac{1}{2}\underline{z}^\top\underline{\underline{A}}_z^{(2)}\underline{z}+\underline{B}_z^{(2)\top}\underline{z}}. \tag{167}$$

The means of $\underline{x}^{(2)}$ and $\underline{z}^{(2)}$ are then updated through

$$\hat{\underline{x}}^{(2)},\hat{\underline{z}}^{(2)} = \underset{\underline{x},\underline{z}}{\arg\min}\left[\|\underline{\underline{W}}\underline{x}-\underline{z}\|^2/\Delta + \underline{x}^\top\underline{\underline{A}}_x^{(2)}\underline{x} - 2\underline{B}_x^{(2)\top}\underline{x} + \underline{z}^\top\underline{\underline{A}}_z^{(2)}\underline{z} - 2\underline{B}_z^{(2)\top}\underline{z}\right], \tag{168}$$

at $\Delta\to 0$. At this point it is advantageous in terms of speed to consider the singular value decomposition $\underline{\underline{W}} = \underline{\underline{U}}\underline{\underline{S}}\underline{\underline{V}}^\top$ and to simplify the form of the variance matrices by taking them proportional to the identify, i.e. $\underline{\underline{A}}_z^{(2)} = A_z^{(2)}\underline{\underline{I}}_M$ etc. Under this assumption the solution of the minimization problem is

$$\hat{\underline{x}}^{(2)} = g_1^x(\underline{B}_x^{(2)},A_x^{(2)},\underline{B}_z^{(2)},A_z^{(2)}) = \underline{\underline{V}}\,\underline{\underline{D}}\left(A_z^{(2)^{-2}}\underline{\underline{S}}\underline{\underline{U}}^\top\underline{B}_z^{(2)} + A_x^{(2)^{-2}}\underline{\underline{V}}^\top\underline{B}_x^{(2)}\right), \tag{169}$$

$$\hat{z}^{(2)} = g_1^z(\underline{B}_x^{(2)},A_x^{(2)},\underline{B}_z^{(2)},A_z^{(2)}) = \underline{\underline{W}}g_1^x(\underline{B}_x^{(2)},A_x^{(2)},\underline{B}_z^{(2)},A_z^{(2)})\,, \tag{170}$$

with $\underline{\underline{D}}$ a diagonal matrix with entries $D_{ii} = (A_z^{(2)^{-1}}S_{ii}^2 + A_x^{(2)^{-1}})^{-1}$. The scalar variances are then updated using the traces of the Jacobians with respect to the $\underline{B}^{(2)}$-s

$$\underline{\underline{C}}^{x(2)} = \frac{A_x^{(2)}}{N}\mathrm{tr}\left(\partial g_2^x/\partial\underline{B}_x^{(2)}\right)\underline{\underline{I}}_N = \frac{1}{N}\sum_{i=1}^N(A_z^{(2)^{-1}}S_{ii}^2 + A_x^{(2)^{-1}})^{-1}\underline{\underline{I}}_N \tag{171}$$

$$= g_2^x(\underline{B}_x^{(2)},A_x^{(2)},\underline{B}_z^{(2)},A_z^{(2)}) \tag{172}$$

$$\underline{\underline{C}}^{z(2)} = \frac{A_z^{(2)}}{M}\mathrm{tr}\left(\partial g_2^z/\partial\underline{B}_z^{(2)}\right)\underline{\underline{I}}_M = \frac{1}{M}\sum_{i=1}^N S_{ii}(A_z^{(2)^{-1}}S_{ii}^2 + A_z^{(2)^{-1}})^{-1}\underline{\underline{I}}_M \tag{173}$$

$$= g_2^z(\underline{B}_x^{(2)},A_x^{(2)},\underline{B}_z^{(2)},A_z^{(2)}). \tag{174}$$

B Update functions for constrained matrix factorization

We consider the interface partition function

$$\mathcal{Z}_{\mathrm{out}}^u = \int ds_\mu p_s(s_\mu)\mathcal{N}(\underline{\lambda}_\mu^1; s_\mu\underline{\omega}_\mu^1, \underline{\underline{\sigma}}_\mu^1 + s_\mu^2\underline{\underline{V}}_\mu^1), \tag{175}$$

from which we deduce the approximate posterior provided by AMP

$$m_\mu^s(s_\mu) = \frac{1}{\mathcal{Z}_{\mathrm{out}}^u}p_s(s_\mu)\mathcal{N}(\underline{\lambda}_\mu^1; s_\mu\underline{\omega}_\mu^1, \underline{\underline{\sigma}}_\mu^1 + s_\mu^2\underline{\underline{V}}_\mu^1). \tag{176}$$

The update functions derived from $\mathcal{Z}_{\mathrm{out}}^u$ are written in terms of expectations over this posterior.

Estimate $\hat{s}$ and variance C^s of the calibration variable

$$\hat{s}_\mu = \mathbb{E}_{m^s_\mu}\left[s_\mu\right], \qquad C^s_\mu = \mathbb{E}_{m^s_\mu}\left[s^2\right] - \hat{s}^2_\mu. \tag{177}$$

Output update functions $\underline{g^1_{\text{out}}}$ and $\partial_{\underline{\omega}}\underline{g_{\text{out}}}^1$

$$\underline{g^1_{\text{out}}}(\underline{\omega}^1_\mu, \underline{\underline{V}}^1_\mu, \underline{\lambda}^1_\mu, \underline{\underline{\sigma}}^1_\mu) = \partial_{\underline{\omega}} \log \mathcal{Z}^u_{\text{out}} \tag{178}$$

$$= \frac{1}{\mathcal{Z}^u_{\text{out}}} \int \mathrm{d}s_\mu \; m^s_\mu(s_\mu) \; s_\mu (s^2_\mu \underline{\underline{V}}^1_\mu + \underline{\underline{\sigma}}^1_\mu)^{-1}(\underline{\lambda}^1_\mu - s\underline{\omega}^1_\mu), \tag{179}$$

$$\partial_{\underline{\omega}}\underline{g^1_{\text{out}}}(\underline{\omega}^1_\mu, \underline{\underline{V}}^1_\mu, \underline{\lambda}^1_\mu, \underline{\underline{\sigma}}^1_\mu) = -\mathbb{E}_{m^s_\mu}\left[s^2_\mu (s^2_\mu \underline{\underline{V}}^1_\mu + \underline{\underline{\sigma}}^1_\mu)^{-1}\right] - \underline{g^1_{\text{out}}} \underline{g^1_{\text{out}}}^\top \tag{180}$$

$$+\mathbb{E}_{m^s_\mu}\left[s^2_\mu (s^2_\mu \underline{\underline{V}}^1_\mu + \underline{\underline{\sigma}}^1_\mu)^{-1}(\underline{\lambda}^1_\mu - s_\mu\underline{\omega}^1_\mu)(\underline{\lambda}^1_\mu - s_\mu\underline{\omega}^1_\mu)^\top (s^2_\mu \underline{\underline{V}}^1_\mu + \underline{\underline{\sigma}}^1_\mu)^{-1}\right].$$

Input update functions $\underline{f^u_1}$ and $\underline{\underline{f^u_2}}$

$$\underline{f^u_1}(\underline{\omega}^1_\mu, \underline{\underline{V}}^1_\mu, \underline{\lambda}^1_\mu, \underline{\underline{\sigma}}^1_\mu) = \underline{\underline{\sigma}}^1_\mu \partial_{\underline{\lambda}} \log \mathcal{Z}^u_{\text{out}} + \underline{\lambda}^1_\mu \tag{181}$$

$$= -\underline{\underline{\sigma}}^1_\mu \mathbb{E}_{m^s_\mu}\left[(s^2_\mu \underline{\underline{V}}^1_\mu + \underline{\underline{\sigma}}^1_\mu)^{-1}(\underline{\lambda}^1_\mu - s\underline{\omega}^1_\mu)\right] + \underline{\lambda}^1_\mu \tag{182}$$

$$\underline{\underline{f^u_2}}(\underline{\omega}^1_\mu, \underline{\underline{V}}^1_\mu, \underline{\lambda}^1_\mu, \underline{\underline{\sigma}}^1_\mu) = \underline{\underline{\sigma}}^1_\mu - \underline{\underline{\sigma}}^{1\,2}_\mu \mathbb{E}_{m^s_\mu}\left[(s^2_\mu \underline{\underline{V}}^1_\mu + \underline{\underline{\sigma}}^1_\mu)^{-1}\right] \tag{183}$$

$$+ \underline{\underline{\sigma}}^{1\,2}_\mu \mathbb{E}_{m^s_\mu}\left[(s^2_\mu \underline{\underline{V}}^1_\mu + \underline{\underline{\sigma}}^1_\mu)^{-1}(\underline{\lambda}^1_\mu - s_\mu\underline{\omega}^1_\mu)(\underline{\lambda}^1_\mu - s_\mu\underline{\omega}^1_\mu)^\top (s^2_\mu \underline{\underline{V}}^1_\mu + \underline{\underline{\sigma}}^1_\mu)^{-1}\right]$$

$$- \underline{\underline{\sigma}}^{1\,2}_\mu \underline{f^u_1} \underline{f^u_1}^\top.$$

These updates do not have a closed form, yet they only require a one-dimensional numerical integral over the calibration variables and therefore remain tractable. Note that here again the corresponding State Evolution can be derived by combining the results from the multi-layer version and the Cal-AMP problem. In the same manner, the online AMP and SE shall be easily written.

Bibliography

[1] Alessandro Achille and Stefano Soatto. Emergence of Invariance and Disentangling in Deep Representations. *Journal of Machine Learning Research*, 19(50):1–34, 2018.

[2] Alessandro Achille and Stefano Soatto. Information Dropout: Learning Optimal Representations Through Noisy Computation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2897–2905, dec 2018.

[3] David H Ackley, Geoffrey E Hinton, and J Sejnowski. A Learning Algorithm for Boltzmann Machine. *Cognitive Science*, 9:147–169, 1985.

[4] Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. Variational information bottleneck. *International Conference on Learning Representations (ICLR)*, pages 1–13, 2017.

[5] Daniel J. Amit, Hanoch Gutfreund, and Haim Sompolinsky. Storing infinite numbers of patterns in a spin-glass model of neural networks. *Physical Review Letters*, 55(14):1530–1533, 1985.

[6] Benjamin Aubin, Bruno Loureiro, Antoine Maillard, Florent Krzakala, and Lenka Zdeborová. The spiked matrix model with generative priors. *arXiv preprint*, 1905.12385, 2019.

[7] Benjamin Aubin, Antoine Maillard, Jean Barbier, Florent Krzakala, Nicolas Macris, and Lenka Zdeborová. The committee machine: Computational to statistical gaps in learning a two-layers neural network. In *Neural Information Processing Systems 2018*, number NeurIPS, pages 1–44, 2018.

[8] Marco Baity-Jesi, Levent Sagun, Mario Geiger, Stefano Spigler, G. Ben Arous, Chiara Cammarota, Yann LeCun, Matthieu Wyart, and Giulio Biroli. Comparing Dynamics: Deep Neural Networks versus Glassy Systems. *Proceedings of the 35th International Conference on Machine Learning*, PMLR(80):314–323, 2018.

[9] Jean Barbier, Florent Krzakala, Nicolas Macris, Léo Miolane, and Lenka Zdeborová. Phase Transitions, Optimal Errors and Optimality of Message-Passing in Generalized Linear Models. *Proceedings of the 31st Conference On Learning Theory*, PMLR 75:728–731, aug 2018.

[10] Adriano Barra, Giuseppe Genovese, Peter Sollich, and Daniele Tantari. Phase transitions in restricted Boltzmann machines with generic priors. *Physical Review E*, 96(4):1–5, 2017.

[11] Adriano Barra, Giuseppe Genovese, Peter Sollich, and Daniele Tantari. Phase diagram of restricted Boltzmann machines and generalized Hopfield networks with arbitrary priors. *Physical Review E*, 97(2), 2018.

[12] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeswar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and R Devon Hjelm. MINE: Mutual Information Neural Estimation. *International Conference on Machine Learning (ICML)*, pages 1–17, 2018.

[13] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, aug 2013.

[14] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy Layer-Wise Training of Deep Networks. *Advances in neural information processing systems*, 19(1):153, 2007.

[15] Hans Bethe. Statistical Theory of Superlattices. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 150(871):552, 1935.

[16] 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, 2017.

[17] Erwin Bolthausen. An iterative construction of solutions of the tap equations for the Sherrington-Kirkpatrick model. *Communications in Mathematical Physics*, 325(1):333–366, 2014.

[18] Ashish Bora, Ajil Jalal, Eric Price, and Alexandros G. Dimakis. Compressed Sensing using Generative Models. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 537—-546, 2017.

[19] Léon Bottou. Large-Scale Machine Learning with Stochastic Gradient Descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Physica-Verlag HD, Heidelberg, 2010.

[20] Tamara Broderick, Nicholas Boyd, Andre Wibisono, Ashia C. Wilson, and Michael I. Jordan. Streaming Variational Bayes. *Neural Information Processing Systems 2013*, pages 1–9, 2013.

[21] Giuseppe Carleo, Ignacio Cirac, Laurent Daudet, Maria Schuld, Naftali Tishby, Leslie Vogt-Maranto, and Lenka Zdeborová. Machine learning and the physical sciences. *arXiv preprint*, 1903.10563, 2019.

[22] Tommaso Castellani and Andrea Cavagna. Spin-glass theory for pedestrians. *Journal of Statistical Mechanics: Theory and Experiment*, (5):215–266, 2005.

[23] Matthew Chalk, Olivier Marre, and Gasper Tkacik. Relevant sparse codes with variational information bottleneck. In *Advances in Neural Information Processing Systems 29*, pages 1957—-1965, 2016.

[24] Kyung Hyun Cho, Tapani Raiko, and Alexander Ilin. Gaussian-Bernoulli deep Boltzmann machine. *Proceedings of the International Joint Conference on Neural Networks*, pages 1–9, 2013.

[25] Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The Loss Surfaces of Multilayer Networks. In *Artificial Intelligence and Statistics*, volume 38 PMLR, pages 192–204., 2015.

[26] Sueyeon Chung, Daniel D. Lee, and Haim Sompolinsky. Classification and Geometry of General Perceptual Manifolds. *Physical Review X*, 8(3):31003, 2018.

[27] Adam Coates, Andrew Y Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *International Conference on Artificial Intelligence and Statistics*, pages 215–223, 2011.

[28] Uri Cohen, SueYeon Chung, Daniel D Lee, and Haim Sompolinsky. Separability and Geometry of Object Manifolds in Deep Neural Networks. *bioRxiv*, 10.1101/64, 2019.

[29] Thomas M. Cover and Joy a. Thomas. *Elements of Information Theory*. Wiley-Interscience, 1991.

[30] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems*, 2(4):303–314, dec 1989.

[31] A. Decelle, G. Fissore, and C. Furtlehner. Thermodynamics of Restricted Boltzmann Machines and Related Learning Dynamics. *Journal of Statistical Physics*, 172(6):1576–1608, 2018.

[32] Aurélien Decelle, Giancarlo Fissore, and Cyril Furtlehner. Spectral dynamics of learning in restricted Boltzmann machines. *EPL (Europhysics Letters)*, 119(6):60001, sep 2017.

[33] C De Dominicis and A P Young. Weighted averages and order parameters for the infinite range Ising spin glass. *Journal of Physics A: Mathematical and General*, 16(9):2063–2075, 1983.

[34] 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, nov 2009.

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

[36] A. Dremeau, Cédric Herzet, and Laurent Daudet. Boltzmann Machine and Mean-Field Approximation for Structured Sparse Decompositions. *IEEE Transactions on Signal Processing*, 60(7):3425–3438, jul 2012.

[37] A Engel and C Van den Broeck. *Statistical Mechanics of Learning*. Cambridge University Press, Cambridge, 2001.

[38] Alyson K. Fletcher, Sundeep Rangan, and Philip Schniter. Inference in Deep Networks in High Dimensions. *2018 IEEE International Symposium on Information Theory (ISIT)*, 1(8):1884–1888, jun 2018.

[39] Marylou Gabrié, Varsha Dani, Guilhem Semerjian, and Lenka Zdeborová. Phase transitions in the q-coloring of random hypergraphs. *Journal of Physics A: Mathematical and Theoretical*, 50(50), 2017.

[40] 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 *Advances in Neural Information Processing Systems 31*, pages 1826—-1836, mar 2018.

[41] Marylou Gabrié, Andre Manoel, Gabriel Samain, and Florent Krzakala. Learning Synthetic Data https://github.com/marylou-gabrie/learning-synthetic-data, 2018.

[42] Marylou Gabrié, Eric W. Tramel, and Florent Krzakala. Training Restricted Boltzmann Machines via the Thouless-Anderson-Palmer Free Energy. *Advances in Neural Information Processing Systems 28*, pages 640—-648, jun 2015.

[43] Conrad C Galland. The limitations of deterministic Boltzmann machine learning. *Network*, 4:355–379, 1993.

[44] Elisabeth Gardner. Maximum Storage Capacity in Neural Networks. *Europhysics Letters (EPL)*, 4(4):481–485, aug 1987.

[45] Elisabeth Gardner. The space of interactions in neural network models. *Journal of Physics A: General Physics*, 21(1):257–270, 1988.

[46] Mario Geiger, Arthur Jacot, Stefano Spigler, Franck Gabriel, Levent Sagun, Stéphane D'Ascoli, Giulio Biroli, Clément Hongler, and Matthieu Wyart. Scaling description of generalization with number of parameters in deep learning. *arXiv preprint*, 1901.01608, 2019.

[47] A Georges and J S Yedidia. How to expand around mean-field theory using high-temperature expansions. *Journal of Physics A: Mathematical and General*, 24(9):2173–2192, 1999.

[48] Ziv Goldfeld, Ewout van den Berg, Kristjan Greenewald, Igor Melnyk, Nam Nguyen, Brian Kingsbury, and Yury Polyanskiy. Estimating Information Flow in Neural Networks. *arXiv preprint*, 1810.05728, 2018.

[49] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[50] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative Adversarial Networks. *Neural Information Processing Systems*, pages 1–9, 2014.

[51] Paul Hand, Oscar Leong, and Vladislav Voroninski. Phase Retrieval Under a Generative Prior. In *Neural Information Processing Systems 2018*, number NeurIPS, 2018.

[52] Paul Hand and Vladislav Voroninski. Global Guarantees for Enforcing Deep Generative Priors by Empirical Risk. In *Proceedings of the 31st Conference On Learning Theory*, volume 75, pages 970–978, 2018.

[53] Geoffrey Hinton. A Practical Guide to Training Restricted Boltzmann Machines A Practical Guide to Training Restricted Boltzmann Machines. *Computer*, 9(3):1, 2010.

[54] Geoffrey E. Hinton. Deterministic Boltzmann learning performs steepest descent in weight-space. *Neural computation*, 1(1):143–150, 1989.

[55] Geoffrey E. Hinton. Training products of experts by minimizing Contrastive divergence. *Neural computation*, 14:1771–1800, 2002.

[56] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

[57] Geoffrey E. Hinton and Ruslan R Salakhutdinov. Reducing the Dimensionality of Data with Neural Networks. *Science*, 313(5786):504–507, jul 2006.

[58] Geoffrey E Hinton and Ruslan R Salakhutdinov. Replicated softmax: an undirected topic model. In *Advances in neural information processing systems*, pages 1607–1614, 2009.

[59] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint*, 1207.0580, 2012.

[60] J J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the United States of America*, 79(8):2554–2558, 1982.

[61] Kurt Hornik. Approximation Capabilities of Multilayer Neural Network. *Neural Networks*, 4(1989):251–257, 1991.

[62] Haiping Huang. Statistical mechanics of unsupervised feature learning in a restricted Boltzmann machine with binary synapses. *Journal of Statistical Mechanics: Theory and Experiment*, 2017(5), 2017.

[63] Yukito Iba. The Nishimori line and Bayesian statistics. *Journal of Physics A: Mathematical and General*, 32(21):3875–3888, may 1999.

[64] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning*, volume 37, pages 448–456, 2015.

[65] Stanisław Jastrzębski, Zachary Kenton, Devansh Arpit, Nicolas Ballas, Asja Fischer, Yoshua Bengio, and Amos Storkey. Three Factors Influencing Minima in SGD. *arXiv preprint*, 1711.04623:1–21, 2017.

[66] Yoshiyuki Kabashima. Inference from correlated patterns: A unified theory for perceptron learning and linear vector channels. *Journal of Physics: Conference Series*, 95(1):012001, jan 2008.

[67] Yoshiyuki Kabashima, Florent Krzakala, Marc Mézard, Ayaka Sakata, and Lenka Zdeborová. Phase Transitions and Sample Complexity in Bayes-Optimal Matrix Factorization. *IEEE Transactions on Information Theory*, 62(7):4228–4265, 2016.

[68] Yoshiyuki Kabashima and Mikko Vehkapera. Signal recovery using expectation consistent approximation for linear observations. *IEEE International Symposium on Information Theory - Proceedings*, pages 226–230, 2014.

[69] Jonathan Kadmon and Haim Sompolinsky. Optimal architectures in a solvable model of deep networks. In *Advances in Neural Information Processing Systems 29*, number Nips, 2016.

[70] Hilbert J Kappen and Francisco De Borja Rodríguez. Boltzmann Machine Learning Using Mean Field Theory and Linear Response Correction. *Advances in Neural Information Processing Systems 10*, pages 280–286, 1998.

[71] Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations (ICLR)*, number Ml, pages 1–14, dec 2014.

[72] Artemy Kolchinsky and Brendan D. Tracey. Estimating mixture entropy with pairwise distances. *Entropy*, 19(7):1–18, 2017.

[73] Artemy Kolchinsky, Brendan D. Tracey, and Steven Van Kuyk. Caveats for information bottleneck in deterministic scenarios. In *International Conference on Learning Representations*, number 1, pages 1–23, 2019.

[74] Artemy Kolchinsky, Brendan D. Tracey, and David H. Wolpert. Nonlinear Information Bottleneck. *arXiv preprint*, 1705.02436, may 2017.

[75] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models Principles and Techniques*. MIT Press, 2009.

[76] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical Review E - Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics*, 69(6):16, 2004.

[77] Florent Krzakala, Marc Mézard, Francois Sausset, Yifan Sun, and Lenka Zdeborová. Probabilistic reconstruction in compressed sensing: algorithms, phase diagrams, and threshold achieving matrices. *J. Stat. Mech*, 2012.

[78] Hugo Larochelle and Yoshua Bengio. Classification using discriminative restricted Boltzmann machines. In *Proceedings of the 25th international conference on Machine learning*, pages 536–543. ACM, 2008.

[79] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[80] Thibault Lesieur, Florent Krzakala, and Lenka Zdeborova. MMSE of probabilistic low-rank matrix estimation: Universality with respect to the output channel. *2015 53rd Annual Allerton Conference on Communication, Control, and Computing, Allerton 2015*, pages 680–687, 2016.

[81] Thibault Lesieur, Florent Krzakala, and Lenka Zdeborová. Constrained Low-rank Matrix Estimation: Phase Transitions, Approximate Message Passing and Applications. *Journal of Statistical Mechanics: Theory and Experiment*, 2017(7):1–64, 2017.

[82] Stefano Sarao Mannelli, Florent Krzakala, Pierfrancesco Urbani, and Lenka Zdeborová. Passed & Spurious: analysing descent algorithms and local minima in spiked matrix-tensor model. In *Proceedings of the 36th International Conference on Machine Learning,*, pages PMLR 97:4333–4342, 2019.

[83] Andre Manoel, Marylou Gabrié, and Florent Krzakala. Deep Neural Networks Entropy from Replicas https://github.com/sphinxteam/dnner, 2018.

[84] Andre Manoel, Florent Krzakala, Marc Mézard, and Lenka Zdeborová. Multi-layer generalized linear estimation. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pages 2098–2102. IEEE, jun 2017.

[85] Andre Manoel, Florent Krzakala, Eric W. Tramel, and Lenka Zdeborová. Streaming Bayesian inference: Theoretical limits and mini-batch approximate message-passing. *55th Annual Allerton Conference on Communication, Control, and Computing, Allerton 2017*, 2018-Janua(i):1048–1055, 2018.

[86] V A Marčenko and L A Pastur. DISTRIBUTION OF EIGENVALUES FOR SOME SETS OF RANDOM MATRICES. *Mathematics of the USSR-Sbornik*, 1(4):457–483, apr 1967.

[87] Marc Mézard. Mean-field message-passing equations in the Hopfield model and its generalizations. *Physical Review E*, 95(2):1–15, 2017.

[88] Marc Mézard and Andrea Montanari. *Information, Physics, and Computation*. Oxford University Press, 2009.

[89] Marc Mézard and Giorgio Parisi. The Bethe lattice spin glass revisited. *The European Physical Journal B*, 20(2):217–233, mar 2001.

[90] Marc Mézard, Giorgio Parisi, and Miuel Virasoro. *Spin Glass Theory and Beyond*, volume 9 of *World Scientific Lecture Notes in Physics*. WORLD SCIENTIFIC, nov 1986.

[91] Marc Mézard, Giorgio Parisi, and Riccardo Zecchina. Analytic and Algorithmic Solution of Random Satisfiability Problems. *Science*, 297(5582):812–815, aug 2002.

[92] Thomas P Minka. A family of algorithms for approximate Bayesian inference. *PhD Thesis*, 2001.

[93] Dustin G. Mixon and Soledad Villar. SUNLayer: Stable denoising with generative networks. *arXiv preprint*, 1803.09319, 2018.

[94] Marcin Moczulski, Misha Denil, Jeremy Appleyard, and Nando de Freitas. ACDC: A Structured Efficient Linear Layer. *International Conference on Learning Representations (ICLR)*, pages 1–12, nov 2016.

[95] Rémi Monasson and Riccardo Zecchina. Weight Space Structure and Internal Representations: a Direct Approach to Learning and Generalization in Multilayer Neural Networks. *Physical Review Letters*, 75(12):2432–2435, 1995.

[96] Rémi Monasson and Riccardo Zecchina. Learning and Generalization Theories of Large Committee-Machines. *Modern Physics Letters B*, 09(30):1887–1897, 2004.

[97] Grégoire Montavon and Klaus Robert Müller. Deep Boltzmann machines and the centering trick. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7700 LECTU:621–637, 2012.

[98] Radford M. Neal. Connectionist learning of belief networks. *Artificial Intelligence*, 56(1):71–113, 1992.

[99] Radford M. Neal. Annealed importance sampling. *Statistics and Computing*, 11(2):125–139, 2001.

[100] H. Chau Nguyen, Riccardo Zecchina, and Johannes Berg. Inverse statistical problems: from the inverse Ising problem to data science. *Advances in Physics*, 66(3):197–261, 2017.

[101] Hidetoshi Nishimori. *Statistical Physics of Spin Glasses and Information Processing: An Introduction*. Clarendon Press, 2001.

[102] Morteza Noshad, Alfred O. Hero, Yu Zeng, and Alfred O. Hero. Scalable Mutual Information Estimation using Dependence Graphs. *arXiv preprint*, 1801.09125, 2018.

[103] Manfred Opper, Burak Çakmak, and Ole Winther. A theory of solving TAP equations for Ising models with general invariant random matrices. *Journal of Physics A: Mathematical and Theoretical*, 49(11):114002, mar 2016.

[104] Manfred Opper and David Haussler. Calculation of the learning curve of Bayes optimal classification algorithm for learning a perceptron with noise. In *COLT '91 Proceedings of the fourth annual workshop on Computational learning theory*, pages Pages 75–87, 1991.

[105] Manfred Opper and David Saad. *Advanced mean field methods: Theory and practice*. MIT press, 2001.

[106] Manfred Opper and Ole Winther. Mean field approach to bayes learning in feed-forward neural networks. *Physical Review Letters*, 76(11):1964–1967, 1996.

[107] Manfred Opper and Ole Winther. Adaptive and self-averaging Thouless-Anderson-Palmer mean-field theory for probabilistic modeling. *Physical Review E*, 64(5):056131, oct 2001.

[108] Manfred Opper and Ole Winther. Tractable approximations for probabilistic models: The adaptive Thouless-Anderson-Palmer mean field approach. *Physical Review Letters*, 86(17):3695–3699, 2001.

[109] Manfred Opper and Ole Winther. Expectation consistent free energies for approximate inference. *Advances in Neural Information Processing Systems*, 17:1001–1008, 2005.

[110] Parthe Pandit, Mojtaba Sahraee, Sundeep Rangan, and Alyson K. Fletcher. Asymptotics of MAP Inference in Deep Networks. *arXiv preprint*, 1903.01293, 2019.

[111] Giorgio Parisi and Marc Potters. Mean-field equations for spin models with orthogonal interaction matrices. *Journal of Physics A: Mathematical and General*, 28(18):5267–5285, sep 1995.

[112] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems*. Elsevier, 1988.

[113] Carsten Peterson and James R. Anderson. A mean field theory learning algorithm for neural networks. *Complex systems*, 1:995–1019, 1987.

[114] T Plefka. Convergence condition of the TAP equation for the infinite-ranged Ising spin glass model. *Journal of Physics A: Mathematical and General*, 15(6):1971–1978, 1982.

[115] Sundeep Rangan. Generalized Approximate Message Passing for Estimation with Random Linear Mixing. In *2011 IEEE International Symposium on Information Theory Proceedings*, pages 2168–2172. IEEE, jul 2011.

[116] Sundeep Rangan, Philip Schniter, and Alyson Fletcher. On the convergence of approximate message passing with arbitrary matrices. *IEEE International Symposium on Information Theory - Proceedings*, pages 236–240, 2014.

[117] Sundeep Rangan, Philip Schniter, and Alyson K. Fletcher. Vector approximate message passing. In *2017 IEEE International Symposium on Information Theory (ISIT)*, volume 1, pages 1588–1592. IEEE, jun 2017.

[118] Galen Reeves. Additivity of information in multilayer networks via additive Gaussian noise transforms. *55th Annual Allerton Conference on Communication, Control, and Computing, Allerton 2017*, 2018-Janua:1064–1070, 2018.

[119] Galen Reeves and Henry D Pfister. The replica-symmetric prediction for compressed sensing with Gaussian matrices is exact. In *2016 IEEE International Symposium on Information Theory (ISIT)*, pages 665–669. IEEE, jul 2016.

[120] D J Rezende, S Mohamed, and D Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *Proceedings of The 31st International Conference in Machine Learning, Beijing China*, 32:1278–, 2014.

[121] S Robbins, H Monro. A stochastic approximation method. *Statistics*, pages 102–109, 1951.

[122] Mehdi S. M. Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, Sylvain Gelly, Olivier Bachem, and Mario Lucic. Assessing Generative Models via Precision and Recall. *Neural Information Processing Systems 2018*, (NeurIPS):1–10, 2018.

[123] Ruslan Salakhutdinov and Geoffrey Hinton. Deep Boltzmann Machines. *Artificial Intelligence and Statistics*, 5(2):448–455, 2009.

[124] Ruslan Salakhutdinov and Geoffrey Hinton. An Efficient Learning Procedure for Deep Boltzmann Machines. *Neural Computation*, 24(8):1967–2006, 2012.

[125] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. Restricted Boltzmann machines for collaborative filtering. In *Proceedings of the 24th international conference on Machine learning*, pages 791–798. ACM, 2007.

[126] Ruslan Salakhutdinov and Iain Murray. On the quantitative analysis of Deep Belief Networks. *Proceedings of the 25th international conference on Machine learning*, pages 872–879, 2008.

[127] Shibani Santurkar, Dimitris Tsipras, and Andrew Ilyas. How Does Batch Normalization Help Optimization ? In *Neural Information Processing Systems*, 2018.

[128] Andrew M Saxe, Yamini Bansal, Joel Dapello, Madhu Advani, Artemy Kolchinsky, Brendan D Tracey, and David D Cox. On The Information Bottleneck Theory of Deep Learning. *International Conference on Learning Representations 2018*, pages 1–27, 2018.

[129] 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. IEEE, nov 2016.

[130] Christophe Schülke, Francesco Caltagirone, Florent Krzakala, and Lenka Zdeborová. Blind Calibration in Compressed Sensing using Message Passing Algorithms. *Advances in Neural Information Processing Systems 26*, pages 1–9, 2013.

[131] Christophe Schülke, Francesco Caltagirone, and Lenka Zdeborová. Blind sensor calibration using approximate message passing. *Journal of Statistical Mechanics: Theory and Experiment*, 2015(11):P11013, nov 2015.

[132] Christopher J. Shallue, Jaehoon Lee, Joseph Antognini, Jascha Sohl-Dickstein, Roy Frostig, and George E. Dahl. Measuring the Effects of Data Parallelism on Neural Network Training. *arXiv preprint*, 1811.03600, 2018.

[133] Claude E Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(July 1928):379–423, 1948.

[134] David Sherrington and Scott Kirkpatrick. Solvable Model of a Spin-Glass. *Physical Review Letters*, 35(26):1792–1796, dec 1975.

[135] Takashi Shinzato and Yoshiyuki Kabashima. Perceptron capacity revisited: classification ability for correlated patterns. *Journal of Physics A: Mathematical and Theoretical*, 41(32):324013, aug 2008.

[136] Takashi Shinzato and Yoshiyuki Kabashima. Learning from correlated patterns by simple perceptrons. *Journal of Physics A: Mathematical and Theoretical*, 42(1):015005, jan 2009.

[137] Ravid Shwartz-Ziv and Naftali Tishby. Opening the Black Box of Deep Neural Networks via Information. *arXiv preprint*, 1703.00810, 2017.

[138] Umut Simsekli, Levent Sagun, and Mert Gurbuzbalaban. A Tail-Index Analysis of Stochastic Gradient Noise in Deep Neural Networks. 2019.

[139] Paul Smolensky. Information Processing in Dynamical Systems: Foundations of Harmony Theory. In *Parallel Distributed Proceuing: Explorations in the Microstructure of Cognition. Volume I: Foundations*, chapter 6. MIT Press, Cambridge, Massachusetts, 186.

[140] Juyong Song, Matteo Marsili, and Junghyo Jo. Resolution and relevance trade-offs in deep learning. *Journal of Statistical Mechanics: Theory and Experiment*, 2018(12):123406, dec 2018.

[141] Stefano Spigler, Mario Geiger, Stéphane D'Ascoli, Levent Sagun, Giulio Biroli, and Matthieu Wyart. A jamming transition from under- to over-parametrization affects loss landscape and generalization. *arXiv preprint*, 1810.09665, 2018.

[142] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

[143] Michel Talagrand. The Parisi formula. *Annals of Mathematics*, 163(1):221–263, jan 2006.

[144] D. J. Thouless, P. W. Anderson, and R. G. Palmer. Solution of 'Solvable model of a spin glass'. *Philosophical Magazine*, 35(3):593–601, 1977.

[145] Tijmen Tieleman. Training restricted Boltzmann machines using approximations to the likelihood gradient. *ICML; Vol. 307*, page 7, 2008.

[146] Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. In *Proc. of the 37-th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377, 1999.

[147] Naftali Tishby and Noga Zaslavsky. Deep Learning and the Information Bottleneck Principle. *2015 IEEE Information Theory Workshop, ITW 2015*, 2015.

[148] Eric W. Tramel, Angélique Drémeau, and Florent Krzakala. Approximate message passing with restricted Boltzmann machine priors. *Journal of Statistical Mechanics: Theory and Experiment*, 2016(7):073401, jul 2016.

[149] Eric W. Tramel, Marylou Gabrié, and Florent Krzakala. Boltzmann.jl, https://github.com/sphinxteam/Boltzmann.jl, 2015.

[150] Eric W. Tramel, Marylou Gabrié, Andre Manoel, Francesco Caltagirone, and Florent Krzakala. Deterministic and Generalized Framework for Unsupervised Learning with Restricted Boltzmann Machines. *Physical Review X*, 8(4):041006, oct 2018.

[151] Eric W. Tramel, Andre Manoel, Francesco Caltagirone, Marylou Gabrié, and Florent Krzakala. Inferring sparsity: Compressed sensing using generalized restricted Boltzmann machines. In *2016 IEEE Information Theory Workshop (ITW)*, pages 265–269. IEEE, sep 2016.

[152] J. Tubiana and R. Monasson. Emergence of Compositional Representations in Restricted Boltzmann Machines. *Physical Review Letters*, 118(13), 2017.

[153] Rene Vidal, Joan Bruna, Raja Giryes, and Stefano Soatto. Mathematics of Deep Learning. *arXiv preprint*, 1712.04741, dec 2017.

[154] Martin J. Wainwright and Michael I. Jordan. Graphical Models, Exponential Families, and Variational Inference. *Foundations and Trends® in Machine Learning*, 1:1–305, 2008.

[155] Chuang Wang, Hong Hu, Yue M. Lu, and John A Paulson. A Solvable High-Dimensional Model of GAN. *arXiv preprint*, 1805.08349, 2018.

[156] Timothy L.H. Watkin, Albrecht Rau, and Michael Biehl. The statistical mechanics of learning a rule. *Reviews of Modern Physics*, 65(2):499–556, 1993.

[157] Max Welling and Ge Hinton. A new learning algorithm for mean field Boltzmann machines. *Artificial Neural Networks—ICANN 2002*, pages 351–357, 2002.

[158] Zichao Yang, Marcin Moczulski, Misha Denil, Nando De Freitas, Alex Smola, Le Song, and Ziyu Wang. Deep Fried Convnets. *Proceedings of the IEEE International Conference on Computer Vision*, 2015 Inter:1476–1483, 2015.

[159] Francesco Zamponi. Mean field theory of spin glasses. *arXiv preprint*, 1008.4844, 2010.

[160] Lenka Zdeborová and Florent Krzakala. Statistical physics of inference: Thresholds and algorithms. *Advances in Physics*, 65(5):453–552, sep 2016.

[161] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding Deep Learning Requires ReThinking Generalization. *International Conference on Learning Representations 2017*, pages 1–15, 2017.

[162] Shengjia Zhao, Jiaming Song, and Stefano Ermon. InfoVAE: Information Maximizing Variational Autoencoders. *arXiv preprint*, 1706.02262, 2017.

RÉSUMÉ

Les algorithmes d'apprentissage automatique utilisant des réseaux de neurones profonds ont récemment révolutionné l'intelligence artificielle. Malgré l'engouement suscité par leurs diverses applications, les excellentes performances de ces algorithmes demeurent largement inexpliquées sur le plan théorique. Ces problèmes d'apprentissage sont décrits mathématiquement par de très grands ensembles de variables en interaction, difficiles à manipuler aussi bien analytiquement que numériquement. Cette multitude est précisément le champ d'étude de la physique statistique qui s'attelle à comprendre, originellement dans les systèmes naturels, comment rendre compte des comportements macroscopiques à partir de cette complexité microscopique. Dans cette thèse nous nous proposons de mettre à profit les progrès récents des méthodes de champ moyen de la physique statistique des systèmes désordonnés pour dériver des approximations pertinentes dans ce contexte. Nous nous appuyons sur les équivalences et les complémentarités entre les algorithmes de passage de message, les développements haute température et la méthode des répliques. Cette stratégie nous mène d'une part à des contributions pratiques pour l'apprentissage non supervisé des machines de Boltzmann. Elle nous permet d'autre part de contribuer à des réflexions théoriques en considérant le paradigme du professeur-étudiant pour modéliser des situations d'apprentissage. Nous développons une méthode pour caractériser dans ces modèles l'évolution de l'information au cours de l'entraînement, et nous proposons une direction de recherche afin de généraliser l'étude de l'apprentissage bayésien des réseaux de neurones à une couche aux réseaux de neurones profonds.

MOTS CLÉS

Systèmes désordonnés, inférence, apprentissage automatique, passage de message, méthode des répliques

ABSTRACT

Machine learning algorithms relying on deep neral networks recently allowed a great leap forward in artificial intelligence. Despite the popularity of their applications, the efficiency of these algorithms remains largely unexplained from a theoretical point of view. The mathematical descriptions of learning problems involves very large collections of interacting random variables, difficult to handle analytically as well as numerically. This complexity is precisely the object of study of statistical physics. Its mission, originally pointed towards natural systems, is to understand how macroscopic behaviors arise from microscopic laws. In this thesis we propose to take advantage of the recent progress in mean-field methods from statistical physics to derive relevant approximations in this context. We exploit the equivalences and complementarities of message passing algorithms, high-temperature expansions and the replica method. Following this strategy we make practical contributions for the unsupervised learning of Boltzmann machines. We also make theoretical contributions considering the teacher-student paradigm to model supervised learning problems. We develop a framework to characterize the evolution of information during training in these model. Additionally, we propose a research direction to generalize the analysis of Bayesian learning in shallow neural networks to their deep counterparts.

KEYWORDS

Disordered systems, inference, message passing, replica method